Backing out of Linear Backoff in Wireless Networks

Mahanth Gowda⋆ Nirupam Roy⋆ Romit Roy Choudhury⋆ Srihari Nelakuditi†
⋆University of Illinois (UIUC) †University of South Carolina
{gowda2,nroy8,croy}@illinois.edu srihari@cse.sc.edu

ABSTRACT

This paper revisits the randomized backoff problem in CSMA networks and identifies opportunities of improvement. The key observation is that today's backoff operation, such as in WiFi, attempts to create a total ordering among all nodes contending for the channel. Total ordering indeed assigns a unique backoff to each node (thus avoiding collisions), but pays the penalty of choosing the random back-offs from a large range, ultimately translating to channel wastage. We envision breaking away from total ordering. Briefly, we force nodes to pick random numbers from a smaller range, so that groups of nodes pick the same random number (i.e., partial order). Now, the group that picks the smallest number – the winners – is advanced to a second round, where they again perform the same operation. We show that narrowing down the contenders through multiple rounds improves channel utilization. The intuition is that time for *partially ordering all nodes* plus *totally ordering each small group* is actually less than the time needed to *totally order all nodes.*

We instantiate the idea with two well known CSMA protocols - WiFi and oCSMA. We resolve new challenges regarding multi domain contentions and group signaling. USRP and simulation based microbenchmarks are promising. We believe the idea of "hierarchical backoff" applies to other CSMA systems as well, exploration of which is left to future work.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication

Keywords

Wireless; Backoff; Correlation; Contention Resolution

1. INTRODUCTION

Randomized backoff has been a well-established approach to distributed resource sharing. First proposed in 1973 in the ALOHA protocol, the core idea has come to underpin a variety of standards, including Ethernet, WiFi, Zigbee. The basic idea with randomized backoff is simple and elegant. In the context of WiFi, nodes contending for the shared channel pick random numbers from a pre-specified range and count-down at the same pace. The node that reaches zero first transmits its packet, while others freeze their counters. When the transmission is complete, i.e., the channel is idle again, the other nodes resume the count-down. If two nodes reach zero at the same time, they transmit simultaneously and collide – the colliding nodes pick new backoffs from an exponentially larger range, and perform count-down again. Upon success, the successful node resets its random-number range to the (original) minimum value.

Randomized backoff may be abstracted as a way of packing nodes in a range of numbers, under the constraints that the packing is tight but no two nodes share the same number. Loosely, these constraints are well satisfied when the range of numbers is super-linear in the number of nodes, $O(n^2)$. Our observation is that the constraint of "no two nodes sharing the same number" may be too strong, which in turn inflates the number range. For collision-free channel access, we could ensure that only nodes close to accessing the channel have unique backoffs; other nodes can still share the same number.

As a first step, our hierarchical backoff scheme packs all n nodes in a much smaller range, so that groups of nodes share the same number – a partial order. Let us say k groups are formed. Now, only the group that lies at the head of the (partial) order can be re-packed under the constraint that they do not share the same number. Since each group is likely to be smaller than the total number of nodes, the range necessary to re-pack them (in total order) is $O((\frac{n}{k})^2)$, appreciably smaller than $O(n^2)$ (We verified the complexity via simulations). Our proposition builds on this intuition that the range needed for partial ordering all nodes in k groups, plus the ranges needed to total order each of the k groups, is still less than the range needed to total order all n nodes. In other words, we can show that $k^2 + k(n/k)^2 < n^2$, for suitably designed k. While this is the key to performance gain, building a system that realizes this gain presents design challenges. We briefly introduce them next.

Consider the example where multiple clients have backlogged traffic for an AP. Unlike WiFi, where nodes pick a random number from a range of $[0, CW]$, let us say we contend in two rounds and use a range $[0, \sqrt{CW}]$ for each round. Fig 1 shows an example with $CW = 16$. Naturally, this causes multiple nodes to pick the same number. Now, as each node counts down, when the group of nodes with the smallest number reaches zero (C_2 and C_3), they transmit a "busy signal" similar to a short preamble. We use our correlation based technique (Section 5) here for group signaling by the group of first round winners. The winners of this round – far fewer than the total number of nodes – perform a second round of contention by picking random numbers again from $[0, \sqrt{CW}]$. The node with the smallest backoff in the second round (R2) transmits its packet first, followed by subsequent nodes in R2. Once completed, the other nodes in R2 count down and transmit their own packets; Of course, the losers of the first round remain frozen, enforced through busy signals from the pending transmitters of R2 (detailed later). Once all R2 nodes have completed their transmissions, the first round

HotWireless'14, September 11 2014, Maui, HI, USA

http://dx.doi.org/10.1145/2643614.2643622

nodes resume count-down, and ultimately advance to R2. The R2 nodes that have just completed transmission now re-join first round contention. Where WiFi consumes ~CW slots in the example in Fig 1, HiBo finishes in $\sim 3\sqrt{CW}$ slots. Alternatively, splitting a single round contention window of CW into two rounds of $CW/2$ each will decrease collision probability (). We take advantage from both backoff compression and collision avoidance.

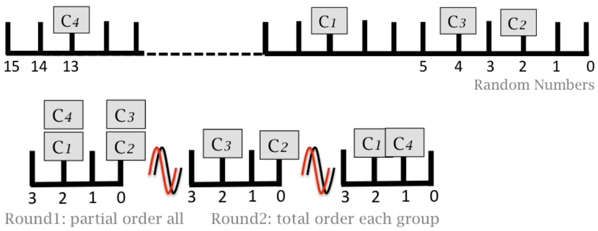

Figure 1: Key observation behind hierarchical backoff: Top WiFi time-line shows an attempt to total-order all nodes, while we attempt to partial-order nodes (into groups) and then total order each smaller group.

Of course, this is only a sketch of the protocol and several component challenges must be addressed. (1) With groups of nodes signaling to each other, detection techniques need to be suitably designed such that concurrent signals can be detected reliably and quickly. Relying on energy based detection is inadequate since multiple weak busy signals could add up and appear strong, forcing nodes to believe that the channel is busy. (2) It is possible that nodes in R2 overhear an ongoing transmission, and are hence silenced, causing "head of the line blocking" to nodes in R1. Such blocking cases need to be handled to attain spatial reuse. (3) New nodes that join the network need to learn the state of the system and begin count-down when R1 nodes are counting down. (4) Finally, colliding nodes should be able to adapt their backoff to cope with the time-varying contention in the network.

We show instantiations of the intuition on two protocols - WiFi and oCSMA - through systems called HiBo and o2CSMA respectively. HiBo attempts to optimize the backoff and decrease collisions in general WiFi to make it scalable at higher node densities. Utility optimal CSMA (oCSMA) - [9] is a distributed stochastic approximation algorithm which optimizes throughput and fairness based on utility functions. While optimality is proven by ignoring collisions, attempts to make it practical [14] suffer from poor scalability (Section 6) or require packet aggregation [11]. *o2CSMA* scales well without aggregation. A surge in mobile and wireless data traffic is expected and hence we believe optimizing backoff and scalability are important problems.

This paper designs a prototype to handle the challenges systematically. A USRP testbed is used to verify the PHY layer techniques, while simulations on NS3 are conducted to evaluate it on larger networks. Performance results show up to 30% increase in throughput at network densities of around 15 nodes, and up to 40% when density exceeds 30 nodes. Fairness also improves considerably across all network densities.

2. RELATED WORK

We only discuss relevant ideas from the vast backoff literature [7].

Hierarchical Backoff: Plenty of works exist on hierarchical contention and tree-based collision resolution algorithms [2, 3]. However, we find that the tree-splitting algorithms here operate at the granularity of packets (slot length is same as packet length), i.e., whenever packet collisions (RTS collisions in [3]) happen, about half of the colliding nodes defer and the other half make a second attempt and so on till eventual contention resolution. An entire packet duration is wasted during such resolutions. Although our idea is similar in nature, we resolve contentions at much finer granularities of WiFi slot lengths (9us), thereby paying negligible overhead at individual steps of the tree based contention resolution scheme. This scheme entails new challenges associated with busy signal decoding and multi-contention domains, which is the main focus of this paper. Closest to our work is [19] which uses multiple backoff stages, nodes in each stage waiting for higher-stage nodes to complete. However, the scheme is not designed for multiple collision domains, and ignores the possibility that new nodes may join the system, external interferences may silence some nodes, and over-exposed terminals may occur. Another paper explores hierarchy in the spatial domain [15], where spatially clustered nodes choose a leader who backoff on their behalf. The scheme is again designed for single collision domains and relies on heavy control traffic, making it impractical for real networks.

Adaptation to Contention: The basic backoff proposal has been optimized for various network parameters. Authors in [13,16] optimize the count-down for collisions, while [4] regulates access probability. WiFi's behavior has been modeled in [1] to offer insights into design choices. However, none of these systems disrupt the core backoff framework in an architectural sense.

Changes to Backoff Architecture: Our inspiration towards developing new backoff mechanisms arose from FICA [18], Back2F [17], and WiFi-Nano [12] and many others. In FICA and Back2F, authors showed a creative use of OFDM sub-carriers to enable control information. While FICA and Back2F warrant almost clean-slate designs, we wondered whether comparable gains can be achieved with lesser modifications. WiFi-Nano [12] was also inspiring in their use of correlation enabled primitives of "group coordination" that influenced our thinking process. Ideas in [8] were also compelling in characterizing WiFi's problems with scalability. The optimization of contention windows like [6] (IdleSense) can be applied to HiBo to identify optimal contention windows of round-1 and round-2, and it is complimentary to HiBo. We believe HiBo, o2CSMA are ideas that would have not been conceived in the absence of these existing works, however, we argue that these systems are completely different from them. The ability to separate groups of nodes in time blocks via concurrent signaling and detection is the key departure.

3. HiBo

We first describe a basic version of HiBo for a single collision domain with no new nodes joining, we relax the assumptions later. For every packet a node intends to transmit, it joins the first round of contention, denoted R1. In R1, each node i picks a random counter c_i^1 in the range [0, $CW1$], where $CW1$ denotes the length of R1's contention window. Then, when node i observes the channel to be idle for a slot duration, it decrements the counter, c_i^1. When this counter reaches 0, node i transmits a `busy` signal to announce the start of second round, denoted R2. It is possible that another node, say j, also counts down c_j^1 to 0 at the same time, transmits a `busy` signal, and enters R2 along with i. Upon detecting the `busy` signal, all other nodes with non-zero first-round counters (say k and l), freeze their countdown. The nodes k and l are expected to resume the countdown only after nodes i and j have contended in R2 and completed their transmissions. We will shortly discuss how this can be ensured, and later describe a technique to reliably detect the `busy` signal, even when multiple nodes (i and j) are transmitting it concurrently.

Now, upon advancing to R2, node i again picks another random counter c_i^2 (similarly, node j picks c_j^2) in the range [0, $CW2$] and begins countdown. Suppose c_i^2 is less than c_j^2, then assuming an isolated WLAN with no other parallel transmissions nearby, c_i^2 should reach zero before c_j^2. Thus, node i initiates data transmission whereas node j freezes its R2 counter c_j^2. Once i's transmission is complete, j resumes counting down and transmits when it reaches zero. Observe that while j is counting down – that is, when the channel is indeed idle – we still need the R1 losers k and l to remain frozen, so that they do not advance into the second round. To achieve this, we require that a node in R2 transmit a `busy` signal whenever it resumes its own R2 countdown. Thus, the order of operation is as follows: i's data transmission and ACK $\Rightarrow$ channel becomes idle $\Rightarrow$ j transmits `busy` signal $\Rightarrow$ j resumes countdown. First round losers detect this busy signal again, infer that transmissions are still pending in R2, and hence, remain frozen. Once i and j have completed transmission, k and l do not hear the `busy` signal anymore and resume countdown, ultimately advancing to R2. Nodes i and j go back to contend in R1, and the process repeats.

Fig 2 illustrates the state transition diagram from the point of view of node i. It starts in a *R1-Watch* state and transitions to the *R1-CountDown* state if it senses an idle channel for a certain duration (say IFS_1 – IFS denotes inter frame spacing in 802.11).It then keeps counting down as long as the channel is idle. If it receives a `busy` signal, indicating R2 is in progress, then it freezes the counter and transitions to the *R1-Defer* state. In that state, it waits for a DATA transmission to begin, and then moves to the *R1-Watch* state. If a `busy` signal is heard in the *R2-Watch* state, indicating R2 is still in progress, it goes back to the *R2-Defer* state. Otherwise, it waits for the channel to idle for a IFS_1 duration, then switches to the *R1-CountDown* state, and resumes countdown. When c_i^1 counts down to zero, it transmits a `busy` signal, picks a random R2 counter c_i^2, and enters *R2-CountDown* state. In this state, it keeps decrementing c_i^2 as long as the channel is idle. But, if it hears a data transmission, it freezes the counter, and waits in *R2-Watch* state for the channel to be idle for an IFS_2 duration (smaller than IFS_1 duration used by nodes in R1), then transmits a `busy` signal, and returns to the *R2-Countdown* state. When the counter c_i^2 reaches 0, it transmits the data frame. If it has more frames to send, it randomly picks a new first round counter, enters the *R1-Watch* state, and restarts the contention.

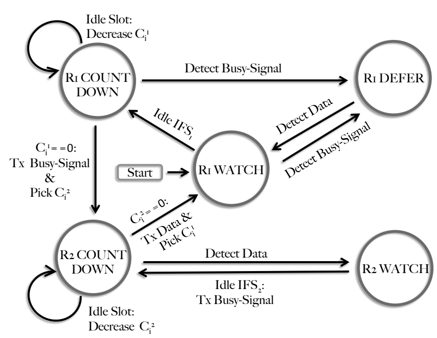

Figure 2: State transition diagram for the perspective of node i. The system starts in the *R1-Watch* state.

3.1 Adaptivity to Collisions

Although HiBo decreases collisions, they occur nevertheless. HiBo can be very conservative with $CW1 = CW2 = 32$. With this, not only does the collision probability become equal to WiFi's lowest (at CW=1024), but also ensures a low backoff overhead (32 instead of 512). However, this overhead could be nonnegligible for low density regimes, hence we consider a dynamic scheme.

Dynamic Rounds and Contention Windows. To curb backoff overhead in low density regimes, HiBo can start with ($CW1 = 8$, $CW2 = 8$), denoted as (8,8). In case of a collision, HiBo can switch from (8,8) to (8,16), (16,8), or (16,16). With (8,16) and (16,8), the backoff overhead and collision probabilities are same. However, when a node picks a larger $CW2$, it affects progress of other nodes in the first round that are awaiting its completion. On other hand, if it picks a larger $CW1$, it does not block any other node's progress. Therefore, it may be effective to switch from (8,8) to (16,8) to cope with a collision. In case of another collision, considering that equal size contention windows are better (as explained earlier), (16,16) should be the next choice. In the face of heavy congestion, the transitions could be (8,8), (16,8), (16,16), (32,16), (32,32). Even with (8,8), collision probability will be considerably lower than WiFi with $CW = 16$, therefore the frequency of transitions will be low. Now, once a transmission is successful, it may not be prudent to bring down the CW immediately (even though WiFi adopts such a policy). This is because HiBo designs for a much lower collision probability, hence, if a node still observes a collision, the congestion in the network is likely quite heavy. Our intuition suggests that a transmitter should perhaps drop down to a lower CW, say (16,16) to (16,8), only after a threshold number (six) of successful transmissions. This is of course a heuristic, reminiscent of the ARF rate control scheme in today's WiFi networks.

3.2 Multiple Collision Domains

We chose single collision domain for easier explanation, we now move to the more realistic case of multi domain contention. Fig 3 exemplifies the situation with 3 collision domains. Nodes in R2 are denoted by two concentric circles, while those in R1, with a single circle. Consider that node X has frozen its counter in R1 because nodes A and B have advanced to R2 (by sending a `busy` signal). Now, while A and B pick random numbers and count down, its possible that M and S in the adjacent collision domains begin transmissions. Nodes A and B obviously hear them and freeze their backoff counters; importantly, X does not hear either of these transmissions. Ideally, X should proceed with countdown and transmit in parallel to M and S. However, it does not know whether A and B are counting their slots or whether they are silenced by other transmissions. As a result, a spatial reuse opportunity is lost. Of course, such a situation does not happen

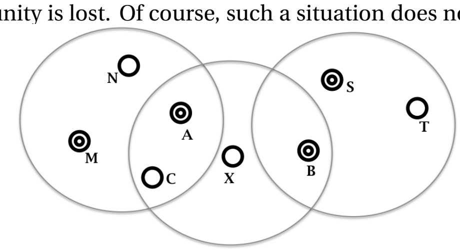

Figure 3: Example of 3 overlapping collision domains: Nodes in R2 denoted with concentric black circles, and nodes in R1 denoted by a single black circle.

if X were in R2 instead of R1, because, X would not hear M or S. It would continue counting down and transmit, exactly like WiFi.

In addressing this, we realize that the root problem arises from

X's inability to understand why nodes A and B are silent in R2. If some form of signaling could indicate the status of A and B, then X could make an informed decision. HiBo resorts to `busy` signaling again, requiring R2 nodes to signal to R1 nodes whenever they are counting down. Towards this goal, all nodes in R2 concurrently transmit a `busy` signal in every alternate slot. This `busy` signal is the same as the one that the nodes sent when they advanced from R1 to R2. All R1 nodes (awaiting the completion of R2) use a separate counter to count the idle slots after every `busy` signal. On detecting the next `busy` signal (either from A, or B or both), the counter is reset. If this counter counts to a value of 2, node X realizes that both A and B must have been silenced by other transmissions. Thus, X can now resume its countdown and potentially advance to R2, and complete transmission[1]. This enables the parallelism that we desire. Of course, since X has to wait for 2 slots before progressing into R2, HiBo would suffer a slight loss in performance, confirmed in simulations.

Although a R2 node transmits `busy` signals in alternate slots, it continues decrementing its counter on those slots. However, say on a given slot, s, node B transmits a `busy` signal, and senses the channel to be busy in slot $(s+1)$. This could happen because node S started transmitting a packet, either on slot s or $(s+1)$. If its the former, then node B should have not decremented its counter. Fortunately, WiFi data packets begin with 5 concatenated preambles, and the first preamble (called the short preamble) is different from the next 4. Thus, if node B observes a short preamble, then it infers that transmission started on slot $(s+1)$, otherwise on s. B adjusts its counter accordingly.

New Nodes Joining the Network. New joinees are introduced to R1. However they would start counting down only after waiting for two slots, thereby ensuring they won't count down while other nodes are contending in R2. By the end of two slots, presence or absence of `busy` signals will indicate whether nodes are active in R2 (Section 3.2). If nodes are present in R2, the new node will freeze its R1 counter. It will begin counting down in R1 otherwise.

4. o2CSMA

Our hierarchical backoff scheme is applicable to other CSMA protocols as well. oCSMA [9] is a distributed stochastic approximation algorithm to optimize a given utility function with CSMA. It prescribes values for random channel access probability (λ) and channel holding times (μ) based on a supply demand differential of packet queue lengths. The V parameter controls the trade-off between accuracy and convergence time of oCSMA. It has been shown that such scheduling converges towards optimality. However, the core assumption was that channel can be accessed at any time (i.e., unslotted), and other transmissions can be sensed instantaneously. This eliminates collisions. Follow up work has developed practical oCSMA [14], oDCF [11], relaxing these assumptions in oCSMA and implementing in a real testbed. They are prone to packet collisions because of the finite slot sizes. While oDCF is still quite effective, it resorts to packet aggregation to keep collisions low and channel utilization high. Packet aggregation introduces unfairness, showing that collision, utilization, and fairness is a zero sum game.

We believe o2CSMA breaks away from this zero-sum game at the cost of some signaling overhead. The key idea behind o2CSMA is simple. For the access probability λ selected by the oCSMA algorithm, the equivalent contention window (CW) is shown to be $\frac{2}{\lambda}-1$ [10]. We split this CW into two rounds of contention like HiBo, such that $CW_1 = CW_2 = \frac{CW}{2}$. This would decrease the collisions in oCSMA dramatically without performing packet aggregation. This makes o2CSMA more robust to the settings of parameter V. With oCSMA, higher values of V perform better in low density regimes, and the vice verse for denser networks. o2CSMA, on the other hand, extends consistently good performance. Finally, the PHY layer and the optimizations for multi-contention domain of o2CSMA is similar to HiBo.

[1]Its possible that when X is in R2, it transmits `busy` signals in the adjacent slots from A and B. However, it does not matter since other R1 nodes will hear `busy` signals on all slots.

5. PHY LAYER

We illustrate the challenges with an example. Consider 3 nodes A, B and C such that A and B each have a SNR of $4dB$ at C. With normal 802.11, node C would not detect either A or B given that the standard carrier sensing threshold is $6dB$, and hence, C should continue with its regular operation. However, with HiBo, when A and B win the first round together, their collision energy at C could be greater than $6dB$, making C an "over-exposed" terminal. Ideally C should continue counting down because none of the individual signals cross the sensing threshold. The problem is worse in reality when many more nodes collide in the first round. Furthermore, in the second round, the busy signals from all the nodes will also add up. This reduces spatial reuse of the channel, and the problem persists regardless of the choice of carrier sensing thresholds. In view of this, we need a technique that can examine whether a strong incoming signal ($> 6dB$ SNR) is actually composed of many weak busy signals.

We use a single 80 sample PN sequence as the busy signal. The choice of 80 samples is required to limit the detection time to less than one WiFi slot ($9us$). The nodes introduce a random jitter between 0 to 16 samples before transmitting their PN sequence. Now, our technique for detecting busy signals is simple. Nodes perform energy detection during every slot, essentially correlating the received signal with the known PN sequence. Since the colliders transmit their PN sequence with random jitters, multiple staggered peaks are expected in the output of correlation. The receiver extracts the following three metrics from the received signal: (1) Signal energy above the noise floor, called $CollisionEnergy$, and (2) Number of peaks detected by the correlator, denoted N_{peaks}, and (3) the correlation strength w_i of each peak. The receiver now decomposes $CollisionEnergy$ into N_{peak} components, where the energy of each component i is proportional to w_i. If the energy of any component is above the energy detection threshold, the receiver freezes its backoff counter; otherwise it continues counting down.

6. PERFORMANCE EVALUATION

We design experiments to evaluate the group signaling detection (PHY layer). We also quantify gains (MAC layer) via Simulations.

6.1 Busy Signal Detection on USRPs

Fig 4(a) shows how a bunch of 3db received signals (with an indoor testbed in office) can add up and easily overshoot the energy detection threshold which was designed for 6db detection. This causes overexposed terminals. HiBo however, can count the number of peaks and accurately infer the number of 3db transmitters. Peak detection is quite consistent with up to 4 transmitters – the mean error was around 11%. By normalizing the total detected energy over the number of colliders, (number of detected peaks), Fig 4(b) shows how HiBo is able to correctly detect overexposed terminals, while 802.11 fails consistently. Also, the accuracy of signal detection for 6db busy signals was 92%.

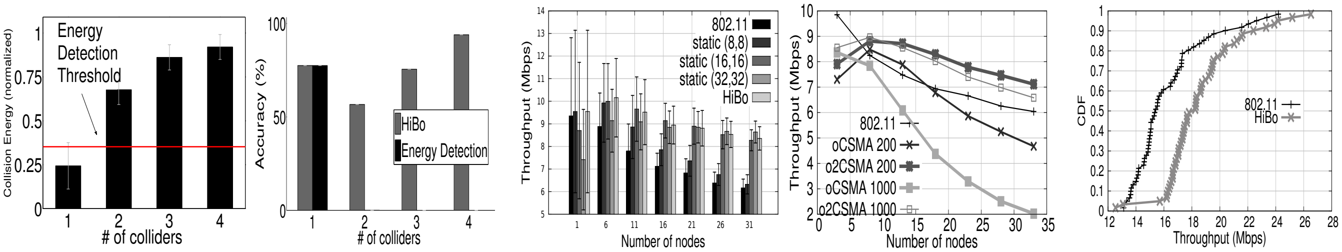

Figure 4: (a) Collision Energy passes the detection threshold with 3db transmitters (b) HiBo accurately detects 3dB over exposed terminals while 802.11 fails consistently (c) HiBo outperforms 802.11 (d) o2CSMA scales better than oCSMA (e)Throughput gains extend to multi contention domain

6.2 Simulation Study

We implemented HiBo and o2CSMA in NS3, by carefully choosing an indoor channel model and other MAC layer control algorithms. Nodes were scattered randomly around the AP. Fig 4(c) compares HiBo(includes results from static contention windows without adaptation) and 802.11 under a single collision domain, over 1-32 nodes, each transmitting 1000-byte packets of fully backlogged UDP traffic. Evidently, HiBo offers significant gains of up to 25% over 802.11 and even higher gains at higher densities. Fig 4(d) shows gains of o2CSMA over oCSMA for extreme values of the V parameter (without packet aggregation). Not only does o2CSMA outperform oCSMA but also demonstrates its robustness with the parameter settings. $V = 200$ with o2CSMA performs well for all regimes of node densities whereas the optimal parameter for oCSMA depends on node density. oCSMA curves have steep downward slopes (worse than 802.11) indicating poor scalability in the future whereas oCSMA scales quite well. Finally, Fig 4(e) gives results from a multi contention domain testbed with 2 APs, 3-32 nodes per AP. We generated 100 random topologies such that $20-30\%$ nodes fall under the overlapping region of two APs. Gains are similar to that of single-contention domain. Also, we do not compromise fairness.

7. LIMITATIONS AND OPPORTUNITIES

(1) **Energy implications.** Busy signaling between groups of nodes may appear energy-consuming. However, this may not be a serious concern given that transmission and carrier-sensing energy is comparable in modern WiFi cards [5]. Given that HiBo saves on backoff slots and collisions, the overall energy per bit may be lower. Precise characterization is left to future work.

(2) **Busy signaling and interference** HiBo's PHY layer solves most of the interference related problems associated with busy signaling like over exposed terminals. Also, nodes will perform carrier sensing and avoid busy signal transmissions during an ongoing transmission. However, two or more busy signals can combine and still interfere with a far away ongoing transmission. This issue persists in WiFi too where three or more nodes can together interfere with a far away transmission.

(3) **Generalized N-Round scheme** The core divide and conquer approach is generic and can be easily extended to N rounds. Compared to 2-round, 3-round contention has to additionally ensure that nodes in R2 abstain from contending with nodes in the third round, R3. This can be realized by having R3 nodes access the channel after IFS_3, such that $IFS_2 > IFS_3$ (Section 3). In general, with $IFS_1 > IFS_2 > IFS_3 ... > IFS_N$, an N-round scheme can work correctly. A lower-round node will continue to freeze as long as there are pending nodes at a higher round.

8. CONCLUSION

This paper is an early effort towards rethinking the backoff mechanism. The core improvement arises from the observation that the random number range to totally order all contenders is super-linear in the number of contenders. Partial ordering them in groups, followed by total ordering each smaller group, may together incur less time. Performance results confirm the intuition. While much remains to be done, we believe there is promise to pursue a longer-term research engagement.

9. ACKNOWLEDGMENT

We sincerely thank the anonymous reviewers for their valuable feedback on this paper. We are also grateful to NSF and Huawei. The research was partially funded by NSF (through grants – CNS-1441638 and CNS-1430033) and Huawei.

10. REFERENCES

[1] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE JSAC*, 2000.

[2] J. Capetanakis. Tree algorithms for packet broadcast channels. *Information Theory, IEEE Transactions on*, 1979.

[3] R. Garcés et al. Floor acquisition multiple access with collision resolution. In *ACM MOBICOM*, 1996.

[4] Z. Haas et al. On optimizing the backoff interval for random access schemes. *IEEE TRANSACTIONS ON COMMUNICATIONS*, 2004.

[5] D. Halperin et al. Demystifying 802.11 n power consumption. In *USENIX HOTPOWER*, 2010.

[6] M. Heusse et al. Idle sense: An optimal access method for high throughput and fairness in rate diverse wireless lans. 2005.

[7] C. Jackson. Dynamic sharing of radio spectrum: A brief history. In *IEEE DySPAN*, 2005.

[8] A. Jardosh et al. IQU: practical queue-based user association management for WLANs. In *MOBICOM*, 2006.

[9] L. Jiang et al. A distributed csma algorithm for throughput and utility maximization in wireless networks. *IEEE/ACM TON*, 2010.

[10] J. Lee et al. Implementing utility-optimal csma. In *IEEE Allerton 2009*. IEEE, 2009.

[11] J. Lee et al. Making 802.11 dcf near-optimal: Design, implementation, and evaluation. In *IEEE Secon*, 2013.

[12] E. Magistretti et al. WiFi-Nano: Reclaiming WiFi Efficiency through 800ns slots. In *ACM MobiCom*, 2011.

[13] T. Nandagopal et al. Achieving MAC layer fairness in wireless packet networks. In *ACM MOBICOM*, 2000.

[14] B. Nardelli et al. Experimental evaluation of optimal csma. In *IEEE INFOCOM*, 2011.

[15] T. Ozugur. Weighted Hierarchical Backoff Algorithm for Wireless Networks. In *IEEE Globecomm*, 2001.

[16] Q. Pang et al. Performance evaluation of an adaptive backoff scheme for WLAN. *IWCMC*, 2004.

[17] S. Sen et al. No time to countdown: migrating backoff to the frequency domain. In *ACM MobiCom*, 2011.

[18] K. Tan et al. Fine-grained channel access in wireless lan. *ACM SIGCOMM*, 2011.

[19] J. Yun et al. Mrca: multi-round collision avoidance for contention-based medium access control. In *IEEE WoWMoM*, 2007.