Quantifying the Quality Attenuation of WiFi

1st Bjørn Ivar Teigen
Department of Informatics
University of Oslo
Oslo, Norway
bjornite@ifi.uio.no

2nd Neil Davies
Predictable Network Solutions Limited
Stonehouse, United Kingdom
neil.davies@pnsol.com

3rd Kai Olav Ellefsen
Department of Informatics
University of Oslo
Oslo, Norway
kaiolae@ifi.uio.no

4th Tor Skeie
Department of Informatics
University of Oslo
Oslo, Norway
tskeie@ifi.uio.no

5th Jim Torresen
Department of Informatics
University of Oslo
Oslo, Norway
jimtoer@ifi.uio.no

***Abstract*—WiFi is one of the most widely deployed networking technologies, and understanding WiFi performance is therefore of great importance. The WiFi MAC layer sometimes introduces significant and variable delays. No existing models of the WiFi protocol describe WiFi performance in terms of complete latency distributions. In this work, we present a novel model of WiFi performance. We explicitly define our model in terms of the latency introduced at each step in the protocol state machine, and the model produces complete latency distributions. We validate the model by comparing its outputs to previous modeling work and real-world measurements. Finally, we use our results to quantify the latency distribution of WiFi as a function of the duration of transmit opportunities and the number of stations competing for the channel. Quantifying this relation represents a significant improvement in our understanding of WiFi performance that would not be possible with existing models.**

***Index Terms*—Multimedia and real-time communication, Performance evaluation of networks, ΔQ, Quality attenuation, Ergodicity networking, WiFi, 802.11**

I. Introduction

WiFi is one of the most widely deployed ways for computers and mobile devices to connect to the Internet. An estimated *16 billion* devices are now WiFi compatible, and this number grows faster every year. Knowledge of the performance of WiFi networks is therefore of great importance to our understanding of the end-to-end performance of applications on the Internet.

WiFi stations compete for access to the radio channel using a *distributed coordination function* that coordinates the channel access of multiple stations in a decentralized way. It is defined by the IEEE 802.11 working group and published as a standard that all WiFi stations must be compatible with [1]. The performance of the distributed coordination function (DCF) has been thoroughly documented in the literature [2]–[9]. The most recent WiFi protocol versions have introduced more efficient ways to use the WiFi channel, such as MU-MIMO and OFDMA. However, these protocol improvements run on top of the DCF. The WiFi DCF is still the basic method for assigning channel resources in WiFi networks [10].

The performance of the WiFi DCF has been extensively studied, so why write yet another paper on the subject? The reason is that we have decided to use an unconventional definition of performance. Having made that choice, we could not find any studies of WiFi performance compatible with our definition. Our preferred definition of performance is that of Thompson and Davies [11], known as *quality attenuation*, or ΔQ for short. The quality attenuation metric combines latency and packet loss into a single variable where packet loss is modeled as infinite latency. The motivation for choosing this performance metric is that ΔQ enables us to reason about network performance in ways that are not possible with conventional performance metrics. For example, it deals in distributions, allowing us to reason about tail risks that average measures fail to capture. Values of ΔQ can be composed both sequentially and in parallel. ΔQ values can also describe and reason about first-to-finish and last-to-finish synchronizations [12]. In addition, we can define a partial order of ΔQ values, which allows us to conclude that one network connection is better than another in a rigorously defined sense [13]. A partial order establishes a notion of comparing two entities which includes the idea of ‘incomparable’ in addition to the usual comparisons equal (=), less than (<), and greater than (>).

Composing network performance measurements is not well defined for throughput, arguably the most popular network performance metric. Other commonly used performance metrics, such as the average and standard deviation of latency, provide some of the benefits of ΔQ because they can be added and subtracted. However, average and standard deviation do not adequately describe general latency distributions. ΔQ describes the full distribution of latency by treating it as a random variable and including packet loss. The ΔQ performance metric has proven useful as a design and reasoning tool in real-world scenarios. Haeri et al. [12] describe how ΔQ was used to design the Cardano blockchain, and Teigen et al. [14] used it to uncover protocol violations in the WiFi implementation on the Raspberry Pi 4B. More details on ΔQ are described in section III.

The question we seek to answer in this work is ”How much quality attenuation does a WiFi link introduce?”. We propose two novel contributions toward answering this question. First, we develop and validate a novel WiFi distributed coordination function (DCF) model based on ΔQ. The model is validated by comparing its outputs to real-world WiFi performance measurements in a testbed. We also verify that the average latency values produced by our model match those derived by Markov chain methods. Second, we use the model to describe how the quality attenuation of WiFi links depends on the duration of transmit opportunities (TxOP duration) and the number of competing stations. To the best of our knowledge, this is the first work describing the ΔQ performance of the WiFi DCF. Quantifying this relation represents a significant improvement in our understanding of WiFi performance that would not be possible with existing models.

Section II lays out the most relevant related work on WiFi modeling, and section III covers the details of the ΔQ metric. We explain our method and its application to WiFi in section IV. In section V, we describe the testbed and the experiments used to validate the model. Section V also presents the results and the discussion of the validation experiments. In section VI, we use our model to explore how a WiFi link's ΔQ changes as a function of the duration of transmit opportunities and the number of competing stations. Our results are discussed in section VII. Finally, we conclude the work in section VIII.

II. Background

Bianchi [2] models the WiFi distributed coordination function (DCF) using a Markov chain. In doing so, Bianchi makes a few key simplifications. The most important simplification is to abstract away the details of delays. The value of the back-off counters defines a time step in Bianchi's model. That is to say; the model does not separate the case where the medium is idle from the case in which the station (STA) has to wait for another transmission to complete before the back-off counter is decremented. The time steps are defined in terms of the model state, not how much actual time has passed. Defining the time steps using back-off counter values is a useful simplification for a Markov chain analysis, but it comes at the cost of discarding timing information. Bianchi also points this out [2, Section IV, A].

Tinnirello et al. [5] extend the methodology of Bianchi [2]. Here, a Markov chain is solved for the steady-state *distribution of back-off timer values*. This approach was chosen to better model the different channel access probabilities of the Wireless Multi-Media (WMM) extension of WiFi. Still, this method is also closer to modeling latency distributions. Tinnirello's model still makes simplifications that hide latency information because the model does not deal with differences in transmission times due to different data rates. Heusse [7] shows that differences in data rates are very important for WiFi performance.

We have drawn attention to a few examples, but several other authors have also based their analysis of WiFi performance on the method first demonstrated by Bianchi [15]–[17]. These papers all report steady-state results and measure performance as aggregate throughput. Aggregate throughput is equivalent to average head-of-line latency under the assumption that all stations get an equal share of the WiFi channel and use the same coding rate [18].

From our review of previous work, we see a need for a more detailed analysis of the latency and packet loss performance of WiFi. We do not dispute the usefulness of throughput as a performance metric. Instead, we emphasize that it is important to be mindful of its limitations. One of these limitations is that long-term average throughput ignores short-term deviations from the long-term average behavior. When these short-term deviations from average behavior are important, it is easy to draw misleading conclusions. Empirical measurements [8] show that WiFi latency follows a heavy-tail distribution and that the WiFi latency of a single packet is sometimes large enough to potentially affect application performance and user experience [19], [20]. It is, therefore, interesting to describe WiFi latency with more fidelity than a long-term average latency value. We address this by developing a novel model of the 802.11 WiFi protocol that allows for the exact computation of the statistical distribution of latency and packet loss.

III. Quality attenuation

Quality attenuation is a network quality metric that captures the latency and packet loss performance of packet-switched networks. It has been developed through several decades of academic work [11]–[13], [21], [22], but it is not widely known in the research community. Therefore, we give a brief description here. The quality attenuation metric combines latency and packet loss into a single variable where packet loss is modeled as infinite latency. Equation 1 formally defines a quality attenuation value as a probability density function $P(t)$ paired with a real number $P(\infty)$ such that $0 \leq P(\infty) \leq 1$. $P(t)$ describes the probability density over all possible latency values, and $P(\infty)$ describes the probability of packet loss (we can think of packet loss as infinite latency). We use "ΔQ" to abbreviate quality attenuation. A ΔQ value can be plotted as a cumulative density function (CDF) that never goes above $1 - P(\infty)$.

One of the most useful features of ΔQ is that values can be composed both sequentially and in parallel. Equation 2 defines how two quality attenuation values are sequentially composed. The operation consists of convolving the probability densities and computing the sequential probability of packet loss [13]. We use $\otimes$ to denote the sequential composition of ΔQ values. Suppose we have a good description of the quality attenuation of each link along a network path. In that case, sequential compositionality allows us to reason about how the end-to-end path will perform. An example of a real-world use-case for sequential composition is shown in Fig. 1.

$$\Delta Q := [P(t),\, P(\infty)],\, t \in \mathbf{R}^+ \tag{1}$$

$$
\begin{aligned}
&\Delta Q^A \otimes \Delta Q^B = \\
&[P_A(t),\, P_A(\infty)] \otimes [P_B(t),\, P_B(\infty)] := \\
&[\int_0^t P_A(\tau) P_B(t-\tau)\, d\tau,\, P_A(\infty) + (1 - P_A(\infty)) P_B(\infty)]
\end{aligned} \tag{2}
$$

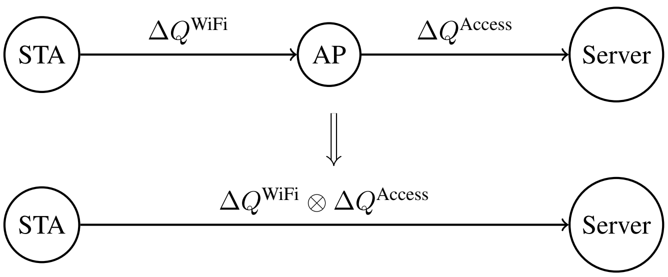

Figure 1. Practical example of the convolution operator

We can also define the parallel composition of ΔQ values. The parallel composition can be thought of as modeling uncertainty about the network state. For instance, we can describe a situation where a WiFi link has either zero, one, or two stations competing for the WiFi channel, as illustrated in Fig. 3. Suppose we do not know how many stations are competing for the channel but have some idea of the probability of zero, one, or two competitors, respectively. In that case, the link's total ΔQ can be computed by calculating the parallel composition of the ΔQ of each number of competitors. The parallel composition operation on ΔQ values extends the notion of a mixture distribution [23] to improper random variables. We use $\oplus_p$ to denote parallel composition. Fig. 2 is a diagram showing parallel composition, and equation 3 formally defines the operation.

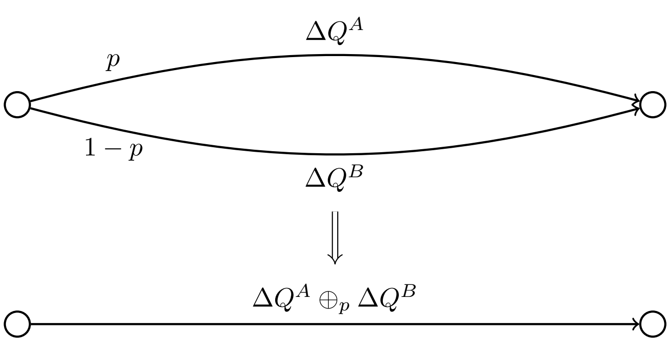

Figure 2. Probabilistic choice

$$
\begin{aligned}
&\Delta Q^A \oplus_p \Delta Q^B = \\
&[P_A(t),\, P_A(\infty)] \oplus_p [P_B(t),\, P_B(\infty)] := \\
&[p P_A(t) + (1-p) P_B(t),\, p P_A(\infty) + (1-p) P_B(\infty)]
\end{aligned} \tag{3}
$$

We can also define a partial order of ΔQ values. The meaning of $\Delta Q^A < \Delta Q^B$ is defined by equation 4. This partial

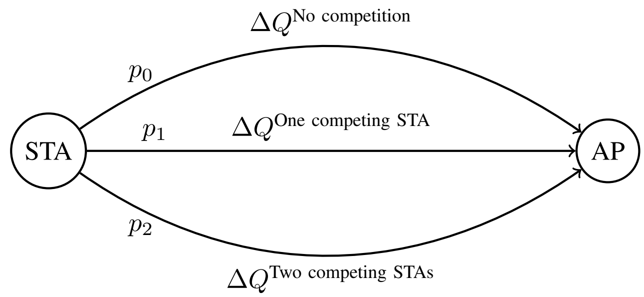

Figure 3. A quality attenuation model of a WiFi link with zero, one or two competing stations

order has a neat geometrical interpretation. When plotted as CDFs, $\Delta Q^A < \Delta Q^B$ implies that the CDF of ΔQ^A is above and to the left of ΔQ^B and the graphs never cross. Thompson and Davies [11] use this partial order of ΔQ values to show how the ΔQ of an end-to-end network path can be related to application performance over that path. This can be done by defining a ΔQ budget for the application and checking whether the delivered ΔQ is less than the application budget. Suppose the network delivers a ΔQ less than the application requires. In that case, we can confidently say that the network is not causing any issues for the application (barring pathological cases where the application cannot handle a network that is too responsive). Equation (4) can be confusing; It expresses that the CDF of $P_A(t)$ is greater than the CDF of $P_B(t)$ for all values of d, which means $P_A(t)$ has lower latency for any given percentile value. A larger CDF for all values of d along with a lower probability of loss means lower quality attenuation, hence $\Delta Q^A < \Delta Q^B$.

$$
\begin{aligned}
&\Delta Q^A < \Delta Q^B = \\
&[P_A(t),\, P_A(\infty)] < [P_B(t),\, P_B(\infty)] := \forall\, d \in \mathbf{R}^+ : \\
&\int_0^d P_A(t)\, dt > \int_0^d P_B(t)\, dt \wedge P_A(\infty) < P_B(\infty)
\end{aligned} \tag{4}
$$

IV. Model Description

This section describes the structure of our model and how we evaluate it. We then describe how we implement the features of the WiFi distributed coordination function in the model.

At a high level of abstraction, our model is a directed graph of communicating stochastic processes. The model is evaluated using the discrete event simulation algorithm described by Cassandras and Lafortune [24, Section 10.2]. The nodes represent queues, links, or computational steps and are modeled as stochastic processes describing the distribution of time until the next event generated by that node. The edges of the graph represent interfaces where the output of one stochastic process is input to another stochastic process. In our model, each event carries a data packet, and each event involves forwarding the data packet from one queue, link, or processing step to another. As an example of a node, consider a link that deterministically takes exactly 1 ms to transmit a

Algorithm 1 Algorithm for evaluating the quality attenuation WiFi model

```
time ← 0
eventqueue ← empty list
trace ← empty list                                  ▷ To record the event history
for n in nodes do
    eventqueue.add(n.get_next_event(time))          ▷ A pair (e, t)
eventqueue.sort()                                   ▷ Earliest event first
while not done do
    e, t ← pop(eventqueue)
    trace.append((e, t))
    time ← t
    for n in nodes affected by e do
        must_resample ← n.handle_event(e, time)
        if must_resample then
            e_n ← eventqueue.lookup(n)
            eventqueue.remove(e_n)
            eventqueue.add(n.get_next_event(time))
    eventqueue.sort()
```

packet. Upon receiving a packet at time t, the node representing this link will generate an event that forwards the packet to a receiver scheduled at time $t + 1ms$.

The simulation algorithm of Cassandras and Lafortune [24, Section 10.2] is described in algorithm 1 for completeness. When the algorithm terminates, the list "trace" contains a timestamped record of the events. We use this record to compute the latency distribution of head-of-line packets in the individual stations by taking the difference between the time at which a packet becomes the head-of-line packet and the time at which it is received at the other end of the WiFi link. By sampling many packets, we can build a distribution (ΔQ) of the time it takes for a packet to be serviced once it reaches the head of the line.

We represent the WiFi channel as a single node in the model graph. We first describe how we handle enqueuing packets at the WiFi stations (the node's handle_event() function) and then describe how we model the operation of the WiFi DCF (the node's get_next_event() function).

1) handle_event(): When a new packet arrives at one of the WiFi stations, the packet is either enqueued or dropped depending on the state of the appropriate queue. If the queue is empty and the relevant back-off timer is zero when the packet arrives, we check whether enough time has passed [1, Section 10.3.3] since the trailing edge of the last transmission. If so, the carrier sense mechanism reports that the frequency is idle, and the packet is transmitted immediately. If the frequency is not idle, then the back-off procedure for that station is triggered. The back-off procedure selects a random number in the range $[0, CW_{min}]$ and assigns that number to a *back-off counter* [1, Section 10.2.2]. "CW" here stands for contention window.

2) get_next_event(time): Our WiFi node implements the rules by which the WiFi DCF controls access to the channel. The WiFi DCF is "clocked" by the trailing edges of busy periods on the channel and the duration "slot time" (see Table I). The state of a WiFi channel with n competing stations is modeled as an allocation of each of the n stations to a back-off counter and a retry counter value, and each station has its own set of internal queues. The WiFi DCF defines rules for updating this state. We call these updates state transitions. When looking at the state of the WiFi DCF between slot boundaries, there must be an ongoing transition, and at one of the slot-time boundaries, only three cases are possible:

1) No stations have a back-off counter value of zero, so all eligible stations decrease their back-off counter value by one. Which stations are eligible depends on AIFSN values [1, Section 10.3.2.3], WMM traffic classes [1, Section 10.22.2] and how much time has passed since the falling edge of the last transmission. The duration of this state transition is one slot time.
2) Exactly one station has a back-off counter value of zero. This station successfully transmits. The state transition lasts for the time required to transmit the packet and receive the following ACK. The transmitting station is then finished sending its packet. The station leaves the system if its queue is empty. Otherwise, it triggers the back-off procedure with the packet at the head of its queue. The transmitting station resets its retry counter to zero. The remaining stations hold their back-off counters constant for the transmission duration.
3) More than one station has a back-off counter value of zero. Several stations initiating a transmission at the same time cause a collision, spending the amount of time required for the longest colliding transmission (including ACK) to complete. All the colliding stations then increase their retry counter by one and select a random back-off counter value from the range $(0, CW(r))$, where r is the new retry counter value at each station [1, Section 10.2.2]. All stations not involved in the collision keep their back-off counter values constant for the duration of the collision.

When the get_next_event(time) function of the WiFi DCF node is called, we iterate the DCF process until a packet is successfully transmitted. This generates an event that passes the transmitted packet to its receiver. If all queues become empty before a successful transmission occurs, the WiFi node generates a null event timestamped at infinity, which amounts to an idle waiting state.

V. Model validation

In this section, we compare the output of our model to the results of Bianchi [2], and we compare model outputs to real-world WiFi measurements in a testbed.

A. Experiments and results

To replicate the results of Bianchi [2] we evaluate our model using stations that always have a packet to send. We perform the evaluation using the same parameters as Bianchi [2, Table 2], shown in table I. Figure 4 shows results for total

Parameter	Bianchi	Testbed
Slot time (μs)	50	20
SIFS (μs)	28	10
DIFS (μs)	128	50
PHY Header (bits)	128	192
MAC Header (bits)	272	284
ACK (μs)	240	304
Base rate (Mbit/s)	1	1
CW_{min}	15	15
CW_{max}	1023	1023
Maximum retries	6	6
packet size	1023	N/A

Table I
PARAMETERS USED FOR COMPARISON TO THE RESULTS OF BIANCHI AND FOR COMPARISON TO TESTBED MEASUREMENTS

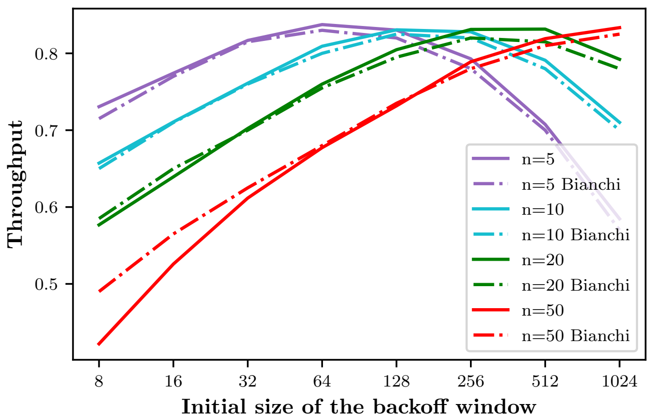

Figure 4. System throughput as a function of initial back-off window size.

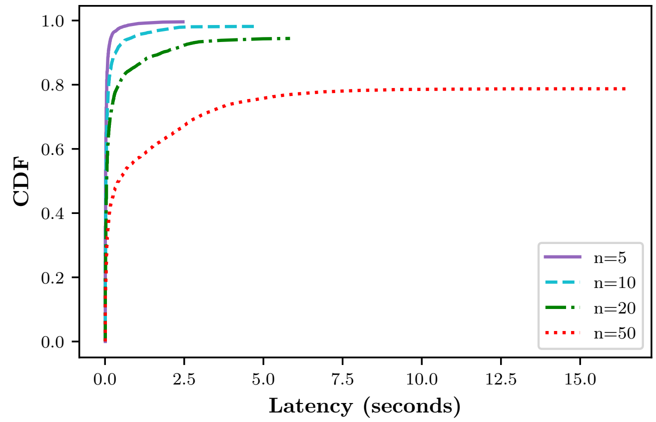

Figure 5. CDF for latency and packet loss with initial back-off window size of 8

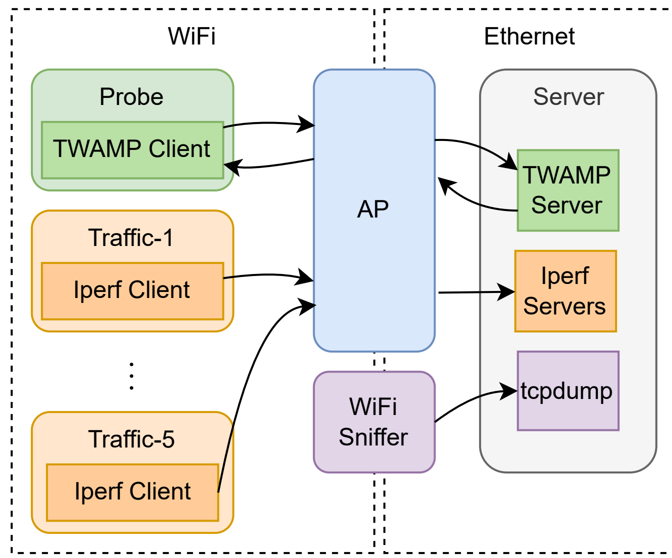

Figure 6. The testbed setup with two traffic generators. The validation runs make use of five traffic generators.

system throughput as a function of initial back-off window size, compared to results from [2, Figure 9].

Figure 5 shows our model predictions for latency and packet loss performance for some of the scenarios we use to compare our model to Bianchi's results. Bianchi's model does not predict distributions, so we can not compare predictions in this case. Packet loss is pictured as the distance from 1 on the y-axis. Note that the results show the latency of a head-of-line packet, so queuing delays and potential packet loss due to full buffers will come in addition to the delays shown here. The shape of the latency distributions in figure 5 is instructive. The shape of the CDFs shows that when a WiFi link is saturated with many stations, some packets receive relatively good service, while some packets receive very bad service indeed. The distribution of head-of-line latency has a long tail. We believe this is an effect of the exponential back-off of the WiFi protocol, where those packets that are unlucky enough to suffer several collisions get extremely large delays.

To check that the quality attenuation predicted by our model is not unrealistic, we compare the model predictions to real-world WiFi performance measurements in a testbed. We first describe the testbed and then compare model outputs with testbed measurements.

The testbed consists of seven Raspberry Pi 4B machines and a WiFi access point, arranged as shown in figure 6. One machine, called the Server, is connected to the WiFi AP with a 1 Gbit/s Ethernet cable. The Server runs a Two-Way Active Measurement Protocol (TWAMP) server, five iperf servers, and records WiFi packets using a WiFi adapter (Alfa AWUS036ACH) with a Realtek RTL8812AU WiFi 5 (802.11ac) chipset. Five machines, called traffic-1 through traffic-5, run UDP uploads to the iperf servers. The final Raspberry Pi runs the TWAMP client, our source of latency measurements. All the Raspberry Pis run NixOS [25] and are configured through NixOps. Table I shows the WiFi parameters for the testbed setup. The Raspberry Pi 4B machines use the Broadcom BCM4345/6 WiFi chipset with brcmfmac43455 firmware version 7.45.229.

We aim to design experiments that will invalidate our model if it outputs incorrect latency distributions. To this end, we developed a set of test scenarios that vary the number of competing stations and the packet arrival rate at each station. These scenarios span a wide range of network conditions and

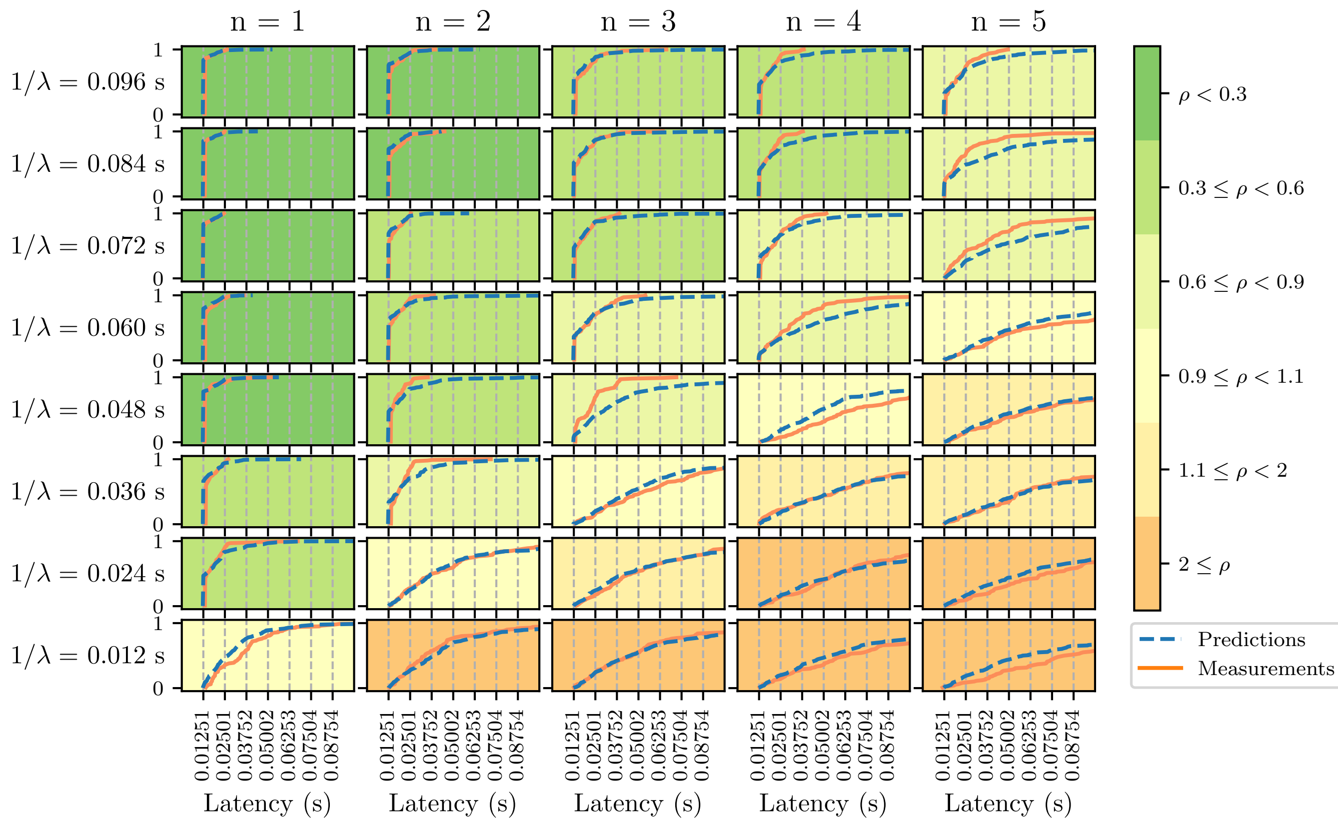

Figure 7. Model predictions compared to testbed measurements for various combinations of n stations and average packet arrival rate λ at each station. Per-packet transmission time (including the WiFi ACK) is 12.506 ms.

capture the effects of interactions between competing stations and overall channel saturation.

We independently vary two different parameters in the validation scenarios. First, we vary the number of participating stations, each running a single UDP upload towards the iperf server. Second, we vary the throughput of the UDP uploads. The packet size is kept constant at 1470 bytes, and the WiFi rate is 1 Mbit/s. The per-packet transmission time, including the ACK, is 12.506 ms. The WiFi configuration of both the testbed and the simulations use the parameters listed in Table I. We have verified that the testbed access point is configured correctly by inspecting beacon frames captured by the WiFi sniffer. All traffic, including the TWAMP measurement packets, are tagged as Best Effort using DSCP, and RTS/CTS is off. Both testbed experiments and simulations are run for 150 seconds of real and simulated time, respectively, and collect 300 samples of head-of-line quality attenuation in that period.

There are a few differences between the model and the testbed. In the model, we use Poisson processes to generate the packet arrivals. In the testbed, packets are spaced with an (almost) constant delay because we use iperf3 to generate traffic. It is hard to faithfully replicate the traffic patterns of iperf3 in the model because it depends on details like clock granularity, CPU scheduling, and, most importantly, the timing of triggering the iperf3 commands via SSH on each of the traffic generators. Therefore, we believe that attempting to model the iperf3 traffic pattern exactly is as likely to introduce new artifacts as it is to remove them. Consequently, we have chosen to stick to a Poisson arrival pattern in the simulation. Another source of prediction errors is the possibility of random noise or interference from other 2.4GHz radios causing packet retransmissions that are not due to collisions with other traffic in the testbed. To mitigate this, we ensure that no other WiFi networks are within range on the same channel in the testbed location.

Figure 7 shows the results for each validation scenario from both the testbed and the model. Each plot shows the cumulative density function (CDF) of latency in the interval from 0 to 0.1 seconds. The blue dotted line shows model predictions, and testbed measurements are shown in orange. The number of active traffic generators (n) varies from 1 to 5, and the average time between packet arrival times varies from 12 ms to 96ms in steps of 12 ms. The vertical grid lines are spaced one packet transmission time apart to aid interpretation. The plot background is colored to indicate how heavily loaded the WiFi channel is in each case. ρ (see the color bar) represents the fraction of time needed to complete transmissions of all the arriving traffic (i.e., the sum of traffic arriving at all stations), assuming perfect time-division multiplexing of the channel. When $\rho << 1$, the WiFi channel is empty some of the time,

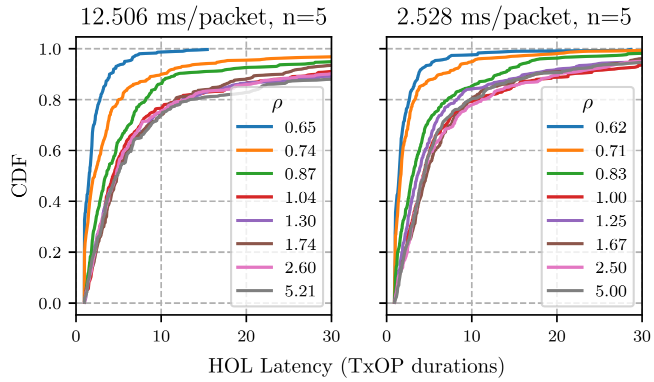

Figure 8. WiFi CDFs normalized to packet transmission durations

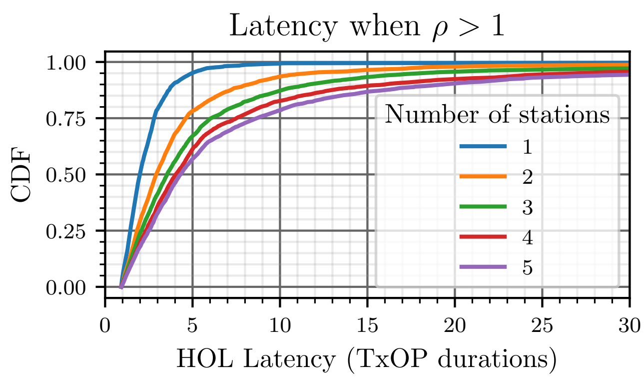

Figure 9. WiFi CDFs normalized to packet transmission durations

and we expect that the queues in each station also become empty occasionally so that fewer than n stations compete for access to the channel at any single moment on average. When $\rho \geq 1$, the WiFi channel can not service all arriving traffic. In this case, we expect the queue in each station to be full most of the time. Therefore, when $\rho \geq 1$, all of the n stations are competing for access to the WiFi channel almost all the time.

B. *Discussion of the model validation*

We consider figure 7 convincing evidence that our WiFi model is valid. The prediction error is much smaller than the packet transmission time for most scenarios. Predictions are less accurate when ρ is slightly less than one, which we believe is because the packet arrival patterns in the testbed are different from the model. Poisson arrivals are phase-less, meaning there is no correlation between subsequent inter-arrival times. The iperf3 arrivals in the testbed have close to constant inter-arrival times, and, therefore, the different traffic sources have the potential to become out of phase with each other. If the packet arrivals are out of phase, they are biased toward a perfect time-division multiplexing of the WiFi channel. That might explain why the measured latency is smaller than the model predicts. If this is the correct explanation, we expect the effect to be most noticeable when ρ is slightly less than one, and it should

n	Mean	STD	50th percentile	90th percentile
1	2.36	1.56	1.96	3.85
2	4.42	7.37	2.91	8.04
3	6.33	13.27	3.54	11.49
4	8.07	17.16	4.02	15.44
5	9.78	22.71	4.29	18.73

Table II
LATENCY STATISTICS MEASURED IN TXOP DURATIONS

disappear completely when $\rho \geq 1$ because then queues start to fill, and any out-of-phase arrival pattern no longer matters. This is indeed what the results show.

Because each frame can be retried up to 6 times before it is lost (see table I), the probability of packet loss is very small. We did not observe any loss of TWAMP packets in the testbed during the validation runs. Simulation of the transit of 3000 packets for the highest number of stations ($n = 5$) yielded a loss rate of 0.10%, so observing no loss in the experimental case (300 recorded packets) is not statistically inconsistent. Frame loss (and hence retries) is a direct consequence of collisions. We know our model captures this phenomenon because collisions, and the resulting back-off behavior, are a significant cause of WiFi latency. The delay distributions would be substantially different if we did not model collisions accurately. Therefore, we can be reasonably confident that the model accurately captures the processes that lead to packet loss.

VI. ΔQ ANALYSIS OF THE WIFI DCF

In this section, we use our model to explore and quantify the impact of some of the factors that affect quality attenuation in the WiFi DCF.

The left plot in figure 8 shows the model predictions from the rightmost column of figure 7 (where $n = 5$) collapsed into a single chart. We have normalized the x-axis to count the number of packet transmission durations (TxOP duration). The right plot of figure 8 shows model predictions for a different combination of packet size and WiFi rate (3788 bytes and 14.4 Mbit/s), which yields a TxOP duration of 2.528 milliseconds. We chose 2.528 ms because it is the default maximum TxOP duration for best-effort traffic in the 802.11ac standard, the most widely deployed version of the WiFi protocol [1, Table 9-137].

We want to illustrate two things with figure 8. First, notice that the left and right plots are very similar. This similarity shows that the quality attenuation of a WiFi link scales linearly with the TxOP duration. Linear scaling with TxOP duration is expected considering that the waiting time for idle slots (typically 9 or 20 μs) is very short compared to the packet transmission times (typically $>$ 1ms). Second, we also observe that the head-of-line quality attenuation is very similar for all cases where $\rho > 1$. This is because any load above capacity is queued within each station, and this does not affect the head-of-line quality attenuation.

Figure 9 shows the TxOP-normalized ΔQ CDFs when $\rho > 1$ for different numbers of competing stations. Table II summarizes some of the results shown in figure 9.

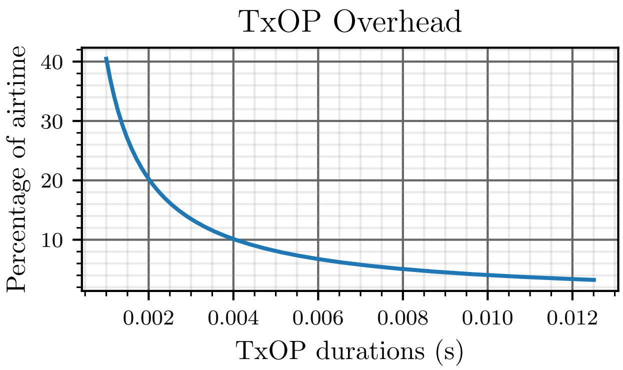

Figure 10. How TxOP efficiency scales with the size of TxOPs

Figure 9 suggests that reducing the TxOP duration will lead to much lower head-of-line quality attenuation. However, the minimum per-transmission overhead of WiFi is constant. Reducing the TxOP duration reduces the maximum throughput a WiFi channel can support. Figure 10 shows the percentage of overhead in each TxOP as a function of the TxOP duration. The TxOP overhead is the time it takes to transmit the minimum per-transmission overhead (PHY and MAC headers, the SIFS between a packet and the following ACK, and the time needed to transmit the ACK) divided by the total duration of each TxOP. Our calculations for figure 10 use the parameters in table I.

We can leverage these results to explore trade-offs in the WiFi DCF. We can use figures 9 and 10 to quantify the cost and benefit of tuning the maximum TxOP size in a WiFi network. The default TxOP limit for best-effort traffic in the 802.11ac standard is 2.528 milliseconds [1, Table 9-137]. Figure 10 shows that this gives us approximately 16.2% overhead. If the maximum TxOP duration is reduced to 1.25 ms, the overhead grows to 32.3%. This increase in overhead buys us a 50% reduction in head-of-line latency. This trade-off might be well worth considering depending on the relative importance of latency vs. throughput for a specific WiFi network. We should also note that the parameters of table I give a relatively large overhead compared to default values for 802.11ac and 802.11ax. The trade-off is more favorable for newer protocol versions because overhead has been reduced while the relationship between TxOP duration, number of stations, and ΔQ remains the same.

Figures 7 and 9 also show how the model can be used to quantify the effects of reducing the number of competing stations (n). Given a network with n stations, we can shape the traffic of each station so that not all n stations compete for WiFi resources all of the time. This can reduce head-of-line quality attenuation significantly, and figure 7 effectively shows examples for specific traffic shaping scenarios using Poisson arrival patterns.

VII. Discussion

Figures 9 and 10 maps out what we believe to be the most critical dimensions of the space of performance trade-offs for the WiFi DCF. It is not surprising that more stations and a larger TxOP duration increase the ΔQ of a WiFi link, but this is the first time the magnitude of this increase has been calculated. We can now make well-informed decisions when optimizing WiFi performance because we can leverage our knowledge of the relationship between TxOP duration, the number of competing stations, and quality attenuation. A WiFi access point can influence both the number of stations and the TxOP duration, so these results provide a clear path towards optimizing ΔQ in real-world WiFi networks.

One of the main limitations of our approach is that the results are long-term averages of ΔQ for static network configurations. In real-world networks, the ΔQ of a link typically varies with time so that samples taken within a limited time interval are correlated. Ignoring short-term correlations is a limitation of all steady-state approaches, and so it also applies to most other WiFi modeling work. Nevertheless, addressing this limitation remains an exciting direction for future study.

VIII. Conclusion and future work

We have built and validated a model of WiFi performance, which allows us to compute the quality attenuation of a WiFi link. Using the newly validated model, we quantified how the performance of a WiFi link depends on the duration of transmissions and the number of competing stations. To the best of our knowledge, this is the first structured approach to describing the quality attenuation of WiFi. Our results lay the foundations for analyzing WiFi networks with tools and methods based on the quality attenuation formalism. Our results provide a clear path toward optimizing the quality attenuation of WiFi networks, but more work is needed in this direction.

References

[1] The Institute of Electrical and Electronics Engineers, "IEEE Standard for Information technology – Telecommunications and information exchange between systems LAN and MAN – Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-2016," pp. 1–3534, 2013. [Online]. Available: http://ieeexplore.ieee.org/document/7786995/

[2] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 3 2000.

[3] G. Bianchi, L. Fratta, and M. Oliveri, "Performance evaluation and enhancement of the CSMA/CA MAC protocol for 802.11 wireless LANs," in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, vol. 2. IEEE, 1996, pp. 392–396.

[4] P. E. Engelstad and O. N. Østerbø, "Non-saturation and saturation analysis of IEEE 802.11e EDCA with starvation prediction," in *ACM MSWiM 2005 - Proceedings of the Eighth ACM Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, vol. 2006, 2006, pp. 224–233.

[5] I. Tinnirello and G. Bianchi, "Rethinking the IEEE 802.11e EDCA performance modeling methodology," *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 540–553, 2010. [Online]. Available: http://ieeexplore.ieee.org.

[6] I. Tinnirello, S. Choi, and Y. Kim, "Revisit of RTS/CTS exchange in high-speed IEEE 802.11 networks," in *Proceedings - 6th IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, WoWMoM 2005*, 2005, pp. 240–248.

[7] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *Proceedings - IEEE INFOCOM*, vol. 2, 2003, pp. 836–843.

[8] T. Høiland-Jørgensen, M. Kazior, D. Täht, P. Hurtig, and A. Brunstrom, "Ending the Anomaly: Achieving Low Latency and Airtime Fairness in WiFi," in *Proceedings of the 2017 USENIX Conference on Usenix Annual Technical Conference*, 2 2017, p. 139–151.

[9] M. Seyedebrahimi, F. Bouhafs, A. Raschella, M. Mackay, and Q. Shi, "Fine-grained radio resource management to control interference in dense Wi-Fi networks," *IEEE Wireless Communications and Networking Conference, WCNC*, 2017.

[10] E. Khorov, A. Kiryanov, A. Lyakhov, and G. Bianchi, "A tutorial on IEEE 802.11ax high efficiency WLANs," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 197–216, 1 2019.

[11] P. Thompson and N. Davies, "Towards a RINA-Based Architecture for Performance Management of Large-Scale Distributed Systems," *Computers*, vol. 9, no. 2, p. 53, 6 2020. [Online]. Available: https://www.mdpi.com/2073-431X/9/2/53

[12] S. Haeri, P. Thompson, N. Davies, P. Van Roy, K. Hammond, and J. Chapman, "Mind Your Outcomes: The ∆QSD Paradigm for Quality-Centric Systems Development and Its Application to a Blockchain Case Study," *Computers*, vol. 11, no. 3, p. 45, 3 2022.

[13] Broadband Forum, "TR-452.1 Quality Attenuation Measurement Architecture and Requirements," Tech. Rep., 2020. [Online]. Available: https://www.broadband-forum.org/download/TR-452.1.pdf

[14] B. I. Teigen, N. Davies, K. O. Ellefsen, T. Skeie, and J. Torresen, "Protocol Violations in the WiFi Distributed Coordination Function of Raspberry Pi 4," in *17th Conference on Wireless On-Demand Network Systems and Services, WONS 2022*. Institute of Electrical and Electronics Engineers Inc., 2022.

[15] R. Draves, J. Padhye, and B. Zill, "Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks," *Proceedings of the 10th annual international conference on Mobile computing and networking - MobiCom '04*, 2004.

[16] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma, "Performance of reliable transport protocol over IEEE 802.11 wireless LAN: Analysis and enhancement," *Proceedings - IEEE INFOCOM*, vol. 2, pp. 599–607, 2002.

[17] P. Nayak, M. Garetto, and E. W. Knightly, "Modeling Multi-User WLANs under Closed-Loop Traffic," *IEEE/ACM Transactions on Networking*, vol. 27, no. 2, pp. 763–776, 4 2019.

[18] G. Bianchi and I. Tinnirello, "Remarks on IEEE 802.11 DCF performance analysis," *IEEE Communications Letters*, vol. 9, no. 8, pp. 765–767, 8 2005.

[19] C. Pei, Y. Zhao, G. Chen, R. Tang, Y. Meng, M. Ma, K. Ling, and D. Pei, "WiFi can be the weakest link of round trip network latency in the wild," in *Proceedings - IEEE INFOCOM*, vol. 2016-July. IEEE, 7 2016.

[20] J. Kennedy, G. Armitage, and J. Thomas, "Household bandwidth and the 'need for speed' Evaluating the impact of active queue management for home internet traffic," *Australian Journal of Telecommunications and the Digital Economy*, vol. 5, no. 2, pp. 113–130, 6 2017.

[21] N. Davies, J. Holyer, and P. Thompson, "A Queueing Theory Model that Enables Control of Loss and Delay of Trafic at a Network Switch," University of Bristol, England, Tech. Rep. November, 1999.

[22] D. C. Reeve, "A New Blueprint for Network QoS," Ph.D. dissertation, University of Kent, Kent, UK, 8 2003. [Online]. Available: http://www.cs.kent.ac.uk/pubs/2003/1892

[23] B. G. Lindsay, *Mixture Models*, ser. NSF-CBMS Regional Conference Series in Probability and Statistics. Haywood CA and Alexandria VA: Institute of Mathematical Statistics and American Statistical Association, 1995. [Online]. Available: https://projecteuclid.org/euclid.cbms/1462106013

[24] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*. Springer US, 2008.

[25] "NixOS - NixOS Linux." [Online]. Available: https://nixos.org/