Compressing Backoff in CSMA Networks

Mahanth Gowda
University of Illinois (UIUC)

Nirupam Roy
University of Illinois (UIUC)

Romit Roy Choudhury
University of Illinois (UIUC)

Srihari Nelakuditi
University of South Carolina

***Abstract*—Randomized backoff is a well-established approach for avoiding collisions in CSMA networks. Today's backoff operation, such as in WiFi, attempts to create a total ordering among all the nodes contending for the channel. Total ordering requires assigning a unique backoff to each node, which is achieved by having nodes choose their back-offs from a large range, ultimately leading to channel wastage. This paper observes that total ordering can be achieved more efficiently. We propose "hierarchical backoff" in which nodes pick random numbers from a smaller range, resulting in groups of nodes picking the same number (i.e., partial order). Now, the group of nodes that picks the smallest number is advanced to a second round, where they again perform the same operation. This results in more efficient backoff because the time for partially ordering all nodes plus totally ordering each small groups is actually less than the time needed to totally order all nodes. Realizing the above intuition requires addressing new protocol challenges in group signaling, the feasibility of which is demonstrated on a USRP/GNUradio prototype. Large scale simulations also show consistent throughput gains by incorporating the proposed backoff approach into two CSMA protocols – WiFi and oCSMA. We also show that the proposed approach can be complementary to and even outperform existing backoff optimization schemes.**

I. Introduction

Channel contention is an extensively studied problem in CSMA networks. The seminal period dates back to 1973 with the introduction of ALOHA. The underlying idea is simple. Consider a group of nodes accessing a shared wireless channel (say in WiFi). To arbitrate this access, each node chooses a random backoff counter from a specific range of integers and decrements the counter till 0. The first node to reach 0 wins the contention and begins accessing the medium, while other nodes freeze their counters. Once the winner finishes transmission and releases the channel, the other nodes will resume counting down. Nodes that choose the same backoff counter will collide. Such nodes pick fresh backoffs from an exponentially larger range in an attempt to minimize future collisions. Upon successful transmission, a node would reset the random number range to its original (minimum) value. This core idea has grounded the design of many modern day CSMA protocols including WiFi and Zigbee.

Through randomized contention, today's WiFi tries to establish a total ordering among the nodes contending for channel access. A total ordering of all nodes ensures that no two nodes choose the same number. A closely packed total ordering minimizes channel wastage during count-down. Roughly, the random number range that ensures close packing for n nodes while minimizing collisions is super linear in n, i.e., $O(n^2)$.

We submit that totally ordering all nodes might be too strict a requirement. This might be analogous to asking all passengers boarding a flight to make one totally ordered queue – a cumbersome process. Our observation is that passengers can be ordered in the granularity of families, and the family that is at the head of the queue can undergo a total ordering of its members. As a result, every passenger need not compete with every other passenger to form the queue – the contention can first be between families, and then within families. We borrow this hierarchical intuition, and attempt to compress the effective backoff range for channel access.

Our main idea is simple. Consider n nodes. We first partially order them using a smaller backoff range. Suppose k groups of size approximately $\frac{n}{k}$ are formed (nodes picking the same backoff counter form a group). We then totally order the $\frac{n}{k}$ nodes in each group. WiFi roughly takes O(n^2) slots to totally order n nodes. On the other hand, we require appreciably lesser. We need O(k^2) slots for partially ordering into k groups and O$(\frac{n}{k})^2$ for ordering the $\frac{n}{k}$ nodes within a group. With suitable design choice for k, we can show that $k^2 + (\frac{n}{k})^2 < n^2$. Our proposition builds on this intuition that the range needed for partial ordering all nodes in k groups, plus the ranges needed to totally order each of the k groups, is still less than the range needed to total order all n nodes, leading to efficient contention resolution. However, building a system presents non-trivial challenges. We briefly introduce them next.

Unlike WiFi, where nodes pick random numbers in the range [0, CW], consider a two round contention scheme with a smaller backoff range of [0,$\sqrt{CW}$] for each round (Fig. 1 shows an example with $CW = 16$). Consequently, many nodes are likely to choose the same counter. The winners (group of nodes with the smallest counter) of the first round ($R1$) enter the second round ($R2$) by transmitting a `busy` signal (details in Section VI) similar to a preamble (C_2 and C_3 in Fig. 1). In $R2$, these nodes that are much fewer in number compared to the total number of nodes, perform a new contention by picking counters from [0,$\sqrt{CW}$]. The $R2$ winner transmits its packet first whereas other nodes freeze their counters. After this, other $R2$ nodes resume countdown and transmit packets.

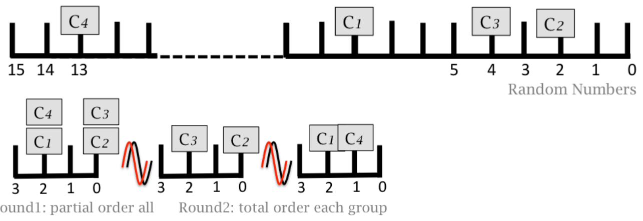

Fig. 1. Key observation behind hierarchical backoff (HiBo): Top WiFi timeline shows an attempt to total-order all nodes, while HiBo attempts to partial-order nodes (into groups) and then total order each smaller group.

Importantly, the losers of $R1$ (nodes C_4 and C_1) freeze their $R1$ counters until the $R2$ nodes have finished. Enforcing this is non-trivial and requires some signaling by $R2$ nodes (detailed later). Once all $R2$ nodes are done transmitting, the $R1$ nodes will resume counting down. Nodes finishing in $R2$ will rejoin $R1$ for a fresh contention. Fig. 1 shows an example. Thus, our scheme consumes only $\sim 3\sqrt{CW}$ slots whereas WiFi uses ∼CW slots, though both yield the same collision probability. Alternatively, splitting a single round contention window of CW into 2 rounds of $CW/2$ each will decrease the collision probability. We take advantage from both backoff compression and collision avoidance.

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 standard WiFi. Beyond WiFi, schemes such as IdleSense [7], [10] and oCSMA [13] that attempt to optimize wireless performance can all benefit from the collision reduction due to the proposed approach. For instance, utility optimal CSMA (oCSMA) 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 [19] suffer from poor scalability (Section VII). We show that o2CSMA, which incorporates the proposed hierarchical backoff approach into oCSMA, can address this problem.

Of course, this is only a sketch of the protocols and several component challenges must be addressed. (1) With groups of nodes signaling each other, detection techniques should detect concurrent signals 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, nodes should be able to adapt their backoff in response to collisions.

This paper designs techniques to handle these challenges systematically. Real world measurements as well as simulations are used to inform parameter choices for the protocol. A USRP testbed verifies the PHY layer techniques, while NS3 simulations evaluate large scale scenarios. 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 improves considerably across all network densities. Our main contributions can be summarized as:

1) **Identifying the super-linear behavior in total ordering all nodes.** Although the intuition is age old, we believe its application to backoff is unexplored.
2) **Translating the intuition to a protocol and developing a reliable group signaling schemes (via correlation techniques).** The ability for a group of nodes to reliably announce information will likely be useful elsewhere.
3) **Prototyping on USRP/GNUradio boxes, alongside NS3 simulation for large, high-density networks.** We show consistent gains against WiFi and oCSMA [19].

II. Some Natural Questions

(1) *Is backoff an important problem to solve?*
Our interest in revisiting the backoff scheme is three-fold.

(a) Backoff is a popular approach to decentralized resource sharing in several technologies, not just WiFi. Any improvement in its core structure can help lower the gap between PHY layer capacity and MAC layer throughput.

(b) In the specific instance of WiFi, the need is more urgent. With higher data rates, the air-time of data packets is fast decreasing, however, backoff durations continue to remain the same because of unchanging slot times. Fig 2(a) reports substantial wastage due to backoff against varying data rates, for 1000 byte packets.

(c) A surge in network density is round the corner, given that a large number of "things" (IoT) will be WiFi-empowered (WiFi-enabled thermostats and light bulbs are already available commercially). The current backoff structure is not designed to scale – curbing collisions without increasing channel idle time becomes harder at high densities. Fig 2(b) captures this tradeoff. While backoff-related waste may appear to go down with higher densities, collisions increase quickly. Fairness suffers as well. We aim to restructure backoff to scale it to higher densities, while preserving the desirable properties in today's standard protocol.

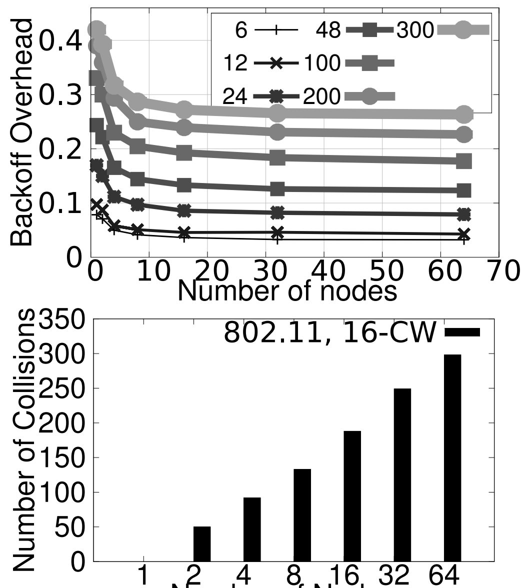

Fig. 2. (a) Channel wastage due to backoff at different data rates in Mbps. The wastage is higher at low densities. (b) However, collisions increase at high densities.

(2) *Backoff is well studied – what's novel in this proposal?*
We are aware that "distributed resource sharing" is a mature area and we have surveyed the literature to our best ability. As detailed in Section III, we found large bodies of work on modeling and analysis of the backoff framework [1], [3], [16], [26]; stability regions and fairness [18], [21]; increase/decrease mechanisms in response to congestion [23]; energy efficiency [24]; and special cases such as directional antennas and multi-

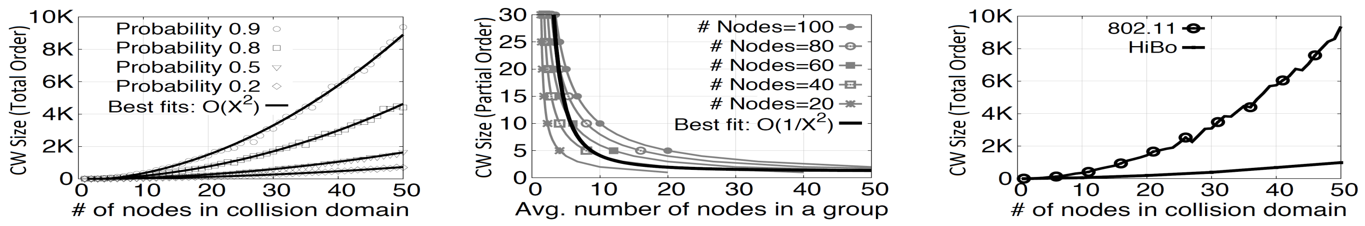

Fig. 3. (a) Number of slots needed for total ordering grows super-linearly. (b) Slots for partial ordering drastically drops down as the group size increases. (c) Difference between WiFi backoff and HiBo, for equalized collision probability.

channel systems [8]. While truly deep, most of them do not attempt to change the core architecture of backoff. Only recently, some efforts have been made towards architectural revisions. Ideas such as frequency-domain backoff [22] and WiFi-Nano [17] have proposed elegant clean-slate techniques (based on sophistications such as interference cancellation and hardware enhancements). Inspired by these, we are investigating if improvements are possible without complete redesign. Our core intuition of divide and conquer is not new, and has been employed before [4], [5]. However they operate at packet granularities (Section III). We resolve contention at much finer granularities of WiFi slot lengths. This entails new challenges which is the main focus of this paper.

(3) *How much is the room for improvement?* In Fig 3(a), we use simulation to determine the number of slots (i.e., CW) needed to totally order n nodes. The curve marked "Probability 0.9" means: there is a 90% chance that 2000 slots will totally order 23 nodes. Observe that this value of CW grows super-linearly in the number of nodes, $O(n^2)$, as derived from simple regression; analytical results from literature also confirm this behavior. Now, Fig 3(b) shows that the required CW is dramatically lower for partially ordering nodes into groups. In other words, partial ordering can save a lot of slots. Thus, while WiFi requires $O(n^2)$ slots, HiBo creates k groups in round 1 (which consumes $O(k^2)$ slots), followed by k second rounds, each $O((n/k)^2)$ slots. Fig 3(c) roughly quantifies the improvement of HiBo over WiFi. Of course, WiFi may not totally order nodes and might use smaller contention windows. However, this would increase collision in high density settings. HiBo, however, achieves smaller backoff without increasing collision probability.

III. Related Work

Before we present the details of the design of HiBo and o2CSMA, we briefly discuss the relevant ideas from the vast literature [11].

Hierarchical Backoff: Hierarchical contention and tree-based collision resolution algorithms have existed for decades [4], [5]. However, we find that tree-splitting algorithms operate at the granularity of packets, i.e., whenever RTS/Data collisions happen [5], about half of the colliding nodes defer and reattempt later for eventual contention resolution. An entire packet duration is wasted during such resolutions. HiBo, however, resolves 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 entails new challenges associated with busy signal decoding and multi-contention domains, which is the main focus of this paper. Closest to ours is the work in [27] 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 [20], where spatially clustered nodes choose a leader who backs off on their behalf. The scheme is again designed for single collision domain and relies on heavy control traffic, making it impractical for real networks. A recent workshop paper [6] proposes a similar scheme but lacks in design details and comprehensive evaluation.

Adaptation to Contention: The basic backoff proposal has been optimized for various network parameters. Authors in [18], [21] optimize the count-down for collisions, while [8] 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 backoff design arose from FICA [25], Back2F [22], and WiFi-Nano [17] 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 minor modifications. WiFi-Nano [17] was also inspiring in their use of correlation enabled primitives of "group coordination" that influenced our thinking. Ideas in [12] were also compelling in characterizing WiFi's problems with scalability. While HiBo and o2CSMA are ideas that would have not been conceived in the absence of these existing works, we argue that these systems are completely different. The ability to separate groups of nodes in time blocks via concurrent signaling and detection is the key departure.

Optimizations to Backoff Window: Several works have proposed optimizing back-off windows based on participation and interaction with other nodes in the system. oCSMA [13] is a distributed stochastic approximation algorithm to optimize a given utility function with CSMA. It prescribes values for random channel access probability (or contention window size) and channel holding times based on a supply demand differential of packet queue lengths. Practical versions of oCSMA [14], [19] and other interesting extensions to handle fairness issues have been proposed in oDCF [15]. As we will show in Section V and Section VII, our scheme could be complementary to these approaches and even outperform them. Similarly, the optimization of contention windows like IdleSense [7], [10] can be applied to HiBo to identify optimal contention windows of round-1 and round-2, and it is complementary to HiBo. Backoff tuning (for number of contenders, hidden terminals etc) as proposed in [2], could also benefit from HiBo which decreases the collision probability and enhances performance.

Overall, we note that the proposed schemes are distinct and complementary to previous proposals towards improving backoff overhead and collision probability in wireless networks.

IV. HiBo Design

The design firmed up after many iterations. To convey the rationale behind the final design, we describe a basic design for single collision domains and relax assumptions later.

A. Two Round HiBo

To transmit each packet, a node 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 one slot, 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 counting only after nodes i and j have contended in R2 and completed their transmissions. We will shortly discuss how to ensure this, and later describe how 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, to prevent their 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 4 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 standard). 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 be 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 it senses the channel 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 begins the contention process again.

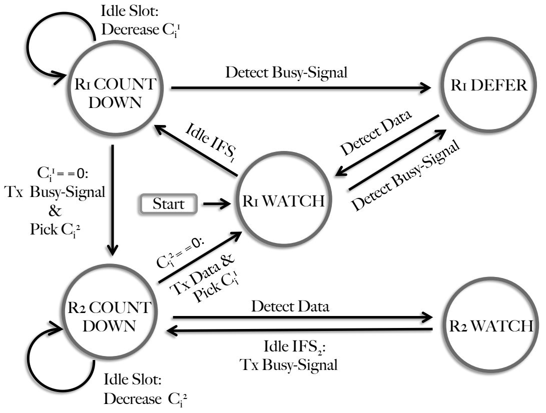

Fig. 4. State transition diagram from the perspective of node i. The system starts in the *R1-Watch* state.

B. Generalized N-Round HiBo

The above description resolves contention in two rounds. However, the core divide and conquer approach is generic and extensible to N rounds. Compared to 2-round contention, 3-round has to additionally ensure that nodes in R2 abstain from contending with nodes in the third round, R3. To achieve this, R3 nodes must access the channel after IFS_3, a smaller inter-frame spacing than IFS_2. 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.

Such generalization creates higher gains with more rounds, particularly with heavy node density. However, with few contenders, the overhead of one slot per round (to convey a busy signal while progressing through each round) can be a wastage. Hence, the number of rounds is an important design choice, but we use 2 rounds here for simplicity. Also, the length of a round depends on the number of rounds too. For a 802.11 contention window of CW, HiBo can maintain the same average backoff by picking its 2-round CWs as $CW1 + CW2 = CW$. By designing $CW1 = CW2 = \frac{\text{CW}}{2}$, the collision probability between 2 contending nodes can be minimized to $\frac{1}{(CW/2)^2} = \frac{4}{\text{CW}^2}$ compared to $\frac{1}{\text{CW}}$ in 802.11. Similarly for 3 rounds, using $\frac{\text{CW}}{3}$ per round minimizes collision probability for the same average backoff.

Equalizing collisions or backoff. The above discussion points to a useful property in terms of collision and backoff. Specifically, HiBo can be designed to equalize WiFi's collision or backoff, and reduce the other. Consider a 2-round scheme with $CW1 = 8$ and $CW2 = 8$ and contrast it with that of WiFi with $CW = 16$. While the average number of backoff slots are equal for both WiFi and HiBo, the collision probability with HiBo would be lower. For a two node example, collision probability with WiFi would be $\frac{1}{16}$, whereas HiBo lowers it to $\frac{1}{8} \times \frac{1}{8} = \frac{1}{64}$. Alternatively, HiBo can reduce the backoff overhead by half with $CW1 = 4$ and $CW2 = 4$, which then maintains the collision probability as $\frac{1}{16}$, equal to WiFi. Ideally, we should minimize the collision probability, particularly when the contention is high, and reduce backoff wastage when the contention is low. This raises the question of adaptivity in HiBo, discussed next.

C. Adaptivity to Collisions

Although HiBo decreases collisions, they still occur. HiBo can be very conservative with $CW1 = CW2 = 32$. With this setting, the collision probability with HiBo equalizes with WiFi's lowest, which happens at a much higher CW size of 1024. Moreover, this also ensures a low backoff overhead (32 instead of 512 with WiFi), which could still be non-negligible for few nodes. 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 $(8, 8)$. On 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 the contrary, 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 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. We suggest 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.

We now relax the assumptions we made at the beginning. We begin by extending HiBo to multiple collision domains.

D. Multiple Collision Domains

We chose single collision domain for easier explanation, we now move to the more realistic case of multi domain contention. Fig 5 illustrates 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 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. 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 count-down 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 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.

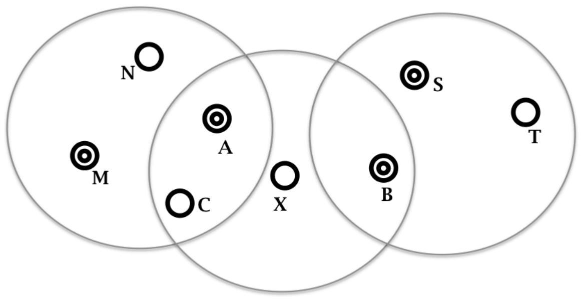

Fig. 5. Example of 3 overlapping collision domains: Nodes in R2 denoted with concentric black circles. Nodes in R1 denoted by a single black circle.

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 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 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 finish transmission[1]. This enables the desired parallelism. Of course, since X has to wait for 2 slots before progressing to R2, HiBo would suffer a slight loss in performance.

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.

E. New Nodes Joining the Network

New joinees are introduced to R1. However they will start counting down only after waiting for 2 slots, thereby ensuring they won't count down while other nodes are contending in R2. By the end of 2 slots, presence or absence of `busy` signals will indicate whether nodes are active in R2 (Section IV-D). If nodes are present in R2, the new node will freeze its R1 counter. Otherwise it will begin counting down in R1 by immediately decreasing its backoff by 2 because the channel was idle for two previous slots.

V. o2CSMA

Our hierarchical backoff scheme is applicable to other CSMA protocols as well. oCSMA [13] 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 [19], oDCF [15], 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 $\frac{2}{\lambda}-1$ [14]. 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.

[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.

VI. PHY Layer

So far, we assumed reliable detection of busy signals. We now discuss the actual PHY layer challenges. Consider 3 nodes A, B and C such that A and B each have a SNR of $4dB$ at C. With 802.11, node C would detect neither A nor B given that the standard carrier sensing threshold is $6dB$, and hence, C should continue with its regular operation. However, 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 nodes collide in the first round. Moreover, in the second round, 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. Hence, we need a technique that can examine whether a strong incoming signal ($> 6dB$ SNR) is actually composed of many weak busy signals.

Addressing Over-Exposed Terminals: 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 (the reason will become clear shortly). Now, our technique for detecting the 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.

VII. Performance Evaluation

We first evaluate the PHY layer of HiBo using real USRP experiments. Since we are interested in high node densities, we use measurement driven NS3 simulations later to evaluate HiBo over dense wireless networks.

A. Busy Signal Detection on USRPs

Experimental set up: Our experimental platform consists of 5 USRP N210 nodes operating in the 5 GHz band with a 5

MHz bandwidth. The goal is to accurately (90% as required by 802.11) detect signals in the carrier sensing range (usually 6 dB) and accurately reject multiple sub 6dB colliding preambles that might add up to 6dB and cause overexposed terminals.

In order to test the worst case conditions, we attempted detecting $6dB$ signals and rejecting collisions of multiple $3dB$ signals. Signal power was accurately controlled based on mutual reciprocity property. A master node would send a preamble, and multiple colliders (slaves) would estimate the power of the received preamble. They would reply with their own preambles such that it reaches the master at the required power. Additionally, the slaves inject different jitters in the 80 sample PN sequence as required by the PHY. The entire logic was implemented in FPGA.

(1) Energy Threshold-based Detection: Fig 6(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. Evidently, this causes overexposed terminals.

(2) Peak Counting Accuracy: To evade over exposed terminals, the PHY counts peaks and reliably infers the number of 3dB colliders (Fig 6(b)). Peak detection is quite consistent – the mean error was around 11%.

(3) Detecting Per-Collider Energy: Fig 6(c) shows the normalized per peak energy of 3 dB colliders (computed as described in Section VI) in comparison with total detected energy. Even though the total energy is above the threshold, the per-peak energy is under the threshold, thereby mitigating over exposed terminal problem.

(4) Accuracy of Detecting Over-Exposed Terminals: While WiFi energy detection can detect 80% of over exposed terminals with one transmitter, it fails completely with more transmitters. The correlation and peak-counting technique detect over exposed terminals with an accuracy of 57%, 75%, and 94% for 2, 3, and 4 colliders, respectively.

(5) Accuracy of Detecting Valid Signals: Finally Fig 6(e) shows that we can accurately detect valid $6dB$ signals with over 90% accuracy with varying number of colliders. Note that the detection accuracy over noise is 100% (using low energy detection threshold), but it increases over-exposed terminals.

B. Simulation Study

We implemented various components of HiBo by incorporating changes in the MAC layer of NS3. We use *log distance path loss* (LDPL) model within the NS3 framework to model propagation losses. The parameters are chosen such that the AP range is 60 meters, in the 2.45 GHz spectrum. We use convolutional coding based error models and SNR look up table based rate selection. Nodes are scattered randomly around the AP in single and multi domain experiments. Results are generated from 50 simulations of random topologies for each case. The following study will extensively test HiBo for throughput savings, collision avoidance, window size adaptations, fairness, robustness to packet sizes, data rates etc. In addition, performance gain of o2CSMA and its robustness to various parameters are also studied.

1) HiBo: We first provide evaluation results for HiBo in this section. o2CSMA is considered next.

Throughput: Fig 7(a) shows the resulting average throughput per node under single collision domain experiments with fully backlogged UDP traffic. When there is very little contention, both schemes yield similar throughput. As the number of contending nodes increases, the average throughput decreases for both schemes. However, throughput under 802.11 degrades steeper than that under HiBo. To get a better sense of relative performance, we plot the throughput gain with HiBo over 802.11 in Fig 7(b). It is evident that HiBo offers significant gains of up to 25% over 802.11. Moreover, the higher the contention, the larger the gain, indicating that HiBo copes better with contention.

The two possible sources for gain include: (1) smaller backoff overhead and (2) lower collision probability. We plot the break up in Fig 8. Fig 8(a) shows the collisions under HiBo relative to 802.11. HiBo incurs less than a third of the number of collisions incurred by 802.11. The difference in backoff overhead between HiBo and 802.11 as indicated in Fig 8(b) is also striking. While the average backoff overhead with 802.11 increases significantly up to 20%, it stays consistently below 14% with HiBo. Fig 8 essentially affirms the intuition behind this work – a multi-round contention scheme like HiBo decreases the collision probability without increasing the backoff overhead.

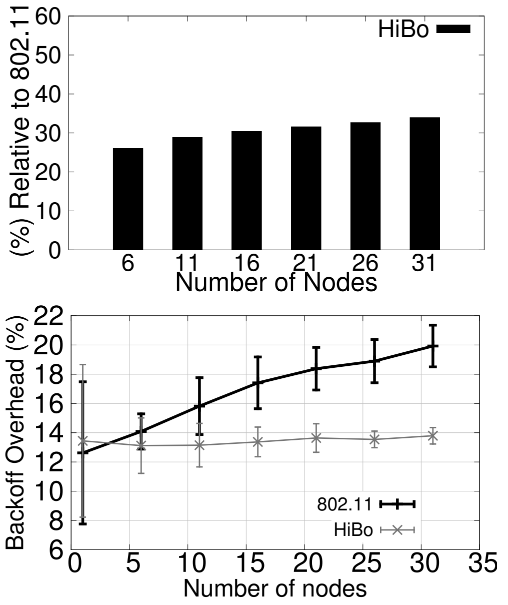

Fig. 8. Collision probability and backoff overhead: (a) Collisions under HiBo as a percentage of collisions under 802.11. HiBo incurs less than a third of the number of collisions incurred by 802.11; (b) Backoff overhead is the percentage of channel time that is spent in backoff. Unlike HiBo, 802.11 incurs increasingly higher backoff overhead.

Comparison with oDCF: oCSMA based scheduling was discussed in Section V. It has been shown that such scheduling converges towards optimality. However, the core assumption in oCSMA theory 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

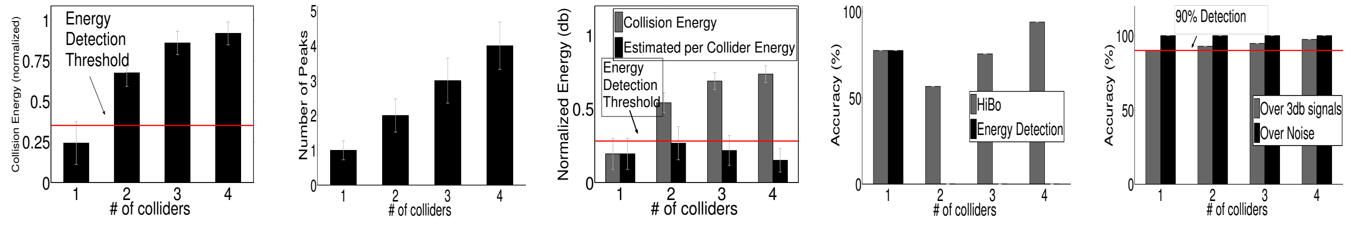

Fig. 6. (a) Collision Energy as a function of number of 3 dB transmitters (b) Number of detected peaks as a function of number of 3 dB colliders (c) Resolved Per collider energy (d) Accuracy of detecting 3dB over exposed terminals (e) 6dB signal detection accuracy.

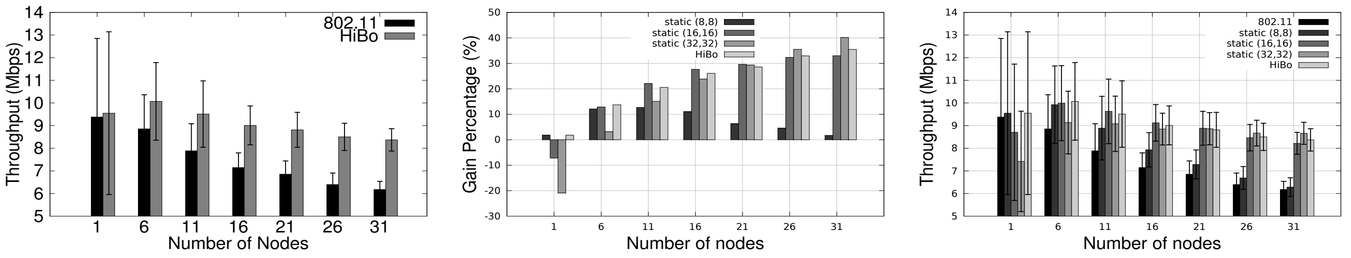

Fig. 7. Performance of HiBo *vs* 802.11: (a) throughput; (b) relative throughput gain; HiBo yields consistently better overall throughput than 802.11 (c) Performance of adaptive scheme is comparable to that of an ideal scheme that chooses the best static setting at any instant.

has developed oDCF [15], [19], relaxing these theoretical assumptions in oCSMA and implementing it on a real testbed. 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.

Although we adapt HiBo as complementary to oCSMA as explained in Section V, here we perform a direct comparison of oDCF (which is also based on oCSMA) with HiBo. We believe HiBo breaks away from this zero-sum game at the cost of some signaling overhead. For fair comparison, we configure oDCF to turn off its proportional fairness property (which HiBo could also adopt). However, we allow oDCF to use its packet aggregation feature and evaluate it for various degrees of aggressiveness (controlled by a parameter V). HiBo on the other hand does not benefit from packet aggregation. Figure 9 shows that higher values of V perform better in low density regimes, and the vice versa for denser networks. HiBo, on the other hand, extends consistently good performance. However, unlike oDCF, HiBo may not be good in handling asymmetric topologies like Flow-in-the-Middle and asymmetric interference. Hence we augment HiBo into o2CSMA in Section V to achieve best of both worlds.

Adaptivity: As described in Section IV-C, HiBo chooses conservative contention windows in response to collisions. To understand adaptivity to collisions, Fig 7(c) compares the throughput gain over 802.11 for adaptive HiBo and that with static contention window set to (8,8), (16,16), and (32,32). The adaptive scheme performs similar to (8,8) under low load and much better at high loads. Compared to settings of (16,16) and (32,32), adaptive scheme is better at low loads, similar at high loads. Overall, across all loads, it offers similar or better performance than the best static setting for that load. Based on this, one could surmise that 802.11 can reduce

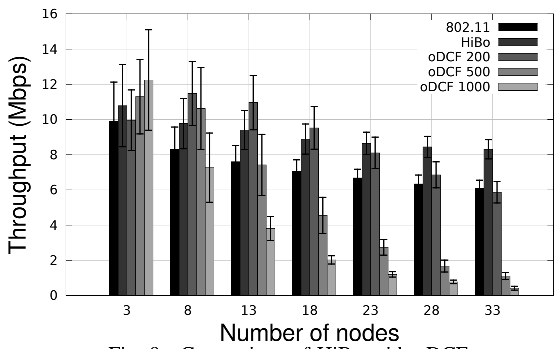

Fig. 9. Comparison of HiBo with oDCF.

collision probability aggressively by picking a larger minimum contention window or quadrupling it after each collision. But with 802.11, going from contention window of 16 to 64, the collision probability of two contenders goes down from 1/16 to 1/64. In contrast, with HiBo, for the same backoff overhead, we can switch from (8,8) to (32,32), drastically reducing the collision probability from 1/64 to 1/1024. This argument is supported by Fig 8. If 802.11 aggressively increased the contention window to curb collisions in Fig 8(a), the backoff overhead will be much worse in Fig 8(b). In contrast, with 2-round contention, we can curb collisions without increasing backoff overhead.

Fairness: HiBo's throughput gains do not sacrifice fairness. Fig 11(a) plots the fairness (averaged over 100 runs) of 802.11 and HiBo for UDP traffic with packet size 1000 bytes. At all loads, HiBo offers better fairness than 802.11, more so at higher loads. To reduce backoff overhead, after a successful transmission, 802.11 resets the contention window to the minimum, which can cause unfairness to the other pending nodes. In contrast, since the backoff overhead is already low under HiBo, it does not have to make the same trade-off, thus providing a fairer access for all nodes.

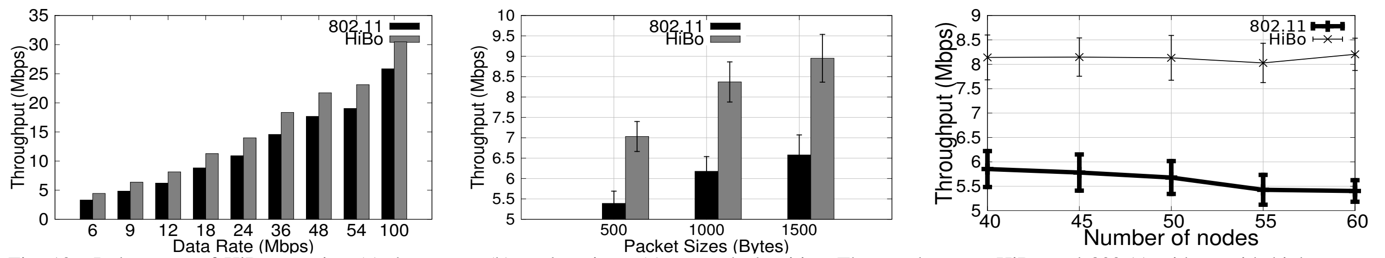

Fig. 10. Robustness of HiBo: varying (a) data rates; (b) packet sizes; (c) network densities. The gap between HiBo and 802.11 widens with higher rates, smaller packets, and heavier loads.

Robustness: Fig 10 evaluates the robustness under various conditions. Unless otherwise stated explicitly, we assume 31 nodes are transmitting UDP traffic with packet size 1000 at the fixed rate of 54Mbps. Fig 10(a) shows the throughput at various data rates of up to 100Mbps. Fig 10(b) gives the same for 3 different packet sizes. Fig 10(c) presents the performance when the number of nodes goes up from 40 to 60. Across all these scenarios, HiBo outperforms 802.11, with larger gain at higher data rates and denser networks.

We analyzed the efficacy of HiBo on a single collision domain so far. We present the average throughput and fairness results for multi-collision domains with two APs in Fig 11(b) and Fig 11(c). Evidently, performance improvements offered by HiBo extend to multiple collision domain scenario too.

UDP traffic helps characterize MAC layer throughput. Fig 12(a) and 12(b) show the average throughput and fairness achieved by 802.11 and HiBo for TCP traffic in multi-collision domain scenario. With TCP traffic, due to flow/congestion control, fewer nodes are likely to contend at any instant. This does not saturate the channel, and hence, gains are less. WiFi experiences fewer collisions due to rate throttling from the TCP source. Fairness is not compromised for throughput.

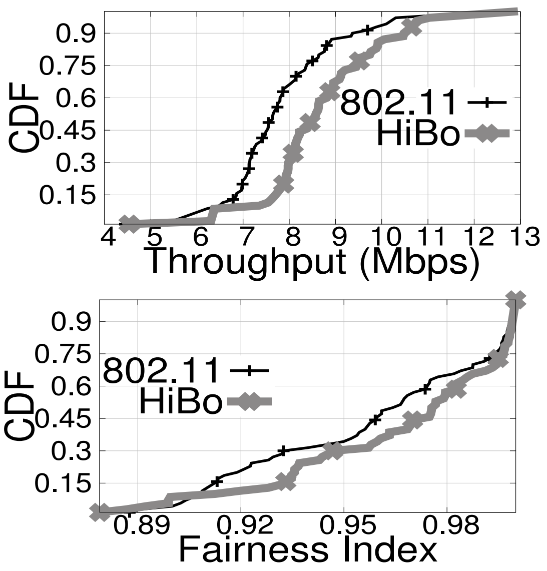

Fig. 12. (a) Multi domain TCP Throughput, (b) TCP Fairness

In retrospect, HiBo achieves the following desirable properties. (1) Resolves contention and avoids collisions even under dense contention. (2) Allows quick channel access, especially when there are only a few contenders. (3) Ensures fair channel access by all nodes, regardless of position.

2) o2CSMA: The set up of o2CSMA is similar to HiBo's single collision domain evaluation with backlogged UDP traffic. We verified that o2CSMA satisfies proportional fairness requirements under special topologies like "Flow in the Middle". We do not include these results in the interest of space.

Collisions: Fig 13(a) plots collision rates for different choices of V. Observe how collisions increase in oCSMA, particularly at high node densities. Channel access probabilities assigned to nodes by oCSMA is independent of slot sizes. However, non-negligible slot sizes will induce high collision rate under higher densities because of increased contention aggressiveness under oCSMA. o2CSMA decreases the collision probability significantly because of 2 round contention. The decrease is roughly around $40 - 50\%$

Throughput: Fig 13(b) shows how collisions have translated into throughput. The throughput in oCSMA decreases with node density because of increased collisions. Also note that there is no single choice of the V parameter that works best for all node densities. With small V, the performance is low at low node densities because of wasted communication cycles (small V causes low transmission aggressiveness and vice-verse). On the other hand, high V has resulted in poor performance in high node densities because of increased collisions. The figure shows the severity of performance degradation over 802.11. By minimizing the collision-rate with o2CSMA, we are able to reclaim the efficiency.

Parameter Sensitivity: Fig 13(c) evaluates the sensitivity of o2CSMA and oCSMA to the V parameter for different node densities. As explained before, the V parameter controls the tradeoff between convergence time of the protocol and efficiency. The variance of the curves for oCSMA is wide. Some values (like 200) perform well with high densities, whereas others (like 1000) perform well in a low density scenario. In contrast, with o2CSMA, a single value (like 1000) could be robust across a larger range of node densities.

VIII. Limitations and Opportunities

(1) **Energy implications.** Busy signaling may appear energy-consuming. Essentially, we partially replace carrier sensing in WiFi with busy signaling in HiBo. Given that transmission and reception energy is not very different in WiFi cards [9], energy consumption with busy signaling is not very different from carrier-sensing. Moreover, since HiBo saves on backoff and collisions, the overall energy requirement is lower.

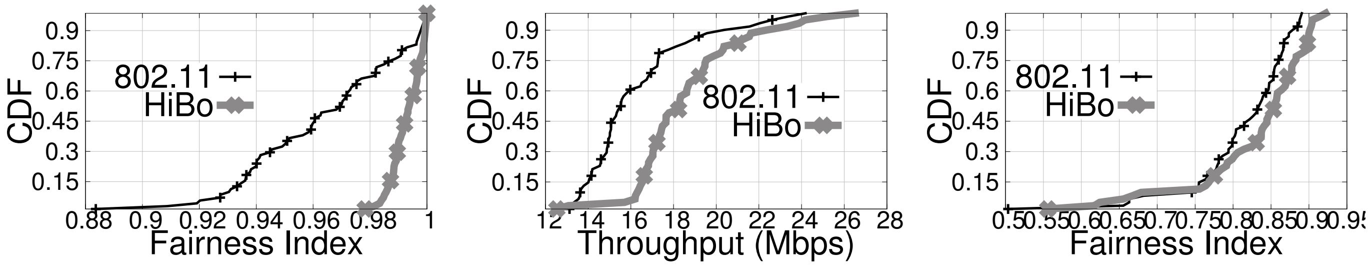

Fig. 11. (a) Single domain fairness (b) Multi domain UDP throughput and (c) Multi domain Fairness

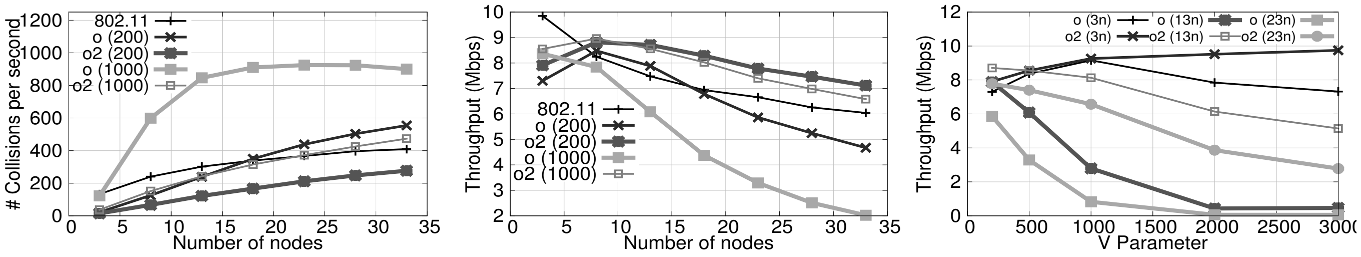

Fig. 13. Performance of o2CSMA (a) o2CSMA minimizes collisions (the legend with o(200) here refers to oCSMA with V=200 and similarly o2 refers to o2CSMA); (b) Higher throughput with o2CSMA, particularly at high node densities where oCSMA under-performs; (c) o2CSMA is robust to V parameter settings (o(3n) here refers to the performance of oCSMA in the setting with 3 nodes).

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

IX. Conclusion

Backoff mechanism in WiFi, has received recent interest, because the fundamental limitation on slot sizes is becoming a bottleneck. Ours 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. Partial ordering them in groups, followed by total ordering each smaller group, together incurs less time. Performance results confirm the intuition. While much remains to be done, we believe that there is enough promise to pursue a longer-term research engagement.

References

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

[2] F. Calì et al. Dynamic tuning of the ieee 802.11 protocol to achieve a theoretical throughput limit. *IEEE/ACM ToN*, 2000.

[3] F. Cali et al. IEEE 802.11 protocol: design and performance evaluation of an adaptive backoff mechanism. *IEEE JSAC*, 2002.

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

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

[6] M. Gowda et al. Backing out of linear backoff in wireless networks. In *ACM HotWireless*, 2014.

[7] Y. Grunenberger et al. Experience with an implementation of the idle sense wireless access method. In *CoNEXT*, 2007.

[8] Z. Haas et al. On optimizing the backoff interval for random access schemes. *IEEE Transactions on Communications*, 2004.

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

[10] M. Heusse et al. Idle sense: An optimal access method for high throughput and fairness in rate diverse wireless lans. In *ACM SIGCOMM*, 2005.

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

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

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

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

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

[16] H. Ma et al. Dynamic optimization of IEEE 802.11 CSMA/CA based on the number of competing stations. In *ICC*, 2004.

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

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

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

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

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

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

[23] N. Song et al. Enhancement of IEEE 802.11 distributed coordination function with exponential increase exponential decrease backoff algorithm. In *VTC*, 2003.

[24] P. Soni et al. Energy efficiency analysis of link layer backoff schemes on point-to-point Markov fading links. In *IEEE PIMRC*, 2002.

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

[26] V. Vitsas. Throughput analysis of linear backoff scheme in wireless LANs. *Electronics Letters*, 39(1):99–100, 2003.

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