Is Our Model for Contention Resolution Wrong?

Confronting the Cost of Collisions*

William C. Anderton
Department of Computer Science and Engineering
Mississippi State University
MS, USA, 39762
wca36@msstate.edu

Maxwell Young
Department of Computer Science and Engineering
Mississippi State University
MS, USA, 39762
myoung@cse.msstate.edu

ABSTRACT

Randomized binary exponential backoff (BEB) is a popular algorithm for coordinating access to a shared channel. With an operational history exceeding four decades, BEB is currently an important component of several wireless standards.

Despite this track record, prior theoretical results indicate that under bursty traffic (1) BEB yields poor makespan and (2) superior algorithms are possible. To date, the degree to which these findings manifest in practice has not been resolved.

To address this issue, we examine one of the strongest cases against BEB: n packets that simultaneously begin contending for the wireless channel. Using Network Simulator 3, we compare against more recent algorithms that are inspired by BEB, but whose makespan guarantees are superior. Surprisingly, we discover that these newer algorithms significantly underperform.

Through further investigation, we identify as the culprit a flawed but common abstraction regarding the cost of collisions. Our experimental results are complemented by analytical arguments that the number of collisions – and not solely makespan – is an important metric to optimize. We argue that these findings have implications for the design of contention-resolution algorithms.

CCS CONCEPTS

• **Networks** → **Network algorithms**; • **Theory of computation** → **Distributed algorithms**;

KEYWORDS

Contention resolution; backoff; wireless; collisions

1 INTRODUCTION

Randomized binary exponential backoff (***BEB***) plays a critical role in coordinating access by multiple devices to a shared communication medium. Given its importance, BEB has been studied at length and is known to yield good throughput under well-behaved traffic [3, 4, 15, 36, 37, 39, 44, 58, 66].

In contrast, when traffic is "bursty", BEB is suspected to perform sub-optimally. Under a ***single batch*** of n ***packets*** that simultaneously begin contending for the channel, Bender et al. [10] prove that BEB has $\Theta(n \log n)$ ***makespan*** (the amount of time until all packets are successfully transmitted). More recent algorithms have been proposed [10–13, 31, 32, 40] with improved makespan regardless of the traffic type.

Together, these results beg the question: *How do newer algorithms compare to BEB in practice?* Here, we make progress towards an answer by restricting ourselves to bursty traffic – in particular, the simplest instance of such traffic: a single burst (***batch***) of packets. This is a prominent case where BEB is anticipated to do poorly, and it should be a straightforward (if laborious) exercise to discover which of the following situations is true:

(1) A newer contention-resolution algorithm outperforms BEB.
(2) BEB outperforms newer contention-resolution algorithms.

Interestingly, neither of these outcomes is very palatable. In one form or another, BEB has operated in networks for over four decades and it remains an essential ingredient in several wireless standards. Bursty traffic can arise in practice [70, 74] and its impact has been examined [14, 19, 35, 64]. If (1) holds, then BEB is potentially in need of revision and the ramifications of this are hard to overstate.

Conversely, if (2) holds, then theoretical results are not translating into improved performance. At best, this is a matter of asymptotics. At worst, this indicates a problem with the abstract model upon which newer results are based. In this latter case, it is important to understand what assumptions are faulty so that the abstract model can be revised.

1.1 A Common Model

Given n stations, the problem of ***contention resolution*** addresses the amount of time until any one of the stations transmits alone. A natural consideration is the time until a subset of k stations each transmits alone; this often falls under the same label, but is also referred to as k-selection (see [6]).

We focus on the case of $k = n$. Here, much of the algorithmic work shares an abstract model. Three common assumptions are:

- *A0.* Time is discretized into **slots**, each of which may accommodate a packet.
- *A1.* If a single packet is transmitted in a slot, the packet **succeeds**, but **failure** occurs if two or more packets transmit simultaneously due to a **collision**.
- *A2.* The failure of a transmission is known to the sender with negligible delay beyond the single slot in which the failure occurred.

Assumption A0 is near universal, but technically inaccurate for reasons discussed in Section 1.2. To summarize, slots in a contention window are used to obtain ownership of the channel. However, transmission of the full packet may occur past this contention-window slot while all other stations pause their execution. Therefore, this assumption is sufficiently close to reality that we should not expect performance to deviate greatly as a result.

Examples of assumption A1 abound (for example [6, 11–13, 20, 32, 48]), although variations exist. A compelling alternative is the signal-to-noise-plus-interference (SINR) model [7, 55] which is less strict about failure in the event of simultaneous transmissions. Another model that has received attention is the affectance model [43]. Nevertheless, these all share the reasonable assumption that simultaneous transmissions may negatively impact performance.

Assumption A2 is also widely adopted (see the same examples for A1) and implicitly addresses two quantities that affect performance: the time to transmit a packet, and the time to receive any feedback on success or failure. Assigning a delay of 1 slot to these quantities admits a model where the problem of contention resolution is treated separately from the functionality for collision detection. Such functionality is provided by a medium access control (MAC) protocol – of which the contention-resolution algorithm is only one component – and is not captured by A2.

1.1.1 Demonstrating a Flawed Assumption. Our main thesis is that A2 is flawed in the wireless setting; that is, the cost of failure is far more significant than the abstract model acknowledges. This is not a matter of minor adjustments to the assumption, or an artifact of hidden constants in the algorithms examined. Rather, the way in which failures – in particular, collisions – are detected cannot be isolated from the problem of contention resolution.

Several corollaries follow from this thesis, all indicating that accounting for such failures should be incorporated into algorithm design. For a range of wireless settings, contention-resolution algorithms that ignore this will likely not perform as advertised when deployed within a MAC protocol (see Section 5.2). We demonstrate this for the popular IEEE 802.11g standard.

1.2 Overview of BEB in IEEE 802.11g

To understand our findings, it is helpful to summarize IEEE 802.11g and how BEB operates within it. However, outside of this section and the description of our experimental setup, discussion of such aspects and terminology is kept to a minimum. Throughout, we will often use interchangeably the terms *packets* and *stations* depending on the context; the two uses are equivalent given that each station seeks to transmit a single packet in the single-batch case.

Exponential backoff [53] is a widely deployed algorithm for distributed multiple access. Informally, a backoff algorithm operates
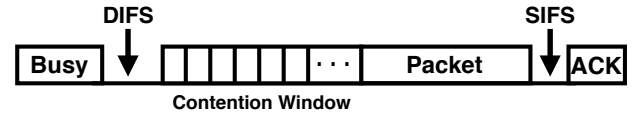

Figure 1: Illustration of DCF.

over a **contention window (CW)** wherein each **station** makes a single randomly-timed access attempt. In the event of two or more simultaneous attempts, the result is a collision and none of the stations succeed. Backoff seeks to avoid collisions by dynamically increasing the contention-window size such that stations succeed.

IEEE 802.11 handles contention resolution via the **distributed coordination function (DCF)** which employs BEB; as the name suggests, successive CWs double in size under BEB. The operation of DCF is summarized as follows. Prior to transmitting data, a station first senses the channel for a period of time known as a **distributed inter-frame space (DIFS)**. If the channel is not in use over the DIFS, the station transmits its data; otherwise, it waits until the current transmission finishes and then initiates BEB.

For a contention window of size CW, a timer value is selected uniformly at random from $[0, CW - 1]$. So long as the channel is sensed to be idle, the timer counts down and, when it expires, the station transmits. However, if at any prior time the channel is sensed busy, BEB is *paused for the duration of the current transmission*, and then resumed (not restarted) after another DIFS.

After a station transmits, it awaits an **acknowledgement (ACK)** from the receiver. If the transmission was successful, then the receiver waits for a short amount of time known as a **short inter-frame space (SIFS)** – of shorter duration than a DIFS – before sending the ACK. Upon receiving an ACK, the station learns that its transmission was successful. Otherwise, the station waits for an ACK-timeout duration before *concluding that a collision occurred*.

This series of actions is referred to as **collision detection**; the cost of which lies at the heart of our argument. If a collision is detected, then the station must attempt a retransmission via the same process with its CW doubled.

Figure 1 illustrates the operation of DCF. Note that both the transmission of data and the acknowledgement process occur "outside" of the backoff component of DCF. Yet, the focus of many algorithmic results is solely on the slots of this backoff component.

Finally, RTS/CTS (request-to-send and clear-to-send) is an optional mechanism. Informally, a station will send an RTS message and await an CTS message from the receiver prior to transmitting its data. Due to increased overhead, RTS/CTS is often only enabled for large packets. Therefore, we focus on the case where RTS/CTS is disabled, although our experiments show that our findings continue to hold when this mechanism is used (see Section 3.2).

2 EXPERIMENTAL SETUP

We employ Network Simulator 3 (NS3) [28] which is a widely used network simulation tool in the research community [72]. Our experimental setup is described here for the purposes of reproducibility.[1]

Our reasons for using NS3 are twofold. First, wireless communication is difficult to model and employing NS3 helps allay concerns that our findings are an artifact of poorly-modeled wireless effects.

---

[1]Our simulation code and data will be made available at www.maxwellyoung.net.

Parameter	Value
Data rate	54 Mbits/sec
Wireless specification	802.11g
Slot duration	$9\mu s$
SIFS	$16\mu s$
DIFS	$34\mu s$
ACK timeout	$75\mu s$
Preamble	$20\mu s$
Transport layer protocol	UDP
Packet overhead	64 bytes
Contention-window size min.	1
Contention-window size max.	1024
RTS/CTS	Off

Table 1: Parameter values used in our experiments.

Second, given the assumptions upon which contention-resolution algorithms are based, NS3 can reveal whether we are being led astray by an assumption that appears reasonable, but results in a *significant* discrepancy between theory and practice.

Table 1 provides our experimental parameters. Path-loss models with default parameters are known to be faithful [67] and, therefore, our experiments employ the log-distance propagation loss model in NS3. For transmission and reception of packets (frames), we use the YANS [50] module which provides an additive-interference model.

At the MAC layer, we make use of IEEE 802.11g and we implement changes to the growth of the contention window based on the algorithms we investigate. All experiments use IPv4 and UDP.

The amount of overhead for each packet is 64-bytes: 8 bytes for UDP, 20 bytes for IP, 8 bytes for an LLC/Snap header, and 28 bytes of additional overhead at the MAC layer.

The duration of an acknowledgement (ACK) timeout is specified by the most recent IEEE 802.11 standard[2] to be roughly the sum of a SIFS ($16\mu s$), standard slot time ($9\mu s$), and preamble ($20\mu s$); a total of $45\mu s$. However, in practice, this is subject to tuning. In our experiments, an ACK-timeout below $55\mu s$ gave markedly poor performance; there is insufficient time for the ACK before the sender decides to retransmit. We use the default value of $75\mu s$ in NS3 since this is the same order of magnitude and performs well.

In our experiments, n stations are placed in a $40m \times 40m$ grid, and they are laid out starting at the south-west corner of the grid moving left to right by 1 meter increments, and then up when the current row is filled. A wireless access point (AP) is located (roughly) at the center of the grid. We do not simulate additional terrain or environmental phenomena; our goal is to test the performance under ideal conditions without complicating factors.

Our experiments are computationally intensive. Computing resources are provided by the High Performance Computing Collaboratory (HPC[2]) at Mississippi State University. We employ four identical Linux (CentOS) systems, each with 16 processors (Intel Xeon CPU E5-2690, 2.90GHz) and 396 GB of memory.

---

[2]This is a large document; please see Section 10.3.2.9, page 1317 of [2].

When a station wishes to transmit a packet:
- Set the window size $W = 1$.
- Repeat until the packet is successfully transmitted:
  - Choose a slot t in the window uniformly at random. Try to transmit in slot t.
  - If the transmission failed, then: (i) wait until the end of the window, and (ii) set $W \leftarrow (1 + r)W$.

Figure 2: Generic algorithm for LLB, LB, and BEB where $r = 1/\lg\lg W$, $r = 1/\lg W$, and $r = 1$, respectively.

3 A SINGLE BATCH

We examine a single batch of n packets that simultaneously begin their contention for the channel. As algorithmic competitors for BEB, we take Log-Backoff (LB), LogLog-Backoff (LLB) from [10] and Sawtooth-Backoff (STB) from [33, 41]. Both LLB and LB are closely related to BEB in that they execute using a CW that increases in size monotonically. The typical pseudocode for algorithms LLB, LB, and BEB is provided in Figure 2.

In contrast, STB is non-monotonic and executes over a doubly-nested loop. The outer loop sets the current window size W to be double that used in the preceding outer loop; this is like BEB. Additionally, for each such W, the inner loop executes over $\lg W$ windows of size $W, W/2, W/4, ..., 2$ and, for each window, a slot is chosen uniformly at random for the packet to transmit; this is the "backon" component of STB.

Algorithm	Contention-Window Slots
BEB	$\Theta(n \log n)$
Log-Backoff	$\Theta\left(\frac{n \log n}{\log \log n}\right)$
LogLog-Backoff	$\Theta\left(\frac{n \log \log n}{\log \log \log n}\right)$
Sawtooth-Backoff	$\Theta(n)$

Table 2: Known guarantees on the number of CW slots for a batch of n packets for BEB, LB, LLB [10] and STB [33, 41].

Our Metrics. For a single batch of n packets, algorithmic results address the number of slots required to complete all n packets. These slots correspond only to those belonging to contention windows, even though many results refer to this as makespan. To avoid confusion, we will refer to this metric more explicitly by ***contention-window slots (CW slots)***.

Table 2 summarizes the known with-high-probability[3] guarantees on CW slots. Note that LB, LLB, and STB each have superior guarantees over BEB, with STB achieving $\Theta(n)$ CW slots which is asymptotically optimal.

We also make use of a second metric. As described in Section 1.2, events occur outside of contention windows (such as SIFS, DIFS, full packet transmission, ACK timeouts). To denote the duration – including the time spent in contention windows – between when the single batch of packets arrives and when the last packet successfully transmits, we refer to ***total time***.[4]

---

[3]With probability at least $1 - 1/n^c$ for a tunable constant $c > 1$.
[4]We exclude the time required for a station to associate with the access point and ARP requests/replies. This would only (unfairly) strengthen the effects we observe.
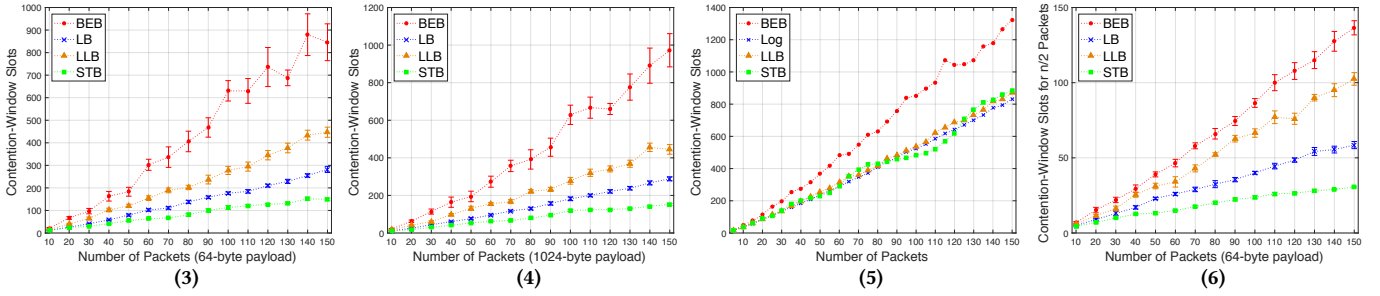
Figures 3-6. Median values are reported: (3) and (4) CW slots from NS3 experiments with 30 **trials for each value of** *n* **with** 64B **and** 1024B **payloads, respectively, (5) CW slots via Java simulation with** 50 **trials for each value of** *n*, **(6) number of CW slots required to finish** $n/2$ **packets from NS3 experiments with** 20 **trials for each value of** *n*. **Bars represent** 95% **confidence intervals.**

3.1 Theory and Experiment

We begin by comparing the number of CW slots. The algorithms we investigate are designed to reduce this quantity since all slots in the abstract model occur within some contention window. Under this metric, LLB, LB, and STB are expected to outperform BEB.

Throughout, when we report on performance, we are referring to median values for $n = 150$. Percentage increases or decreases are calculated as $100 \times (A - B)/B$ where B is always the value for BEB (the "old" algorithm) and A corresponds to a value for one of LLB, LB, or STB (the "new" algorithms).

3.1.1 Contention-Window Slots. We provide results from our NS3 experiments using both small packets, with a 64-byte (B) payload, and large packets, with a 1024B payload.

Figures 3 and 4 illustrate our experimental findings with respect to CW slots.[5] The behavior generally agrees with theoretical predictions that each of LLB, LB, and STB should outperform BEB.

Interestingly, LLB incurs a greater number of CW slots than LB despite despite the former's better asymptotic guarantees. We suspect this is an artifact of hidden constants/scaling and evidence of this is presented later in Section 5.1.

Nevertheless, LLB, LB, and STB demonstrate improvements over BEB, giving a respective decrease of 49.4%, 68.2%, and 83.0%, respectively, with a 64B payload. Similarly, LLB, LB, STB demonstrate a respective decrease of 54.2%, 69.9%, 84.2% with a 1024B payload.

For comparison, Figure 5 depicts CW slots derived from a simple Java simulation that implements only the assumptions of the abstract model (it ignores wireless effects, details in the protocol stack, etc.). Our NS3 results also roughly agree with this data in terms of magnitude of values and the separation of BEB from the other algorithms; albeit, the performances of LLB, LB, and STB do not separate cleanly in this data.

Finally, Figure 6 presents the number of CW slots (for 64B) required to complete half the packets, and we make two observations. First, the remaining $n/2$ packets are responsible for the bulk of the CW slots. Second, the improvement over BEB decreases to 25.0%, 56.4%, and 77.7% for LLB, LB, and STB, respectively (and similarly for 1024B). This difference is due to "straggling" packets which survive until relatively large windows are reached. This impacts BEB

---

[5]The following common approach is used to identify outliers in our data. Let Δ be the distance between the first and third quartiles. Any data point that falls outside a distance of 1.5Δ from the median is declared an outlier. We emphasize this results in very few points being discarded; for example, only a *single n* value for our 64B experiments had 5 outliers (out of 30 trials), and the vast a majority had none.

more than other algorithms given its rapidly increasing window size (and, unlike STB, it does not have a "backon" component).

> **Result 1.** *Experiment confirms theoretical predictions that LLB, LB, and STB outperform BEB with respect to CW slots.*

3.1.2 Total Time. It is tempting to consider the single-batch scenario settled. However, if we focus on the total time for both the 64B and 1024B payload sizes, then a different picture emerges.

The degree to which these newer algorithms outperform BEB is erased as seen from Figures 7 and 8. In fact, the order of performance is reversed with total time ordered from least to greatest as BEB, LLB, LB, STB. For 64B payloads, LLB, LB, and STB suffer an increase of 5.6%, 19.3%, and 26.5%, respectively, over BEB. For 1024B payloads, the increase is 9.1%, 25.4%, and 35.4%, respectively. Notably, the larger packet size seems to favor BEB.

What about the time until $n/2$ packets are successfully transmitted? Perhaps newer algorithms do better for the bulk of packets, but suffer from a few stragglers? Interestingly, Figures 9 and 10 suggest that this is not the case. Indeed, for a 64B payload, BEB performs even better over LLB, LB, and STB with the latter exhibiting an increase of 13.1%, 17.3%, 25.4%, respectively. Similarly, for 1024B, the percent increase is 10.1%, 16.6%, 26.6%, respectively.

> **Result 2.** *In comparison to BEB, the total time for each of LLB, LB, and STB is significantly worse.*
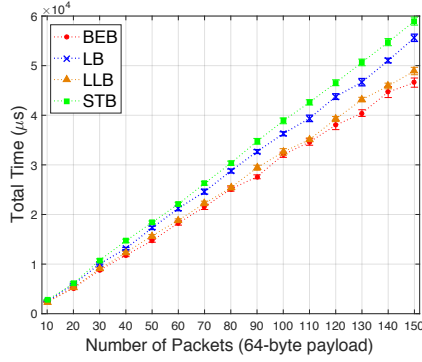
These findings are troubling since, arguably, total time is a more important performance metric in practice than CW slots. Critically, we note that this behavior is detected only through the use of NS3; it is not apparent from the simpler Java simulation. What is the cause of this phenomenon?
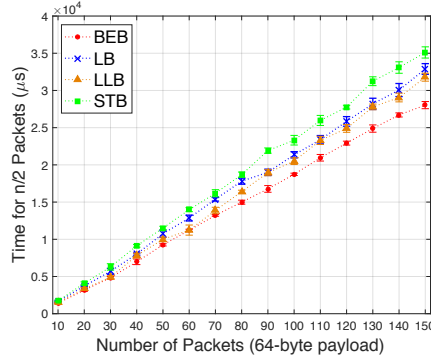
3.2 The Cost of Collisions

The number of ACK timeouts per station provides an important hint. As Figure 8 shows, the newer algorithms are incurring substantially more ACK timeouts which, in turn, corresponds to more (re)transmissions.

This evidence points to collisions as the main culprit. In particular, the way in which collision detection is performed means that each collision is costly in terms of time. In support of our claim, we decompose this delay into three portions using BEB (for $n = 150$) as an example throughout:
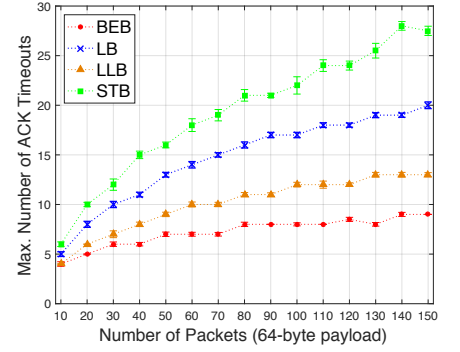
Figures 7-12. NS3 results with the median reported from 30 **trials for each value of** n**: (7) and (8) total time for** 64**B and** 1024**B payloads, (9) and (10) time required to complete** $n/2$ **packets with a** 64**B payload and** 1024**B payloads, (11) the maximum number of ACK timeouts per station over all stations with a** 64**B payload, (12) corresponding time spent waiting for ACK timeouts by the station with maximum ACK timeouts. Data plotted with** 95**% confidence intervals.**

(I) Transmission Time. (Re)transmissions are expensive. A packet of size 128B (64B payload plus 64B overhead) requires roughly $19\mu s$ plus the associated $20\mu s$ preamble. The maximum number of ACK timeouts for BEB – and, thus, the number of collisions – experienced by an unlucky station is 9.

Most collisions should involve only a handful of stations given the growth of CWs. (Why? Recall from Figure 6 that $n/2$ packets require the vast majority of CW slots to finish. These $n/2$ packets succeed only in larger windows – of size roughly equal to or greater than n for BEB – and do not finish immediately due to collisions, each of which should involve only a few stations given such window sizes). If two stations are involved in each collision, this results in $75(9/2)$ non-overlapping (or *disjoint*) collisions, for an aggregate duration of roughly $75(9/2)(19\mu s + 20\mu s) = 13,163\mu s$.[6]

(II) ACK Timeouts. Given a collision, the AP fails to obtain the transmission and the corresponding stations incur an ACK timeout before concluding that a collision occurred. This delay is significant – roughly $1,100\mu s$ for BEB with $n = 150$ (see Figure 12) – but an order-of-magnitude less than the transmission time.

(III) Contention-Window Slots. BEB incurs 886 CW slots for $n = 150$, each of duration $9\mu s$, spent in CWs which yields $7,974\mu s$.

For BEB at $n = 150$, these three values yield a very conservative lower bound on the total time of $22,237\mu s$; for instance, we have not accounted for the SIFS and DIFS. This back-of-the-envelope calculation conforms to the magnitude of values observed in Figure 7, and it highlights two important facts. First, both transmission time and CW slots contribute significantly to total time, with ACK timeouts being a distant third. Second, collisions greatly impact total time – far more than CW slots – by forcing retransmissions.

Underlining this second point, we note that the transmission time for the 1024B payload is larger at roughly $75(9/2)(161\mu s + 20\mu s) = 61,088\mu s$.[7] By comparison, the CW slots contribute roughly $973 \times 9\mu s = 8,757\mu s$.

ACK Timeout $\approx$ Collision. It is true that not all ACK timeouts necessarily imply that the corresponding packet suffered a collision. For example, an ACK might be lost due to wireless effects even if the packet was transmitted without any collision. Note that, in such a case, the sending station still diagnoses that a collision has occurred and so the same costs described in (I)-(III) hold.

For our simple NS3 setup, virtually all ACK failures result from a collision. This is evident from Figure 13 which illustrates a trial with $n = 20$ under BEB using a 64B payload. Collisions occur only when

---

[6]We do not add the time for the final/successful transmissions (this would only increase the value); our focus is on the transmissions associated with collisions.

[7]The number of ACK timeouts for 1024B is roughly the same, even though packet size has increased. This aligns with our findings in Section 4.
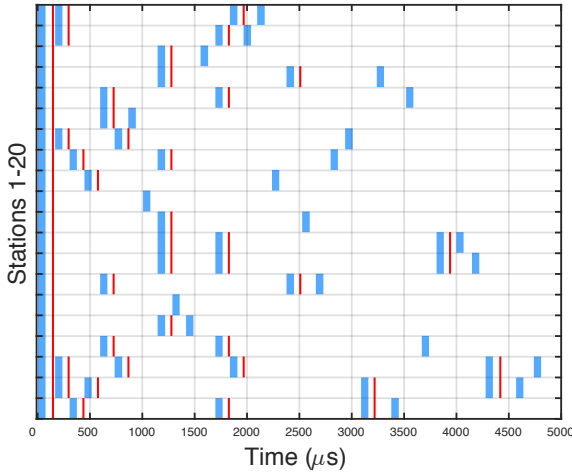
Figure 13. Execution of BEB with 20 stations.

two or more stations transmit (duration of transmission denoted by a thick blue line) at the same time and the result is an ACK timeout event (indicated by a thin red line); in all other cases, the transmission is successful and the corresponding ACK is received.

Disjoint Collisions. We observe that the total time does not grow linearly with the maximum number of ACK timeouts – equivalently, collisions – experienced by a station. Under LB, an unlucky station suffers roughly twice the number of ACK timeouts, but the total time of LB is not twice that of BEB.

Why? Consider n stations where each collision involves only two stations. Then, there are $n/2$ disjoint collisions and each is added to the total time. In contrast, consider the opposite extreme where all n stations transmit at the same time and collide. Then, there is one collision which adds only a single failed transmission time to the total time.

The number of stations involved in a single collision is larger for algorithms whose CWs grow more slowly, such as LB and LLB. For STB, a similar phenomenon is at work; the backon component yields collisions involving many stations. That is, LB, LLB, and STB are closer to the second case, while BEB is closer to the first.

From our experimental results, we see assumption A2 is not accurate with regards to the cost of failure:

> **Result 3.** *The impact from collision detection is not properly accounted for by A2. This impact is primarily a function of:*
> - *transmission time, and*
> - *time spent in contention windows,*
> *with the former dominating.*

RTS/CTS. Although it is not examined in detail in our work, we briefly remark on the use of RTS/CTS. When enabled, stations can experience collisions among the RTS frames (instead of the packets). These are smaller in size (20B), but the remainder of the total-time calculation remains the same, and additional time is incurred due to additional inter-frame spaces and the transmission of CTS frames.

For very large packets, we may expect RTS/CTS to mitigate the transmission-time cost, but not for small to medium-sized packets where the overhead from this mechanism might even cause worse performance. Ultimately, we observe the same qualitative behavior

when RTS/CTS is enabled. For example, without RTS/CTS, recall from Section 3.1.2 that the total time for LLB (BEB's closest competitor) increases by 5.6% and 9.1% for the 64B and 1024B, respectively, over BEB. With RTS/CTS, the increases are 10.7% and 7.5%.

3.2.1 Backing Off Slowly is Bad. The reason for the discrepancy between theory and experiment is now apparent. LLB increases each successive contention window by a smaller amount than BEB; in other words, LLB is *backing off more slowly*; the same is true of LB. Informally, this slower-backoff behavior is the reason behind the superior number of CW slots for LB and LLB since they linger in CWs where the contention is "just right" for a significant fraction of the packets to succeed. However, backing off slowly also inflicts a greater number of collisions.

Note that BEB backs off faster, jumping away from such favorable contention windows and thus incurring many empty slots. This is undesirable from the perspective of optimizing the number of CW slots. However, the result is fewer collisions. Given the empirical results, this appears to be a favorable tradeoff.

We explicitly note that LLB backs off faster than LB. In this way, LLB is "closer" to BEB and, therefore, is not outperformed as badly as illustrated in Figures 7 and 8.

> **Result 4.** *For algorithm design, optimizing CW slots at the expense of increased collisions is a poor design choice.*

Can we quantify the tradeoff between CW slots and collisions? From our discussion in Section 3.2, the total time for an algorithm A, denoted by $\mathcal{T}_A$, is approximated as:

$$\mathcal{T}_A = C_A \cdot (P + \rho) + \mathcal{W}_A \cdot s$$

where C_A is the number of disjoint collisions, P is the transmission time for a packet, ρ is the preamble duration, $\mathcal{W}_A$ is the corresponding number of CW slots, and s is the duration of a slot.

Abstracting further, we may treat ρ and s as constants to get:

$$\mathcal{T}_A = \Theta\left(C_A \cdot P + \mathcal{W}_A\right)$$

In other words, total time depends on the number of disjoint collisions (which depends on n) — each of which has a severity that depends on P — and the number of CW slots (which depends on n).

How does P behave? We assume it is proportional to packet size. For small values of n, it seems reasonable to consider $P = \Theta(1)$. However, if we are interested in the asymptotic behavior of $\mathcal{T}_A$, then P should not be treated as a constant. Arguably, as n scales, the number of bits required to address devices must also increase, and it is not uncommon to assume P scales as the logarithm of n.

Previous results have already established $\mathcal{W}_A$, so the parameter of interest is C_A, which we investigate next.

4 BOUNDS ON COLLISIONS

In order to provide additional support for our empirical findings, we derive asymptotic bounds on C_A. In comparison to BEB, we demonstrate that STB is asymptotically equal while both LLB and LB suffer from asymptotically more disjoint collisions.

Our arguments are couched in terms of packets and slots, but what follows is a balls-into-bins analysis. To bound C_A, we are interested in the number of bins (where bins make up the slots in a CW) that contain two or more balls; this is a disjoint collision (or just a *collision*). The second column of Table 3 presents our results.

4.1 Upper Bounding Collisions in BEB

Claim 1. *For a single batch of n packets, with high probability the number of collisions for BEB is $O(n)$.*

Proof. In the execution of BEB, consider a contention window of size $n2^i$ for an integer $i \geq 0$; let the windows be indexed by i. Note that up to window $i = 0$, we have $O(n)$ collisions since there are $O(n)$ slots by the sum of a geometric series.

Let the indicator random variable $X_j = 1$ if slot j in window i is a collision; $X_j = 0$ otherwise. We map this to a balls-and-bins problem, where a ball corresponds to a packet and a bin corresponds to a slot in a contention window.

$$Pr[X_j = 1] \quad = \quad O\left(\binom{n}{2}\left(\frac{1}{n2^i}\right)^2\left(1 - \frac{1}{n2^i}\right)^{n-2}\right) = O\left(\frac{1}{2^{2i}}\right)$$

Pessimistically, assume n balls are dropped in each consecutive window i; in actuality, packets finish over these windows and reduce the probability of collisions. Let $L_i = \sum_{j=1}^{n2^i} X_j$ be the number of collisions in window i. By linearity of expectation:

$$E[L_i] = \sum_{j=1}^{n2^i} E\left[X_j\right] = O\left(\frac{n}{2^i}\right)$$

Using the method of bounded differences [29], w.h.p. L_i is tightly bounded to its expectation. By [10], w.h.p. BEB finishes within $m = O(\lg \lg n)$ windows and so $C_{BEB} = \sum_{i=0}^{m} L_i = O(n/2^i) = O(n)$. □

4.2 Lower Bounding Collisions in LLB and LB

The specification of LLB analyzed here is slightly different from the description in Section 3; the contention window size doubles, but each such window of size w is repeated for $\lg \lg w$ iterations. With respect to CW slots and disjoint collisions, this is asymptotically equivalent to the strictly monotonic version described earlier [10].

The window size of interest is $\Theta(n/\lg \lg \lg n)$, since LLB finishes within a window of this magnitude [10]. We first prove an upper bound of $o(n)$ successes in a single execution of this window. This allows us to claim $\Omega(\lg \lg n)$ iterations exist where $\Theta(n)$ packets remain unfinished. Next, we prove that for each such iteration, $\Omega(n/\lg \lg \lg n)$ collisions occur yielding a total of $\Omega(n \lg \lg n/\lg \lg \lg n)$ collisions.

Lemma 4.1. *Assume ϵn packets and a CW of size $cn/\lg \lg \lg n$ for a sufficiently large constant $\epsilon \leq 1$ and sufficiently small constant $c > 0$. With high probability, at most $O(n/(\lg \lg n)^d)$ packets succeed in the CW for a constant $d > 1$ depending on ϵ and c.*

Proof. Let $Y_j = 1$ if a packet succeeds in slot j, otherwise $Y_j = 0$. We have:

$$Pr[Y_j = 1] \quad = \quad \binom{\epsilon n}{1}\left(\frac{\lg \lg \lg n}{cn}\right)\left(1 - \frac{\lg \lg \lg n}{cn}\right)^{\epsilon n-1}$$

$$\leq \quad \frac{\epsilon \lg \lg \lg n}{c(\lg \lg n)^{\epsilon \lg(e)/c}} \leq O\left(\frac{1}{c(\lg \lg n)^d}\right)$$

for a constant $d > 1$ where the last line follows from noting that $\epsilon \lg(e)/c > 1$ for a sufficiently large constant ϵ and a sufficiently small constant $c > 0$. Let $Y = \sum_j Y_j$, then:

$$E[Y] = O\left(\frac{n}{(\lg \lg n)^d}\right)$$

By the method of bounded differences, w.h.p. this is tight. □

Claim 2. *For a single batch of n packets, with high probability LogLog-Backoff experiences $\Omega\left(\frac{n \lg \lg n}{\lg \lg \lg n}\right)$ collisions.*

Proof. We focus on a contention window w of size $cn/\lg \lg \lg n$ for a sufficiently small constant $c > 0$. Conservatively, we do not count collisions prior to this window (counting these can only improve our result).

Prior to this window, w.h.p. $o(n)$ packets have succeeded. To see this, note that w.h.p. no packet finishes prior to a window of size $\Theta(n/\lg n)$. The number of intervening windows before reaching size $cn/\lg \lg \lg n$ is less than $\lg \lg n$. Pessimistically assume each intervening window has size $cn/\lg \lg \lg n$, then each results in $O(n/(\lg \lg n)^d)$ successful packets w.h.p. by Lemma 4.1 for $d > 1$. Each such intervening window executes $O(\lg \lg n)$ times. Therefore, the total number of packets finished is still $O(n/(\lg \lg n)^{d'})$ for a constant d' depending only d, and so $\Omega(n)$ packets remain.

In this window, assume that ϵn packets exist for some constant $\epsilon > 0$. Let $X_j = 1$ if slot j contains a collision; otherwise, $X_j = 0$. Then we have $Pr[X_j = 1]$:

$$= 1 - \sum_{k=0}^{1}\binom{\epsilon n}{k}\left(\frac{\lg \lg \lg n}{cn}\right)^k\left(1 - \frac{\lg \lg \lg n}{cn}\right)^{\epsilon n-k}$$

$$\geq \quad 1 - \left(1 - \frac{\lg \lg \lg n}{cn}\right)^{\epsilon n} - \frac{\epsilon \lg \lg \lg n}{c}\left(1 - \frac{\lg \lg \lg n}{cn}\right)^{\epsilon n-1}$$

$$\geq \quad 1 - O\left(\frac{\epsilon \lg \lg \lg n}{c\,(\lg \lg n)^{2\epsilon \lg(e)/c}}\right) \text{ by Taylor series}$$

$$= \quad \Omega(1)$$

Let $X = \sum_j X_j$. The expected number of collisions over the contention window w is:

$$E[X] \quad = \quad \sum_j E[X_j] = \Omega\left(\frac{cn}{\lg \lg \lg n}\right)$$

By the method of bounded differences, w.h.p. this is tight.

The window w is executed $\lg \lg(cn/\lg \lg \lg n) = \Omega(\lg \lg n)$ times. By Lemma 4.1, $O(n/(\log \log n)^d)$ packets are succeeding in each such execution for some constant $d > 1$ given c is sufficiently small. Thus, there will be at least ϵn packets remaining in each execution for some sufficiently small constant $\epsilon > 0$. By the above lower bound on the number of collisions, w.h.p. this results in $C_{LLB} = \Omega(cn \lg \lg n/\lg \lg \lg n)$ collisions. □

An argument similar to that used to support Claim 2 yields:

Claim 3. *For a single batch of n packets, with high probability Log-Backoff experiences $\Omega\left(\frac{n \lg n}{\lg \lg n}\right)$ collisions.*

4.3 Upper Bounding Collisions in STB

For a single batch of n packets, it is known that w.h.p. STB has $W_{STB} = O(n)$ and this is a trivial upper bound on the number of collisions. A straightforward argument allows us to derive a lower bound of $\Omega(n)$.

Claim 4. *For a single batch of n packets, with high probability Sawtooth-Backoff experiences $\Omega(n)$ collisions.*

Proof. Consider a window of size $n/8$. The total number of slots up to the end of this window (including all the backon windows) is less than $n/2$; therefore, more than $n/2$ packets have not finished by

Algorithm A	Num. of Collisions C_A	Total Time $\mathcal{T}_A$
BEB	$O(n)$	$O\left(n \cdot P + n \log n\right)$
LB	$\Theta\left(\dfrac{n \log n}{\log \log n}\right)$	$\Omega\left(\dfrac{n \log n}{\log \log n} \cdot P\right)$
LLB	$\Theta\left(\dfrac{n \log \log n}{\log \log \log n}\right)$	$\Omega\left(\dfrac{n \log \log n}{\log \log \log n} \cdot P\right)$
STB	$\Theta\left(n\right)$	$\Theta\left(n \cdot P\right)$

Table 3: Asymptotic bounds on collisions and total time.

this point. In the next window, which has size $n/4$, the probability of a collision is constant. Therefore, the expected number of collisions is $C_{STB} = \Omega(n)$ and this is tight by the method of bounded differences.   □

Although BEB and STB are asymptotically equal in the number of collisions suffered, we expect the hidden constant in the big-O notation for STB to be larger due to the backon component. We consider this question, amongst others involving asymptotic performance, later in Section 5.1.

4.4 Asymptotic Behavior of Total Time

Plugging in the results from Section 4, our formula for $\mathcal{T}_A = \Theta\left(C_A \cdot P + \mathcal{W}_A\right)$ yields the third column Table 3.

Recall that for small n, treating P as a constant is reasonable. However, for large values of n, one may argue that P ought to be treated as a slowly growing function of n, such as $\Omega(\log n)$. In this case, we note that both $\mathcal{T}_{LB}$ and $\mathcal{T}_{LLB}$ exceed $\mathcal{T}_{BEB}$ asymptotically. In fact, even a smaller bound $P = \omega(\lg n \lg \lg \lg n / \lg \lg n)$ is sufficient to yield this asymptotic behavior.

> **Result 5.** *Theoretical bounds on total time imply that LLB and LB should underperform both BEB and STB for sufficiently large n and P.*

This analysis offers support for our conjecture that the number of collisions is an important metric – perhaps more so than the number of CW slots – when it comes to the design of contention-resolution algorithms.

With respect to total time, recall that in Section 3.1.2 an increase in packet size was seen to favor BEB over LLB, the latter being the closest competitor to BEB in our experiments (although, STB is asymptotically superior and we address this issue in Section 5.1). This aligns with the above discussion. Furthermore, as empirical support for our claim, we use NS3 to examine the relative performance of these two algorithms as packet size increases in Figure 14.

As the packet size grows, LLB performs increasingly worse than BEB. We fit a linear regression model of LLB - BEB on the number of packets. This fitting model implies that when the payload size increases by 100B, the average increase in total time for LLB is roughly $700\mu s$ more than the increase experienced by BEB. The increase rate is statistically significant (p-value less than 0.001).

5 DISCUSSION

We have presented our evidence for why assumption A2 is flawed. In this section, we conclude our argument by considering a few unresolved observations, and discussing the legitimacy of our findings in the context of other protocols/networks.
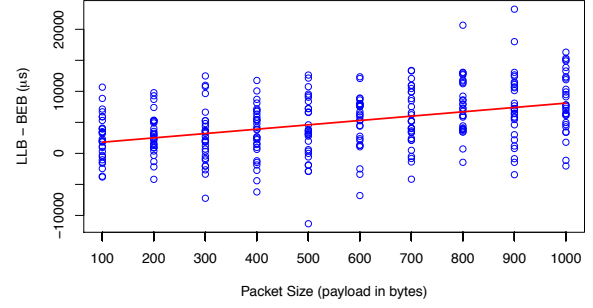

Figure 14. The difference in total time between LLB and BEB for $n = 150$ as packet size increases (30 trials per size).

5.1 Oddities at Small Scale

A few issues remain unaddressed:

   (i) In terms of asymptotic bounds on CW slots, the newer algorithms are ordered "best" to "worst" as STB, LLB, and LB. Yet, Figure 3 shows LB outperforming LLB.

  (ii) In terms of asymptotic bounds on the number of collisions, the newer algorithms are ordered "best" to "worst" as STB, LLB, and LB. Yet, Figure 11 shows STB suffering a larger number of ACK timeouts than both LLB and LB.

 (iii) BEB and STB have an asymptotically equal number of collisions, but STB is expected to suffer more and this is supported by Figures 7 and 8. What is the long-term behavior?

As we discussed previously in Section 2, NS3 is valuable in revealing flawed assumptions via the extraordinary level of detail it provides; however, this also prevents experimentation with NS3 at larger scales. We attempt to shed light on (i) - (iii) by examining larger values of n in order to see if our predictions are met, and we employ our simpler Java simulation for this task.

To address (i), we look at $n \leq 10^5$ as plotted in Figure 15. Now we see that STB performs best in terms of CW slots, and that LLB is indeed outperforming LB. This supports prior theoretical results for CW slots given sufficiently large n.

In regard to (ii), we again take $n \leq 10^5$ and plot the ratio of collisions: LB vs STB and LLB vs STB. Figure 16 demonstrates that the number of collisions for LB quickly exceeds STB. The tougher case is LLB which only begins to evidence a greater number of collisions at approximately $n = 30,000$. Nevertheless, we observe a trend towards exceeding parity, as expected. Moreover, the sluggish trajectory is not surprising given our analysis in Section 4.

Finally, for (iii), we observe that the number of collisions for STB is larger than BEB by roughly a factor of 2 over this large range of n. Note that the plot of BEB/STB is (roughly) flat, as expected from our asymptotic analysis of collisions.

5.2 Scope of Our Findings

In this section, we consider to what extent our findings are an artifact of IEEE 802.11g, and whether LB, LLB, STB might do better inside other protocols.

IEEE 802.11g uses a truncated BEB, is this significant? In our experiments, the maximum congestion-window size is 1024 which differs from the abstract model where no such upper bound exists. However, even for $n = 150$, this maximum is rarely reached during
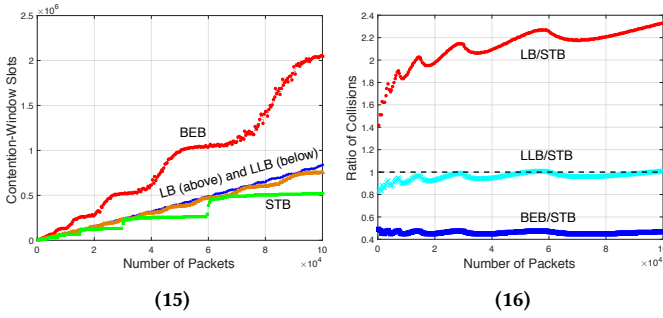
Figures 15-16. Results from Java simulation results with 200 **trials per** $n \leq 10^6$ **in increments of** 400**: (15) CW slots with median values plotted, (16) ratio of median number of collisions for BEB, LLB, and LB versus STB.**

an execution of BEB and this does not seem to have any noticeable impact on the trend observed in Figures 3 and 4.

What if smaller packets are used? During a collision, the time lost to transmitting would be reduced. In an extreme case, if the transmission of a packet fit within a slot, this would align more closely with A2.

Due to overhead, packet size has a lower bound in IEEE 802.11. Additionally, in NS3, there is a 12-byte payload minimum which translates into a minimum packet size of 76 bytes for our experiments.[8] The same qualitative behavior is observed in terms of CW slots and total time. For total time, the increase by LLB, LB, and STB is 6.6%, 17.8%, and 20.6%.

Alternatives to 802.11 might see more significant decreases. However, there is a tradeoff for *any* protocol. A smaller packet implies a reduced payload given the need for control information (for routing, error-detection, etc.) and this means that throughput is degraded.

What if the ACK-timeout duration is reduced or acknowledgements are removed altogether? This would also bring us closer to A2, although less so than having smaller packets – the delay from ACK timeouts does not dominate as discussed in Section 3.2. In our experiments, the ACK-timeout is $75\mu s$ (recall Section 2) and values below this threshold will lead a station to consider its packet lost before the ACK can be received. This results in unnecessary retransmissions and, ultimately, poor throughput.

Totally removing acknowledgements (or some form of feedback) is difficult in many settings since, arguably, they are critical to *any* protocol that provides reliability; more so when transmissions are subject to disruption by other stations over a shared channel.

To what extent do these findings generalize to other protocols? We do *not* claim that our findings hold for all protocols. If (a) sufficiently small packets are feasible *and* (b) reliability is not paramount, performance should align better with theoretical guarantees derived from using assumption A2.

We *do* claim that the performance of how collision detection is performed – and which is ignored under A2 – seems common to several other protocols. Examples include members of the IEEE 802.11 family, IEEE 802.15.4 (for low-rate wireless networks), and

---

[8]This is set within the `UdpClient` class of NS3.

IEEE 802.16 (WiMax). These employ some form of backoff and, regarding (a) and (b), each incurs header bloat and uses feedback via acknowledgements or a timeout to determine success or failure. This is a significant slice of current wireless standards that, given our findings, could *potentially* experience performance degradation if BEB is replaced by LB, LLB, and STB.

> **Result 6.** *Designing contention-resolution algorithms using assumption A2 seems likely to translate into poor performance in practice for a range of protocols.*

A setting where the abstract model may be valid is networks of multi-antenna devices. If a collision can be detected more efficiently, perhaps by a separate antenna, the delay due to transmission time can be reduced. Canceling the signal at the sending device so that other transmissions (that would cause the collision) can be detected is challenging. However, this is possible (for an interesting application, see [38]) and such schemes have been proposed using multiple-input multiple-output (MIMO) antenna technology [46, 54].

Finally, we note that future standards may satisfy (a) and (b). A possible setting is the Internet-of-Things (IoT); for example, [9] characterizes IoT transmissions as "small" and "intermittent, delay-sensitive, and short-lived". To reduce delay, the authors argue for removing much of the control messaging used by traditional MAC protocols. Therefore, this setting seems more closely aligned with A2. However, using this same logic, [9] also argues for the removal of any backoff-like contention-resolution mechanism. Nevertheless, these standards are in flux and we may see protocols that avoid the issues we identify here.

6 A SIZE-ESTIMATION APPROACH

Given our findings, we consider an alternative approach to the design of contention-resolution algorithms. Feedback is a useful ingredient as it allows stations to tune their sending probabilities. For windowed algorithms, this feedback is obtained via collisions which, as we have seen, is costly.

To avoid this problem, we examine a different approach. Stations first estimate n and then execute ***fixed backoff*** where the size of each contention window is set to this one-time estimate. So long as the algorithm avoids an underestimate, the large number of collisions incurred by BEB, LB, LLB and STB should be avoided.

Prior work in [13, 16–18, 40] examines size estimation as a means for improving performance, although the methods and traffic assumptions differ (see Section 7). Our goal is to experiment with an algorithm for a single batch of arrivals, whose specification lends itself to implementation, and whose improved performance manifests for practical values of n.

To this end, the size-estimation component of our algorithm, BEST-OF-k, specified in Figure 17 is a variant of a well-known "folklore" result (see [45]). For $k = \Theta(1)$, a significant overestimate may occur, but the amount by which it can underestimate is bounded; w.h.p. the estimate will be $\Omega(n/\log n)$.

Dummy packets of 28 bytes are used in the size-estimation phase; this small size is possible because the packets contain none of the upper-layer headers (these are not used in our IEEE 802.11 experiments since routing requires the upper-layer headers). Execution

Best-of-k
- For $i = 0$ to 10, do:
  - For each of k consecutive slots, do:
    - With probability $\frac{1}{2^i}$, send dummy packet; otherwise, sense the channel.
  - If channel is clear for more than $k/2$ slots, then:
    - $W \leftarrow 2^i$
    - Terminate and start executing fixed backoff using a contention window of size W

Figure 17. The Best-of-k contention-resolution algorithm.

proceeds in $35\mu s$ rounds during which a dummy packet is transmitted. Channel sensing is used to distinguish "busy" from "clear"; therefore, we avoid any collision detection and the use of any acknowledgements for these dummy packets.

As expected, for $k = 3$, the estimates are somewhat noisy, but this improves with $k = 5$; see Figure 18. Notably, increasing k does not significantly impact performance since the time required to run the size-estimation component is negligible (less than 5%) of the total time; instead, running fixed backoff is the main source of delay. We also observe that only overestimates occur, as predicted. This has the benefit of yielding good performance due to the lack of collisions. As demonstrated by Figure 19, both versions of the size-estimation approach outperform BEB, with $k = 3$ and $k = 5$ yielding a decrease in total time of 26.0% and 24.7% respectively.

> **Result 7.** *A size-estimation approach to designing contention-resolution algorithms appears promising.*

Finally, we note our assumption of synchronization might not hold in a dynamic setting. Furthermore, a real-world deployment would likely be "messier" with respect to interference from other devices/networks running different applications, impact of terrain and weather on transmissions, etc. and our simple setup does not account for such phenomena. Additional methods would be needed to address these issues. However, as an initial proof of concept, our results suggest an approach to contention resolution that *may* compete with BEB in the single-batch case.

7 RELATED WORK

Exponential backoff has been studied under Poisson-distributed traffic (see [36, 37, 44, 58]). Guarantees on stability are known [3, 4, 39], and under saturated conditions [15].

There is a vast body of literature addressing the performance of IEEE 802.11 (for examples, see [1, 26, 30, 51, 56, 71, 75]). There are several results that focus on the performance of BEB within IEEE 802.11; however, they do not address issues of the abstract model, bursty traffic, or the newer algorithms examined here. Nonetheless, we summarize those works that are most closely related.

Under continuous traffic, windowed backoff schemes are examined in [68] with a focus on the tradeoff between throughput and fairness. The authors focus on polynomial backoff and demonstrate via analysis and NS2 (the predecessor to NS3) simulations that quadratic backoff is a good candidate with respect to both metrics.
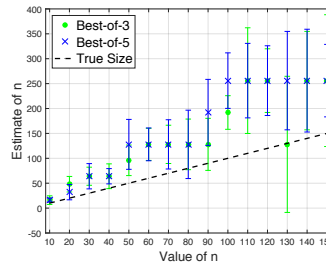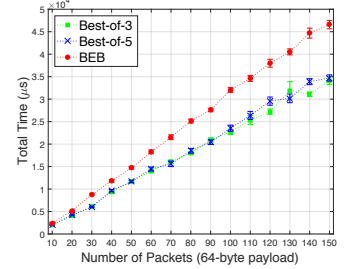
(18)   (19)

Figures 18-19. Median values are reported for NS3 results with 20 trials for each value of n: (18) estimation values, (19) total time (μs) for BEB, Best-of-3, and Best-of-5. Bars represent 95% confidence intervals.

Work in [49] addresses saturated throughput (each station always has a packet ready to be transmitted) of exponential backoff; roughly, this is the maximum throughput under stable packet arrival rates. Custom simulations are used to confirm these findings.

In [65], the authors propose backoff algorithms where the size of the contention window is modified by a small constant factor based on the number of successful transmissions observed. NS2 simulations are used to demonstrate improvements over BEB within 802.11 for a steady stream of packets (i.e. non-bursty traffic).

Lastly, in [52], the authors examine a variation on backoff where the contention window increases multiplicatively by the logarithm of the current window size (confusingly, also referred to as "logarithmic backoff"). NS2 simulations imply an advantage to their variant over BEB within IEEE 802.11, again for non-bursty traffic.

In regard to size-estimation approaches, there is prior work on tuning the probability of a transmission under Poisson-distributed traffic [34, 42, 47]. Subsequent work in [17, 18] offers (custom) simulation and analytical results on performance improvements assuming that the transmission interval for which a station backs off is sampled from the geometric distribution. In [16], the authors propose a method for estimating the number of contending stations under saturation conditions, and custom simulations demonstrate the accuracy of this approach. More recent work in [13] proposes a size-estimation scheme with small (asymptotic) sending and listening costs; however, no experimental results are provided and implementing this scheme may be challenging.

Regarding the time required for a single successful transmission, [73] demonstrates a lower-bound of $\Omega(\log \log n)$. In different communication models, other bounds are known [31, 32].

A class of tree-based algorithms for contention resolution is proposed in [20]. Work in [11] addresses the case of heterogeneous packet sizes. The case where packets can arrive dynamically is examined in [22, 23, 27].

Energy efficiency is important to multiple access in many low-power wireless networks [13, 21, 45]. When the communication channel is subject to adversarial disruption, several results address the challenge of multiple access [8, 12, 57, 59–63, 69]. Finally, deterministic broadcast protocols have also received significant attention [5, 24, 25].

8 CONCLUDING REMARKS

We have presented evidence that a model commonly used for designing contention-resolution algorithms is not adequately accounting for the cost of collisions. A number of interesting questions remain.

In terms of analytical work, we have argued for why collisions matter at small scale and asymptotically, but what is the optimal tradeoff between collisions and CW slots? Does this change when we consider multi-hop networks or long-lived bursty traffic? Assuming that this tradeoff is known, can we design algorithms that leverage this information?

Regarding future experimental work, it may be of interest to perform a similar evaluation on other protocols. For example, much of what is examined in this work seems to apply to contention resolution under IEEE 802.15.4, and we expect collisions to be similarly expensive. However, are there subtle differences in the protocol that allow IEEE 802.15.4 to avoid collisions? What about newer wireless standards? Understanding any such behavior may aid in the design of future contention-resolution algorithms.

REFERENCES

[1] 2014. How Penalty Leads to Improvement: A Measurement Study of Wireless Backoff in IEEE 802.11 Networks. *Computer Networks* 75, Part A (2014), 37 – 57.
[2] 2016. IEEE Standard for Information Technology–Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks – Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)* (2016), 1–3534.
[3] H. Al-Ammal, A. L. Goldberg, and P. MacKenzie. 2001. An Improved Stability Bound for Binary Exponential Backoff. *Theory of Computing Systems* 34, 3 (2001), 229–244.
[4] Hesham Al-Ammal, Leslie Ann Goldberg, and Phil MacKenzie. 2000. *Binary Exponential Backoff Is Stable for High Arrival Rates.* Springer Berlin Heidelberg, Berlin, Heidelberg, 169–180.
[5] Lakshmi Anantharamu, Bogdan S. Chlebus, and Mariusz A. Rokicki. 2009. Adversarial Multiple Access Channel with Individual Injection Rates. In *Proceedings of the 13th International Conference on Principles of Distributed Systems (OPODIS).* 174–188.
[6] Antonio Fernández Anta and Miguel A. Mosteiro. 2010. *Contention Resolution in Multiple-Access Channels: k-Selection in Radio Networks.* 378–388.
[7] Chen Avin, Yuval Emek, Erez Kantor, Zvi Lotker, David Peleg, and Liam Roditty. 2009. SINR Diagrams: Towards Algorithmically Usable SINR Models of Wireless Networks. In *Proceedings of the 28th ACM Symposium on Principles of Distributed Computing (PODC).*
[8] Baruch Awerbuch, Andrea Richa, and Christian Scheideler. 2008. A Jamming-Resistant MAC Protocol for Single-Hop Wireless Networks. In *Proceedings of the 27th ACM Symposium on Principles of Distributed Computing (PODC).* 45–54.
[9] A. Bakshi, L. Chen, K. Srinivasan, C. E. Koksal, and A. Eryilmaz. 2016. EMIT: An Efficient MAC Paradigm for the Internet of Things. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications.* 1–9.
[10] Michael A. Bender, Martin Farach-Colton, Simai He, Bradley C. Kuszmaul, and Charles E. Leiserson. 2005. Adversarial Contention Resolution for Simple Channels. In *Proc. 17th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA).* 325–332.
[11] Michael A. Bender, Jeremy T. Fineman, and Seth Gilbert. 2006. Contention Resolution with Heterogeneous Job Sizes. In *Proceedings of the 14th Conference on Annual European Symposium (ESA).* 112–123.
[12] Michael A. Bender, Jeremy T. Fineman, Seth Gilbert, and Maxwell Young. 2016. How to Scale Exponential Backoff: Constant Throughput, Polylog Access Attempts, and Robustness. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '16).* 636–654.
[13] Michael A. Bender, Tsvi Kopelowitz, Seth Pettie, and Maxwell Young. 2016. Contention Resolution with Log-logstar Channel Accesses. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2016).* 499–508.
[14] B. N. Bhandari, R. V. Raja Kumar, and S. L. Maskara. 2008. Performance of IEEE 802.16 MAC Layer Protocol Under Conditions of Self-Similar Traffic. In *TENCON 2008 - 2008 IEEE Region 10 Conference.* 1–4.
[15] Giuseppe Bianchi. 2006. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *IEEE Journal on Selected Areas in Communications* 18, 3 (Sept. 2006), 535–547.
[16] Giuseppe Bianchi and Ilenia Tinnirello. 2003. Kalman Filter Estimation of the Number of Competing Terminals in an IEEE 802.11 Network. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM),* Vol. 2. 844–852.
[17] Frederico Cali, Marco Conti, and Enrico Gregori. 2000. Dynamic Tuning of the IEEE 802.11 Protocol to Achieve a Theoretical Throughput Limit. *IEEE/ACM Transactions on Networking* 8, 6 (2000), 785–799.
[18] Frederico Cali, Marco Conti, and Enrico Gregori. 2000. IEEE 802.11 Protocol: Design and Performance Evaluation of an Adaptive Backoff Mechanism. *IEEE Journal on Selected Areas in Communications* 18, 9 (2000), 1774–1786.
[19] B. Canberk and S. Oktug. 2009. Self Similarity Analysis and Modeling of VoIP Traffic under Wireless Heterogeneous Network Environment. In *Telecommunications, 2009. AICT '09. Fifth Advanced International Conference on.* 76–82.
[20] J. Capetanakis. 2006. Tree Algorithms for Packet Broadcast Channels. *IEEE Trans. Inf. Theor.* 25, 5 (Sept. 2006), 505–515.
[21] Yi-Jun Chang, Tsvi Kopelowitz, Seth Pettie, Ruosong Wang, and Wei Zhan. 2017. Exponential Separations in the Energy Complexity of Leader Election . In *Proceedings of the Annual ACM SIGACT Symposium on Theory of Computing (STOC).*
[22] Bogdan S. Chlebus, Leszek Gasieniec, Dariusz R. Kowalski, and Tomasz Radzik. 2005. On the Wake-up Problem in Radio Networks. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP).* 347–359.
[23] Bogdan S. Chlebus and Dariusz R. Kowalski. 2004. A Better Wake-up in Radio Networks. In *Proceedings of 23rd ACM Symposium on Principles of Distributed Computing (PODC).* 266–274.
[24] Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. 2006. Adversarial queuing on the multiple-access channel. In *Proc. Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing (PODC.* 92–101.
[25] Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. 2012. Adversarial Queuing on the Multiple Access Channel. *ACM Transactions on Algorithms* 8, 1 (2012), 5.
[26] Nakjung Choi, Yongho Seok, Yanghee Choi, Sungmann Kim, and Hanwook Jung. 2005. P-DCF: Enhanced Backoff Scheme for the IEEE 802.11 DCF. In *2005 IEEE 61st Vehicular Technology Conference,* Vol. 3.
[27] Marek Chrobak, Leszek Gasieniec, and Dariusz R. Kowalski. 2007. The Wake-Up Problem in Multihop Radio Networks. *SIAM J. Comput.* 36, 5 (2007), 1453–1471.
[28] NS-3 Consortium. 2017. NS-3. (2017). www.nsnam.org.
[29] Devdatt Dubhashi and Alessandro Panconesi. 2009. *Concentration of Measure for the Analysis of Randomized Algorithms* (1st ed.). Cambridge University Press.
[30] Andrzej Duda. 2008. Understanding the Performance of 802.11 Networks. In *2008 IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications.* 1–6.
[31] Jeremy T. Fineman, Seth Gilbert, Fabian Kuhn, and Calvin Newport. 2016. Contention Resolution on a Fading Channel. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing (PODC '16).* 155–164.
[32] Jeremy T. Fineman, Calvin Newport, and Tonghe Wang. 2016. Contention Resolution on Multiple Channels with Collision Detection. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing (PODC '16).* 175–184.
[33] Mihály Geréb-Graus and Thanasis Tsantilas. 1992. Efficient Optical Communication in Parallel Computers. In *Proceedings 4th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA).* 41–48.
[34] Mario Gerla and Leonard Kleinrock. 1977. Closed Loop Stability Controls for S-aloha Satellite Communications. In *Proceedings of the Fifth Symposium on Data Communications (SIGCOMM '77).* 2.10–2.19.
[35] Sayeed Ghani. 2010. The Impact of Self Similar Traffic on Wireless LAN. In *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference (IWCMC '10).* 52–56.
[36] L.A. Goldberg and P.D. MacKenzie. 1996. *Analysis of Practical Backoff Protocols for Contention Resolution with Multiple Servers.* ALCOM-IT Technical Report TR-074-96. Warwick. http://www.dcs.warwick.ac.uk/~leslie/alcompapers/contention.ps.
[37] Leslie Ann Goldberg, Philip D. MacKenzie, Mike Paterson, and Aravind Srinivasan. 2000. Contention Resolution with Constant Expected Delay. 47, 6 (Nov. 2000), 1048–1096.
[38] Shyamnath Gollakota, Haitham Hassanieh, Benjamin Ransford, Dina Katabi, and Kevin Fu. 2011. They Can Hear Your Heartbeats: Non-invasive Security for Implantable Medical Devices. In *Proceedings of the ACM SIGCOMM 2011 Conference (SIGCOMM '11).* 2–13.
[39] Jonathan Goodman, Albert G. Greenberg, Neal Madras, and Peter March. 1988. Stability of Binary Exponential Backoff. *J. ACM* 35, 3 (June 1988), 579–602.
[40] Albert G. Greenberg, Philippe Flajolet, and Richard E. Ladner. 1987. Estimating the Multiplicities of Conflicts to Speed Their Resolution in Multiple Access Channels. *JACM* 34, 2 (April 1987), 289–325.
[41] Ronald I. Greenberg and Charles E. Leiserson. 1985. Randomized Routing on Fat-Trees. In *Proc. of the Symp. on Foundations of Computer Science (FOCS).* 241–249.

[42] Bruce Hajek and Timothy van Loon. 1982. Decentralized Dynamic Control of a Multiaccess Broadcast Channel. *IEEE Trans. Automat. Control* 27, 3 (1982), 559–569.

[43] Magnús M. Halldórsson and Roger Wattenhofer. 2009. *Wireless Communication Is in APX*. Springer Berlin Heidelberg, Berlin, Heidelberg, 525–536.

[44] Johan Hastad, Tom Leighton, and Brian Rogoff. 1987. Analysis of Backoff Protocols for Multiple Access Channels. In *STOC'87*. New York, New York, 241–253.

[45] Tomasz Jurdziński, Mirosław Kutyłowski, and Jan Zatopiański. 2002. Energy-Efficient Size Approximation of Radio Networks with No Collision Detection. In *Proceedings of the 8th Annual International Conference (COCOON)*. 279–289.

[46] M. Kawahara, K. Nishimori, T. Hiraguri, and H. Makino. 2014. A New Propagation Model for Collision Detection using MIMO Transmission in Wireless LAN Systems. In *2014 IEEE International Workshop on Electromagnetics (iWEM)*. 34–35.

[47] Frank P. Kelly. 1985. Stochastic Models of Computer Communication Systems. *Journal of the Royal Statistical Society, Series B (Methodological)* 47, 3 (1985), 379–395.

[48] J. Komlos and A. Greenberg. 2006. An Asymptotically Fast Nonadaptive Algorithm for Conflict Resolution in Multiple-access Channels. *IEEE Trans. Inf. Theor.* 31, 2 (Sept. 2006), 302–306.

[49] Byung-Jae Kwak, Nah-Oak Song, and L. E. Miller. 2005. Performance Analysis of Exponential Backoff. *IEEE/ACM Transactions on Networking* 13, 2 (2005), 343–355.

[50] Mathieu Lacage and Thomas R. Henderson. 2006. Yet Another Network Simulator. In *Proceeding from the 2006 Workshop on NS-2: The IP Network Simulator (WNS2 '06)*.

[51] Jiandong Li, Zygmunt J. Haas, Min Sheng, and Yanhui Chen. 2003. Performance Evaluation of Modified IEEE 802.11 MAC for Multi-Channel Multi-Hop Ad Hoc Networks. *Journal of Interconnection Networks* 4, 3 (2003), 345–359.

[52] Saher S. Manaseer, Mohamed Ould-Khaoua, and Lewis M. Mackenzie. 2009. On the Logarithmic Backoff Algorithm for MAC Protocol in MANETs. In *Integrated Approaches in Information Technology and Web Engineering: Advancing Organizational Knowledge Sharing*. IGI Global, Chapter 12, 174–184.

[53] Robert M. Metcalfe and David R. Boggs. 1976. Ethernet: Distributed Packet Switching for Local Computer Networks. *CACM* 19, 7 (July 1976), 395–404.

[54] Y. Morino, T. Hiraguri, T. Ogawa, H. Yoshino, and K. Nishimori. 2014. Analysis Evaluation of Collision Detection Scheme Utilizing MIMO Transmission. In *2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 28–32.

[55] T. Moscibroda. 2007. The Worst-Case Capacity of Wireless Sensor Networks. In *2007 6th International Symposium on Information Processing in Sensor Networks*. 1–10.

[56] Qiang Ni, Lamia Romdhani, and Thierry Turletti. 2004. A Survey of QoS Enhancements for IEEE 802.11 Wireless LAN: Research Articles. *Wireless Communications and Mobile Computing* 4, 5 (Aug. 2004), 547–566.

[57] Adrian Ogierman, Andrea Richa, Christian Scheideler, Stefan Schmid, and Jin Zhang. 2014. Competitive MAC under adversarial SINR. In *IEEE Conference on Computer Communications (INFOCOM)*. 2751–2759.

[58] Prabhakar Raghavan and Eli Upfal. 1995. Stochastic Contention Resolution With Short Delays. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on the Theory of Computing (STOC)*. 229–237.

[59] Andrea Richa, Christian Scheideler, Stefan Schmid, and Jin Zhang. 2010. A Jamming-Resistant MAC Protocol for Multi-Hop Wireless Networks. In *Proceedings of the International Symposium on Distributed Computing (DISC)*. 179–193.

[60] Andrea Richa, Christian Scheideler, Stefan Schmid, and Jin Zhang. 2011. Competitive and Fair Medium Access Despite Reactive Jamming. In *Proceedings of the 31st International Conference on Distributed Computing Systems (ICDCS)*. 507–516.

[61] Andrea Richa, Christian Scheideler, Stefan Schmid, and Jin Zhang. 2012. Competitive and Fair Throughput for Co-Existing Networks Under Adversarial Interference. In *Proceedings of the 31st ACM Symposium on Principles of Distributed Computing (PODC)*.

[62] Andrea Richa, Christian Scheideler, Stefan Schmid, and Jin Zhang. 2013. An Efficient and Fair MAC Protocol Robust to Reactive Interference. *IEEE/ACM Transactions on Networking* 21, 1 (2013), 760–771.

[63] Andrea Richa, Christian Scheideler, Stefan Schmid, and Jin Zhang. 2013. Competitive Throughput in Multi-Hop Wireless Networks Despite Adaptive Jamming. *Distributed Computing* 26, 3 (2013), 159–171.

[64] N. I. Sarkar and K. W. Sowerby. 2009. The Effect of Traffic Distribution and Transport Protocol on WLAN Performance. In *Telecommunication Networks and Applications Conference (ATNAC), 2009 Australasian*. 1–6.

[65] M. Shurman, B. Al-Shua'b, M. Alsaedeen, M. F. Al-Mistarihi, and K. A. Darabkh. 2014. N-BEB: New Backoff Algorithm for IEEE 802.11 MAC Protocol. In *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. 540–544.

[66] Nah-Oak Song, Byung-Jae Kwak, and Leonard E. Miller. 2003. On the Stability of Exponential Backoff. *Journal of Research of the National Institute of Standards and Technology* 108, 4 (2003).

[67] M. Stoffers and G. Riley. 2012. Comparing the NS-3 Propagation Models. In *Proceedings of the IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*. 61–67.

[68] X. Sun and L. Dai. 2015. Backoff Design for IEEE 802.11 DCF Networks: Fundamental Tradeoff and Design Criterion. *IEEE/ACM Transactions on Networking* 23, 1 (2015), 300–316.

[69] Henry Tan, Chris Wacek, Calvin Newport, and Micah Sherr. 2014. *A Disruption-Resistant MAC Layer for Multichannel Wireless Networks*. 202–216.

[70] Shahram Teymori and Weihua Zhuang. 2005. *Queue Analysis for Wireless Packet Data Traffic*. Springer Berlin Heidelberg, Berlin, Heidelberg, 217–227.

[71] I. Tinnirello, G. Bianchi, and Y. Xiao. 2010. Refinements on IEEE 802.11 Distributed Coordination Function Modeling Approaches. *IEEE Transactions on Vehicular Technology* 59, 3 (2010), 1055–1067.

[72] Elias Weingärtner, Hendrik Vom Lehn, and Klaus Wehrle. 2009. A Performance Comparison of Recent Network Simulators. In *Proceedings of the 2009 IEEE International Conference on Communications (ICC'09)*. 1287–1291.

[73] Dan E. Willard. 1986. Log-logarithmic Selection Resolution Protocols in a Multiple Access Channel. *SIAM J. Comput.* 15, 2 (May 1986), 468–477.

[74] J. Yu and A. P. Petropulu. 2006. Study of the Effect of the Wireless Gateway on Incoming Self-Similar Traffic. *IEEE Transactions on Signal Processing* 54, 10 (2006), 3741–3758.

[75] Y. H. Zhu, X. Z. Tian, and J. Zheng. 2011. Performance Analysis of the Binary Exponential Backoff Algorithm for IEEE 802.11 Based Mobile Ad Hoc Networks. In *2011 IEEE International Conference on Communications (ICC)*. 1–6.