Contention Window Adaptation using the Busy-Idle Signal in 802.11 WLANs

Michael N. Krishnan, Shicong Yang, and Avideh Zakhor
Department of EECS, U.C. Berkeley
Email: {mkrishna, yangshic, avz}@eecs.berkeley.edu

***Abstract*—In a wireless local area network (LAN), packets can be lost for a variety of reasons, including collisions due to high traffic and channel errors due to poor channel conditions. In practice, however, nodes cannot easily differentiate between these types of loss. As a result, adaptations based on packet loss alone can result in significantly degraded performance. In 802.11 networks, wireless nodes avoid collisions via the Binary Exponential Backoff (BEB) protocol. This performs well for moderate numbers of nodes and low channel error rates, but is inefficient for large numbers of nodes, high channel error rates, or in the presence of hidden terminals. In this paper, we propose a contention window adaptation scheme in which nodes use information shared by the AP to optimize contention window sizes in a distributed fashion to improve network utility. We show via NS-2 simulations that our method can improve throughput by as much as 24% in the high node count scenario, 35% in the high channel error scenario, and 350% in the presence of hidden terminals.**

I. Introduction and Related Work

In 802.11 networks, wireless nodes avoid collisions via the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol with the Binary Exponential Backoff (BEB) algorithm. In CSMA/CA, when a node has a packet to send but senses the channel as busy, it choses a random number W between 0 and CW, the contention window size, and waits until it observes the channel as idle for a total of W slots before beginning its transmission. This way, if competing nodes choose different values of W, they can avoid collision. Ideally CW should be large enough that the probability of multiple nodes completing their backoff in the same slot is sufficiently small, but also small enough to avoid excessive delay. In the BEB protocol, CW is adjusted with every retransmission attempt of a packet. It initially starts at a value $CW_{min} = 32$ and increases by a factor of $\alpha = 2$ for each retransmission attempt up to $CW_{max} = 1024$. The rationale is that a relatively small CW should be used to limit delay when there is low collision probability, but when there are many nodes, collision probability increases, and a larger CW is needed. This is based on the often invalid assumption that all losses are due to neighboring nodes competing for the channel; in practice losses can also be caused by poor channel conditions or collisions with hidden terminals. Additionally, the BEB protocol suffers when there is a sufficiently large number of nodes such that $CW_{min} = 32$ is too small to prevent collisions. Thus there are three major situations in which standard contention window adaptation is ineffective. The first is when the number of nodes is large, the second is when the number of nodes is small, but there are is a poor channel, and the third is in the presence of the hidden terminal problem.

Most suggested modifications to contention window adaptation tend to focus on the first case. This is because contention window adaptation was originally designed for collision avoidance. In [1] and [2], analytical models are derived to determine the optimal CW as a function of the number of nodes assuming all losses are due to collision. In [3], an algorithm with memory is proposed using packet loss counts to keep an updated estimate of the level of traffic; however, since only local loss rates are used, it is impossible to distinguish between losses due to collisions and those due to channel errors. There are also several other adaptation schemes which update the contention window in a Markov manner that differs from the BEB [4–7].

Contention window adaptation in the situation with a high probability of channel error is largely ignored in the literature, but in practice is perhaps more common than the high collision situation, and can lead to similar levels of throughput loss. As a simple example, consider the scenario where there is only a single node with a high loss rate due to a weak signal to the AP. Using standard CW adaptation, the node increases its contention window after each loss, resulting in contention window sizes of up to 20 ms, during which time the node could have attempted as many as 10 more transmissions of 2 KB packets at 11 Mbps. A major reason this is ignored is the difficulty in distinguishing between collisions and channel errors using only local information. We have recently developed a method in which APs can share their local channel occupancy information with associated nodes at an overhead of less than 2% using the binary-valued *busy-idle signal* [8]. With this information, nodes are able to estimate their probability of various loss types, including direct collisions with neighboring nodes, channel errors, and staggered collisions caused by hidden nodes. In this paper, we show that this ability to differentiate between types of loss can be beneficial for contention window adaptation.

A hidden node for a given node A transmitting to a node B in a wireless network is one which is capable of interfering with the reception of the packet at B, while being unable to sense the transmission of A. This can lead to staggered collisions, when node A's packet is interrupted by the hidden node, causing reception to fail. While this can be dealt with via the RTS/CTS mechanism, practical networks often do not employ RTS/CTS due to the excessive overhead and delay. This can lead to severely degraded throughput in the presence of the hidden terminal problem. In this paper, we propose a contention window adaptation method which is robust to this

type of loss and can improve performance even in the presence of hidden nodes, with no required changes in settings.

Specifically, we propose a distributed algorithm whereby each node uses information from a local and AP busy-idle signal to adapt its contention window size in order to improve overall network utility. In doing so, it is not necessary for individual nodes to know the contention window sizes or throughput of other nodes; rather, the local and AP busy-idle signal contain enough information for each node to estimate the derivative of total network utility with respect to that node's average backoff length. Nodes can then employ a gradient ascent method to reach the optimal operating point for the network. This is done in a timescale on the order of seconds, and can be combined with packet-level adaptations, including the traditional BEB or the methods proposed in [4–7] to adapt to fast timescale changes to network traffic.

In Section II, we derive an expression for total network throughput and show that the derivative of utility with respect to the average backoff length of node i depends only on quantities which can either be observed by node i or are contained in the busy-idle signal of its AP. In Section III, we describe a contention window adaptation algorithm in which each node estimates the derivative of total network utility with respect to its contention window size by exploiting the AP busy-idle signal and locally observable quantities, and adapts its contention window size accordingly. In Section IV we present simulation results showing improvements in utility with significant throughput gains for the three major cases where standard contention window adaptation fails: high node count, high channel error, and hidden nodes.

II. Analytical Framework

In this section we present an expression for total network utility and derive an expression for its derivative with respect to the average backoff length of node i. It is important to maximize utility, rather than individual throughput, because the latter results in the undesirable trivial solution of all nodes setting their CW_{min} to zero. As such, we assume that all nodes seek to maximize this global objective, namely the utility.

To enforce fairness, we consider the utility of each node to be the log of its throughput [9] and maximize the total network utility given by:

$$U = \sum_j \log TP_j. \tag{1}$$

The throughput, TP_i, of node i can be expressed as

$$\begin{aligned} TP_i &= L_i S_i (1 - P_{Li}) \\ &= L_i S_i (1 - P_{SCi})(1 - P_{DCi})(1 - P_{ei}) \end{aligned} \tag{2}$$

where L_i is the number of bits per packet, S_i is the number of packets sent per time slot, i.e. $1/S_i$ is the average number of timeslots that elapse between the beginning of consecutive transmissions of node i, P_{Li} is the packet loss rate for node i, and P_{SCi}, P_{DCi}, and P_{ei} are the probabilities of staggered collision with hidden nodes, direct collision with neighboring nodes, and channel error for node i, respectively.

S_i can be thought of as the probability of node i sending in any given slot. This can be broken down via the chain rule into:

$$\begin{aligned} S_i = P(\text{node } i \text{ sends}) &= P(\text{channel idle in previous slot}) \\ &\times P(\text{node } i \text{ sends}|\text{channel idle in previous slot}) \end{aligned} \tag{3}$$

If we define $\overline{W}_i$ as the average backoff chosen for node i, we have

$$P(\text{node } i \text{ sends}|\text{channel idle in previous slot}) = \frac{1}{\overline{W}_i}. \tag{4}$$

Since the channel alternates between busy and idle states, we also have that

$$P(\text{channel idle in previous slot}) = \frac{T_i}{B_i + T_i}, \tag{5}$$

where B_i and T_i are the average durations of busy and idle periods, respectively.

We can thus express S_i as

$$S_i = \frac{T_i}{B_i + T_i} \frac{1}{\overline{W}_i} \tag{6}$$

Let $\mathcal{N}_i$ be the set of "neighboring" nodes to node i, i.e. the set of nodes whose transmissions can be sensed by node i. Then, assuming that the length of the idle periods is roughly geometric, which is true for Poisson traffic as well as saturated traffic [10], it is clear that

$$\begin{aligned} T_i &= 1/P(\text{any node in } \mathcal{N}_i \cup \{i\} \text{ starts sending in given slot}) \\ &= \left(1 - \prod_{k \in \mathcal{N}_i} \left(1 - \frac{1}{\overline{W}_k}\right)\right)^{-1} \end{aligned} \tag{7}$$

For each node $k \in \mathcal{N}_i$, the probability of collision is the probability that node i completes its backoff at the same time slot as node k, which is $1/\overline{W}_k$. This event occurs independently for each k. Thus, the probability of direct collision can be expressed as

$$P_{DCi} = 1 - \prod_{k \in \mathcal{N}_i} \left(1 - \frac{1}{\overline{W}_k}\right) \tag{8}$$

The staggered collisions for node i are caused by a set of hidden nodes $\mathcal{H}_i$ which can interfere at the AP but cannot be sensed by node i. Although these nodes may share some neighbors with i, for the sake of simplicity, they are assumed to behave independently of node i. The total rate at which the nodes in $\mathcal{H}_i$ send is $\sum_{j \in \mathcal{H}_i} S_j$. Assume they are sent according to a Poisson process, and that all packets are of duration D backoff slots. This assumption is reasonable for the saturated case, where the hidden node problem is most severe, since with saturated traffic, nodes use maximum length packets. In practice, there may also be shorter control packets, which cause the adaptation to be conservative, choosing slightly larger contention windows than necessary. However, we have empirically found that including them in our simulations still results in significant improvements in overall network utility.

Under these assumptions, the probability that a packet sent by node i does not collide with a packet from a node in $\mathcal{H}_i$ is the probability that there is no packet sent within $D-1$ slots before or D slots after. Basic Poisson theory states that this is the probability a Poisson random variable with parameter $(2D-1)\sum_{j\in\mathcal{H}_i} S_j$ is equal to zero, which results in the following expression for P_{SCi}:

$$\begin{aligned} P_{SCi} &= 1 - e^{-(2D-1)\sum_{j\in\mathcal{H}_i} S_j} \\ &= 1 - \prod_{j\in\mathcal{H}_i} e^{-(2D-1)S_j} \end{aligned} \tag{9}$$

Substituting Equations (6), (8), and (9) into Equation (2) yields:

$$TP_i = \frac{L_i T_i (1-P_{ei})}{(B_i+T_i)\overline{W}_i} \prod_{k\in\mathcal{N}_i} \left(1 - \frac{1}{\overline{W}_k}\right) \prod_{j\in\mathcal{H}_i} e^{-(2D-1)S_k} \tag{10}$$

In order to maximize the utility in a distributed manner, each node i must be able to estimate the derivative of Equation (1) with respect to $\overline{W}_i$ using only locally available information.

This derivative can be broken up into 3 terms: one related to the node itself, one to its neighbors, and one to its hidden nodes:

$$\begin{aligned} \frac{\partial}{\partial \overline{W}_i} U =& \frac{\partial}{\partial \overline{W}_i} \log TP_i + \sum_{k\in\mathcal{N}_i} \frac{\partial}{\partial \overline{W}_i} \log TP_k \\ &+ \sum_{j\in\mathcal{H}_i} \frac{\partial}{\partial \overline{W}_i} \log TP_j \end{aligned} \tag{11}$$

We evaluate each of these terms in Appendix A to obtain:

$$\begin{aligned} \frac{\partial}{\partial \overline{W}_i} U =& -\frac{1}{\overline{W}_i} + \left(\frac{1}{T_i} - \frac{1}{B_i+T_i}\right)(1-P_{DCi})\frac{T_i^2}{\overline{W}_i^2} \\ &+ |\mathcal{N}_i| \left[\frac{B_i}{T_i(B_i+T_i)}(1-P_{DCi})\frac{T_i^2}{\overline{W}_i^2} - \frac{1}{\overline{W}_i^2 - \overline{W}_i}\right] \\ &+ |\mathcal{H}_i| \frac{(2D-1)T_i}{(B_i+T_i)\overline{W}_i^2} \end{aligned} \tag{12}$$

We use this equation in Section III to derive a contention window adaptation algorithm.

III. Adaptation Algorithm

We assume an 802.11 network in infrastructure mode, potentially with multiple APs. APs periodically broadcast the BI signal to all associated nodes, which the nodes can use to estimate their probabilities of each type of loss. Additionally, from this same information, nodes can estimate several quantities necessary for computation of the derivative in Equation (12).

Notice that all the quantities on the right-hand side of Equation (12) can be observed or estimated by node i. Specifically, $\overline{W}_i$ can be observed by recording the chosen backoffs; B_i and T_i can be observed from the busy-idle signal; P_{DCi} can be estimated from the local and AP busy-idle signal; $|\mathcal{N}_i|$ can be observed from headers of overheard packets; and $|\mathcal{H}_i|$ can be obtained if the AP additionally broadcasts a list of the nodes it hears or it can be estimated as in [8].

By estimating the derivative of network utility with respect to their average backoff, nodes can tune their contention window in order to improve network utility. We begin by examining the case where $\alpha = 1$, so that it is sufficient to find the optimal $\overline{W}_i$, then we extend to other values of α, where it is necessary to find both $\overline{W}_i$ and CW_{min}.

There are two ways to exploit the derivative in Equation (12). The first is to use a gradient ascent method, whereby each node changes its contention window size by an amount proportional to its derivative. We call this the "gradient" approach. The second is to set the derivative equal to zero, and solve the resulting quadratic equation. Nodes can then immediately change their contention window size to this value. We call this the "zero" approach.

The zero approach cannot find the actual zero of the derivative when the current average contention window size, W_0, is far from the optimal $\overline{W}_i$ because the observed quantity T_i depends on the current contention window size. More precisely, if we use the notation $T_i(\overline{W}_i)$ to denote the dependency of T_i on $\overline{W}_i$, the observed quantity is $T_i(W_0)$. An example can be seen in Figure 1 where the solid blue curve plots $\frac{\partial}{\partial \overline{W}_i} U(T_i(\overline{W}_i), \overline{W}_i)$, and the dashed red curve plots $\frac{\partial}{\partial \overline{W}_i} U(T_i(W_0), \overline{W}_i)$. As seen, at $\overline{W}_i = W_0$, these are equal; however, for $\overline{W}_i > W_0$, $|\mathcal{N}_i| \geq 1$, and $W_0 \geq 1$, Equation (12) implies that $T_i(\overline{W}_i) > T_i(W_0)$. Therefore, at the zero of the dashed curve, namely $\overline{W}_i = W_0^*$, the solid curve is still positive. The zero of the solid curve must therefore occur at $\overline{W}_i^*$ such that $|\overline{W}_i^* - W_0| > |W_0^* - W_0|$. Thus if the algorithm were to jump to the estimated zero, W_0, it would systematically undershoot the actual optimal and converge slowly. To increase the speed of the convergence, it is useful to slightly overshoot the target W_0^*. We have empirically found via NS-2 simulations that an overshoot of 25% works well.

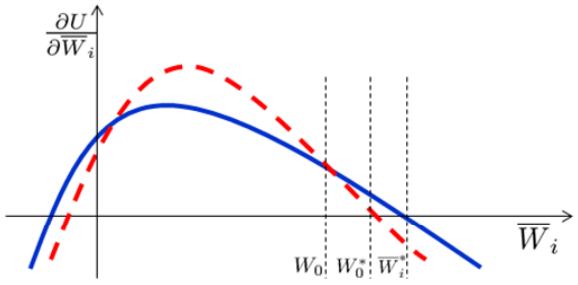

Fig. 1. *An example plot of* $\frac{\partial}{\partial \overline{W}_i} U(T_i(\overline{W}_i), \overline{W}_i)$ *as a function of* $\overline{W}_i$*, represented by the solid blue curve, and* $\frac{\partial}{\partial \overline{W}_i} U(T_i(W_0), \overline{W}_i)$ *as a function of* $\overline{W}_i$*, represented by the dashed red curve.*

The gradient approach is guaranteed to converge to the optimal solution assuming that all the relevant quantities are estimated accurately, but the convergence can be slow. The zero approach comes with no such guarantees, and in fact it is possible for the function $\frac{\partial}{\partial \overline{W}_i} U(T_i(\overline{W}_i), \overline{W}_i)$ computed in a given step

to be convex, rather than concave, for very small values of T_i, potentially causing $\overline{W}_i$ to change by a large amount in the wrong direction. However, by using a combination of the current derivative estimate along with the approximated zero, it is possible to assure convergence at a faster rate than using the gradient approach alone. Specifically, our proposed approach is the use the zero method when its direction of the change is in agreement with the derivative. Otherwise, the change in contention window size is adjusted by an amount proportional to the value of the derivative. The complete algorithm, for each node i, is shown in Algorithm 1. The values in step (9) are found via empirical tuning within NS-2.

Algorithm 1 *The CW adaptation algorithm*

1) Transmit at current rate for 5 seconds and observe:
   - $\overline{W}_i$
   - $|\mathcal{N}_i|$
   - busy-idle signal
2) Receive busy-idle signal from AP
3) Estimate P_{DCi} and $|\mathcal{H}_i|$ using BI signal
4) Estimate $\texttt{dUdW} = \frac{\partial}{\partial \overline{W}_i} U$ via Equation (12)
5) Solve for $\texttt{target\_W} = \{\overline{W}_i : \frac{\partial}{\partial L} TP = 0\}$
6) If $\alpha \neq 1$, compute $\texttt{target\_Wo}$ via Equation (18)
7) If $(sign(\texttt{target\_Wo} - \texttt{Wo}) = sign(\texttt{dUdW}))$
   - $\texttt{Wo} = \texttt{Wo} + 0.25(\texttt{target\_Wo} - \texttt{Wo})$

   Else
   - $\texttt{Wo} = \texttt{Wo} + 0.25(\texttt{target\_Wo} - \texttt{Wo}) + c \cdot \texttt{dUdW}$
8) Truncate changes of over 50%
9) Update c as follows:
   - If change is in same direction as last step increase c by 25%
   - Else decrease c by 50%

In the situation where $\alpha = 1$, it is sufficient to set $CW_{min,i} = 2\overline{W}_i$, but for $\alpha \neq 1$ or in the case of any Markovian packet-level adaptation, it is not trivial to select the appropriate $CW_{min,i}$ from the optimal $\overline{W}_i$. For a given backoff procedure, it is possible to derive an expression for the relationship between CW_{min} and $\overline{W}_i$, but this expression typically depends on the probability of packet loss, P_L. Fortunately, an estimate of P_L can be obtained from an empirical count assuming relatively stationary traffic.

We now derive the expression for $CW_{min,i}$ for the standard backoff procedure. To do so, we can consider a Markov chain with M states, where M is the retransmit limit. The state transition probabilities are:

$$p_{m,m+1} = P_L \qquad \text{for } m = 0, ..., M-1 \tag{13}$$

$$p_{m,0} = 1 - P_L \qquad \text{for } m = 0, ..., M-1 \tag{14}$$

$$p_{M,0} = 1 \tag{15}$$

All other transitions have zero probability. From steady-state equations, we can solve for the steady-state proportion of backoffs chosen from each state, π_m. The average value of the chosen backoff in state m is $\frac{1}{2}\min(CW_{min,i} \cdot \alpha^i, CW_{max,i})$. Thus the average contention window size can be expressed as

$$\overline{W}_i = \sum_{i=0}^{M} \pi_i \frac{1}{2} \min(CW_{min,i} \cdot \alpha^i, CW_{max}) \tag{16}$$

$$= \frac{1}{2} CW_{min} \sum_{i=0}^{M} \pi_i \min\left(\alpha^i, \frac{CW_{max}}{CW_{min}}\right) \tag{17}$$

If we define $m = CW_{max}/CW_{min,i}$ as a constant, then there is a linear relationship between $\overline{W}_i$ and $CW_{min,i}$:

$$CW_{min,i} = \frac{2\overline{W}_i}{\sum_{i=0}^{M} \pi_i \min(\alpha^i, m)} \tag{18}$$

In order to select the optimal $CW_{min,i}$, node i estimates the optimal $\overline{W}_i$ and observes P_L. It then uses P_L to solve for π_m and uses these values along with $\overline{W}_i$ to select $CW_{min,i}$ via Equation (18).

IV. Simulation Results

To characterize the throughput gains for our contention window adaptation algorithm, we use the NS-2 simulation package. We have made modifications to allow for collection of the BI signal, estimation of collision probabilities, and execution of the the adaptation algorithm. Simulations are carried out for 802.11b, but the results are generally extensible to any MAC which uses carrier-sense multiple access.

Figure 2 shows the relationship between throughput and contention window for various channel error rates. The topology consists of two nodes and a single AP in a single collision domain with no hidden nodes. Since the topology is effectively symmetric, increasing utility is equivalent to increasing throughput, and thus it is sufficient to examine throughput. The nodes send saturated traffic, and use the fixed contention window size indicated on the x-axis. The process is repeated for 10 different contention window sizes and 3 different channel error probabilities. Throughput is plotted as a function of contention window size as the solid lines, with the upper blue curve corresponding to $P_{ei} = 0$, the middle red curve corresponding to $P_{ei} = 0.3$, and the lower green curve corresponding to $P_{ei} = 0.6$. It can be seen that the optimal contention window size is less than $CW_{min} = 32$ in all cases.

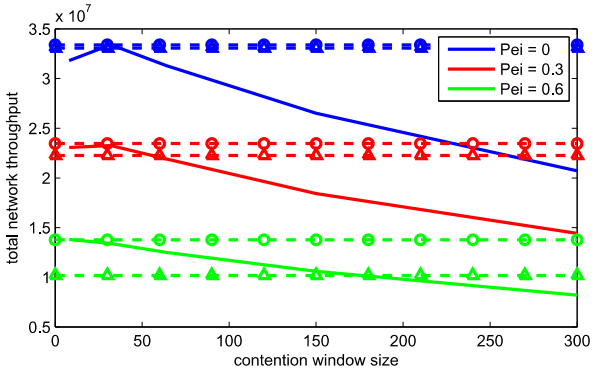

Fig. 2. *Throughput as a function of contention window, for 2 nodes.*

The throughput achieved by the standard BEB algorithm is shown as the horizontal dashed lines with triangles. It is close to optimal for $P_{ei} = 0$, but suffers as P_{ei} increases. This is because the standard BEB algorithm assumes all losses are due to collisions, thus causing nodes to misinterpret the losses

caused by channel error as losses caused by high channel contention. The throughput achieved by our proposed algorithm is shown as the horizontal dashed lines with circles. It can be seen that it achieves near-optimal throughput regardless of P_{ei}, achieving a 35% throughput improvement for $P_{ei} = 0.6$.

While the standard BEB protocol works reasonably well for small numbers of nodes in the absence of channel errors or hidden nodes, performance drops off rapidly as the number of nodes increases, since for a large number of nodes $CW_{min} = 32$ is small enough that the collision probability of the first attempt is too large. This drop in performance can be seen in Figure 3, which shows throughput as a function of the number of nodes. The blue curve is for the standard BEB algorithm, and the red and green are for the adaptive algorithm with $\alpha = 1$ and $\alpha = 2$, respectively. While the throughput for all methods decreases with number of nodes, the slope is significantly smaller for the adaptive algorithms, resulting in 14% improved throughput for 20 nodes and 24% for 40 nodes as compared to the standard.

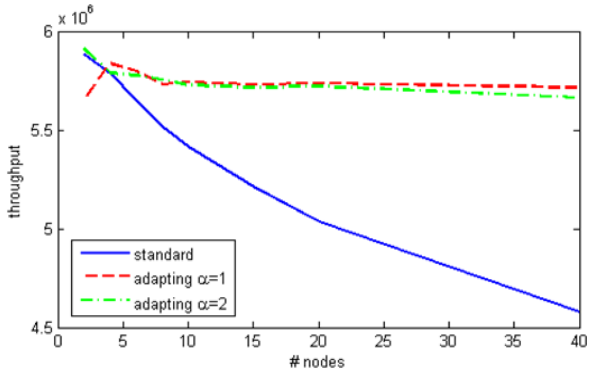

Fig. 3. *Throughput vs number of nodes for standard and adaptive with $\alpha = 1$ and $\alpha = 2$.*

Figure 4 plots CW_{min} as a function of time for one node in the same scenario as Figure 3 for 5, 10, and 15 nodes. It can be seen that the convergence time increases with the number of nodes, taking about 40 seconds, or 8 iterations, for 10 nodes.

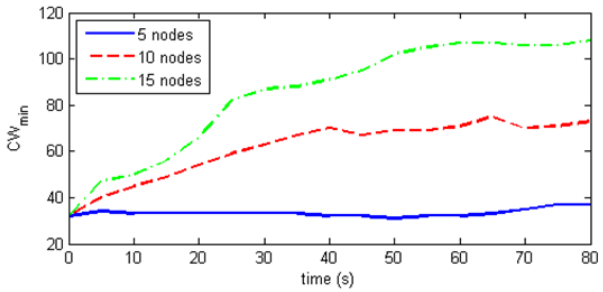

Fig. 4. *CW_{min} vs time for single AP with no hidden nodes and 5-15 nodes.*

Figure 5 shows the total throughput for a scenario with 20 nodes sharing a single access point with no hidden nodes and various frame error rates (FERs) for standard BEB and adaptation for various values of α. It can be seen that the value of α does not make a significant difference in the throughput performance. For $FER = 0$, regardless of α, the adaptive algorithm outperforms the standard by 14%. As the FER increases, there is less to gain via adaptation.

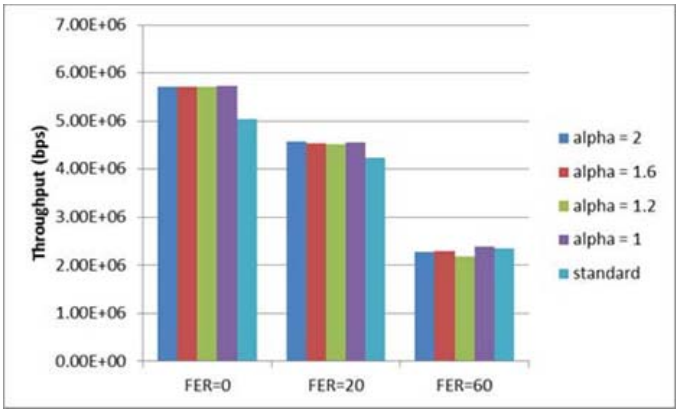

Fig. 5. *Throughput vs standard for various FER and alpha for 20 nodes with no hidden nodes.*

Figure 6 shows a scenario with 20 nodes associated with a single AP arranged uniformly in a large circle with a large radius such that all nodes have the same non-zero number of neighboring and hidden nodes. As seen, even for large values of P_{ei}, there is an improvement of as much as 350% compared to standard. This improvement decreases with P_{ei}, but increases with α. The decrease with P_{ei} is due to the fact that as P_{ei} increases, the standard algorithm increases the contention window not because it recognizes the hidden node problem, but because it assumes there is a greater number of collisions which improves the behavior despite the incorrect reasoning. The increase with α is due a phenomenon which causes bursts of packets to be sent when hidden nodes choose long backoffs. With large values of α, a given $\overline{W}_i$ results in a smaller CW_{min}. Under these conditions, when a transmission by node i is successful, suggesting that nodes in $\mathcal{H}_i$ are currently in a long backoff, node i is more likely to transmit again immediately, allowing better utilization of the time during long backoffs of hidden nodes.

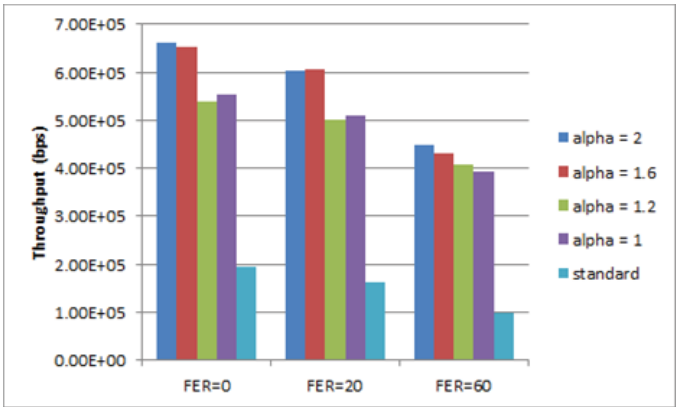

Fig. 6. *Throughput vs standard for various FER and alpha for 20 nodes with hidden nodes.*

In a multi-access point network with asymmetric hidden node conditions, fairness becomes an issue. Nodes which do not suffer from the hidden node problem can dominate channel

access and starve out nodes with less favorable conditions. Figure 7(a) show a histogram of the throughput of 50 nodes placed randomly over an area covered by 7 APs with hexagonal cells. It can be seen that in this scenario, 31 of the 50 nodes have throughput under 20 kbps. However, when adaptation is applied with $\alpha = 1$ or $\alpha = 2$, as shown in Figures 7(b) and 7(c), respectively, these nodes have improved throughput, resulting in much greater fairness. Total throughput is decreased, but overall utility is improved.

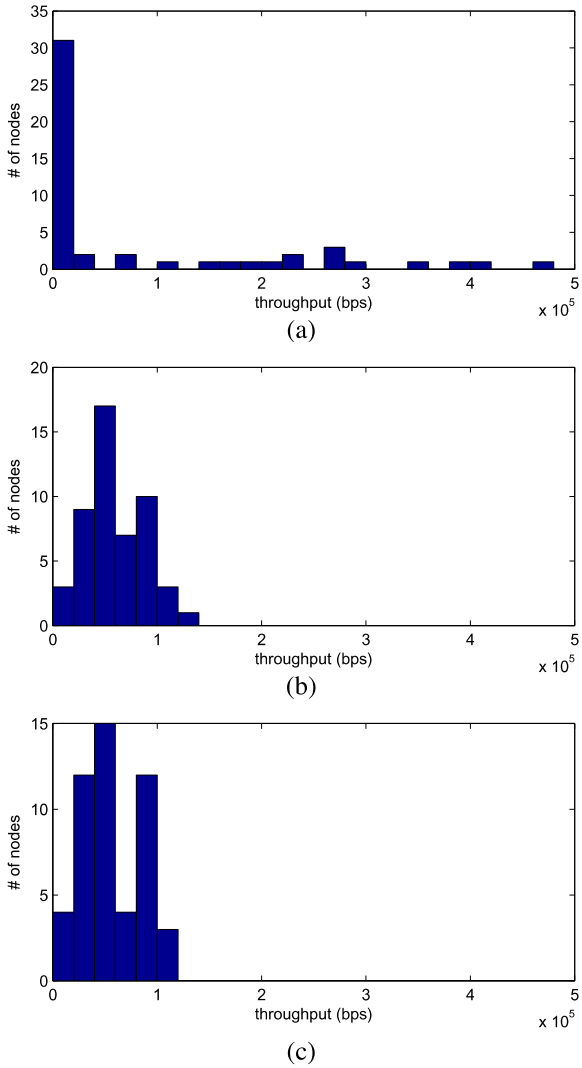

Fig. 7. *Histogram of throughputs of each of 50 nodes for (a) standard BEB, (b) adaptation with $\alpha = 1$, and (c) adaptation with $\alpha = 2$.*

To quantify the improvement in fairness and utility, we introduce 2 metrics. The first is Jain's fairness index [12], which quantifies the fairness on a scale of $\frac{1}{n}$ to 1:

$$J = \frac{(\sum_i TP_i)^2}{n \cdot \sum_i TP_i^2} \tag{19}$$

The second we call the "equivalent equal throughput":

$$eeTP = e^{\frac{1}{n}\sum_i log(TP_i)} \tag{20}$$

This expresses the average throughput n nodes with equal throughput would have to achieve to obtain the given utility, $U = \sum_i log(TP_i)$. This quantity is easier to compare than the actual utility, since it has units of bits/sec.

We ran 15 simulations on various random topologies with 7 APs 40-60 nodes and found the average throughput decreased by 44%, but the average utility increased by 7%, corresponding to an equivalent equal throughput gain of 165% and a 52% improvement in the Jain fairness index.

V. Conclusions and Future Work

In this paper, we proposed a novel contention window adaptation scheme in which nodes use information shared by the AP to optimize contention window sizes in a distributed fashion in order to improve network utility. We have examined three cases in which the standard BEB algorithm performs poorly and shown via NS-2 simulations that our method can improve throughput by as much as 24% in the high traffic scenario, 35% in the high channel error scenario and 350% in the presence of the hidden terminal problem.

Our results indicate that performance can vary depending on the chosen value of α. Determining the optimal alpha remains an open problem. Additionally, it may be the case that exponential backoff is inferior to other per-packet adaptation approaches such as those in [4–7]. Since our algorithm works at a slower time scale, it can potentially be combined with these other methods to further improve performance. Additionally, the adaptation of contention window is the natural complement of the adaptation of modulation rate presented in [11]. A natural extension of this work would be to jointly apply these adaptations.

References

[1] Y. Xu, M. Huang, M. Lin, Y. Zheng, "A Self-adaptive Minimum Contention Window Adjusting Backoff Algorithm in IEEE 802.11 DCF", in *Proc. of CECNet 2012*, YiChang, China, April 2012.

[2] C. Weng, C. Chen, "Performance Study of IEEE 802.11 DCF with Optimal Contention Window", in *Proc. of IMIS 2012*, Palermo, Italy, July 2012.

[3] S. Chun, D. Xianhua, L. Pingyuan, Z. Han, "Adaptive Access Mechanism with Optimal Contention Window Based on Node Number Estimation Using Multiple Thresholds", in *IEEE Transactions on Wireless Communications*, Vol. 11, No. 6, June 2012

[4] N. Song, B. Kwak, J. Song, L. Miller, "Analysis of EIED backoff algorithm for the IEEE 802.11 DCF", in *Proc. of IEEE VTC 2005 Fall*, Dallas, Texas, September 2005.

[5] Y. Yang, S. Lam, "General AIMD Congestion Control". in *Proc. of IEEE ICNP 2000*, Osaka, Japan, November 2000.

[6] V. Bharghavan, A. Demers, S. Shenker, L. Zhang, "MACAW: A Media Access Protocol for Wireless LANs", in *Proc. of ACM SIGCOMM 1994*, London, UK, September 1994.

[7] A. Lukyanenko, A. Gurtov, E. Morozov, "An Adaptive Backoff Protocol with Markovian Contention Window Control", in *Communications in Statistics - Simulation and Computation, vol. 41*, 2012.

[8] Michael N. Krishnan, Sofie Pollin, and Avideh Zakhor, "Local Estimation of Probabilities of Direct and Staggered Collisions in 802.11 WLANs", in *Proc. of IEEE GLOBECOM 2009*, Honolulu, Hawaii, December 2009.

[9] F. Kelly, A. Maulloo, and D. Tan, "Rate Control In Communication Networks: Shadow Prices, Proportional Fairness and Stability", in *Journal of the Operational Research Society*, vol. 49, 1998.

[10] Guiseppe Bianchi, "Performance Analysis of the 802.11 Distributed Coordination Function", *IEEE JSAC*, Vol. 18, No. 3, March 2000, pp.535-547.

[11] M. Krishnan and A. Zakhor, "Throughput Improvement in 802.11 WLANs using Collision Probability Estimates in Link Adaptation", in *Proc. of IEEE WCNC 2010*, Sydney Australia, April 2010.

[12] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer system," DEC Technical Report 301, 1984.

APPENDIX

The derivative of utility with respect to $\overline{W}_i$ can be broken up into 3 terms: one relating to the node itself, one to its neighbors, and one to its hidden nodes.

$$\frac{\partial}{\partial \overline{W}_i} U = \frac{\partial}{\partial \overline{W}_i} \log TP_i + \sum_{k \in \mathcal{N}_i} \frac{\partial}{\partial \overline{W}_i} \log TP_k + \sum_{j \in \mathcal{H}_i} \frac{\partial}{\partial \overline{W}_i} \log TP_j \tag{21}$$

We now compute each of these terms.

The first is straightforward.

$$\frac{\partial}{\partial \overline{W}_i} \log TP_i = \frac{1}{TP_i} R_i (1 - P_{Li}) \frac{\partial}{\partial \overline{W}_i} S_i \tag{22}$$

$$= \left(\frac{T_i}{B_i + T_i} \frac{1}{\overline{W}_i} \right)^{-1} \frac{\partial}{\partial \overline{W}_i} \left(\frac{T_i}{B_i + T_i} \frac{1}{\overline{W}_i} \right) \tag{23}$$

$$= -\frac{1}{\overline{W}_i} + \left(\frac{1}{T_i} - \frac{1}{B_i + T_i} \right) \frac{\partial}{\partial \overline{W}_i} T_i \tag{24}$$

where $\frac{\partial}{\partial \overline{W}_i} T_i$ can be computed as:

$$\frac{\partial}{\partial \overline{W}_i} T_i = \frac{\partial}{\partial \overline{W}_i} \left(1 - \prod_{k \in \mathcal{N}_i} \left(1 - \frac{1}{\overline{W}_k} \right) \right)^{-1} \tag{25}$$

$$= \frac{\partial}{\partial \overline{W}_i} \frac{1}{1 - (1 - P_{DCi})(1 - \frac{1}{\overline{W}_i})} \tag{26}$$

$$= (1 - P_{DCi}) T_i^2 / \overline{W}_i^2 \tag{27}$$

The second term in (21) is:

$$\frac{\partial}{\partial \overline{W}_i} \sum_{k \in \mathcal{N}_i} \log TP_k$$

$$= \sum_{k \in \mathcal{N}_i} \frac{1}{TP_k} \frac{R_k (1 - P_{ei})}{\overline{W}_k} \frac{\partial}{\partial \overline{W}_i} \left(\frac{T_k}{B_i + T_k} (1 - P_{DCk}) \right) \tag{28}$$

$$= \sum_{k \in \mathcal{N}_i} \left[\frac{B_i + T_k}{T_k} \frac{\partial}{\partial \overline{W}_i} \left(\frac{T_k}{B_i + T_k} \right) - \frac{\frac{\partial}{\partial \overline{W}_i} P_{DCk}}{1 - P_{DCk}} \right] \tag{29}$$

$$= \sum_{k \in \mathcal{N}_i} \left[\frac{B_i}{T_k (B_i + T_k)} \frac{\partial}{\partial \overline{W}_i} T_k - \frac{\frac{\partial}{\partial \overline{W}_i} P_{DCk}}{1 - P_{DCk}} \right] \tag{30}$$

$\frac{\partial}{\partial \overline{W}_i} P_{DCk}$ can be computed as:

$$\frac{\partial}{\partial \overline{W}_i} P_{DCk} = \frac{\partial}{\partial \overline{W}_i} \left[1 - \prod_{j \in \mathcal{N}_k} \left(1 - \frac{1}{\overline{W}_j} \right) \right] \tag{31}$$

$$= - \left[\prod_{j \in \mathcal{N}_k \setminus i} \left(1 - \frac{1}{\overline{W}_j} \right) \right] \frac{\partial}{\partial \overline{W}_i} \left(1 - \frac{1}{\overline{W}_i} \right) \tag{32}$$

$$= - \frac{1}{\overline{W}_i^2} \prod_{j \in \mathcal{N}_k \setminus i} \left(1 - \frac{1}{\overline{W}_j} \right) \tag{33}$$

$$= - \frac{1 - P_{DCk}}{1 - \frac{1}{\overline{W}_i}} \frac{1}{\overline{W}_i^2} \tag{34}$$

Substituting this in to Equation (30) yields

$$\frac{\partial}{\partial \overline{W}_i} \sum_{k \in \mathcal{N}_i} \log TP_k = \sum_{k \in \mathcal{N}_i} \left[\frac{B_i}{T_k (B_i + T_k)} \frac{\partial}{\partial \overline{W}_i} T_k - \frac{1}{\overline{W}_i^2 - \overline{W}_i} \right] \tag{35}$$

If we assume $T_i \approx T_k$, which is reasonable since the nodes are neighbors, we get

$$\frac{\partial}{\partial \overline{W}_i} \sum_{k \in \mathcal{N}_i} \log TP_k = \sum_{k \in \mathcal{N}_i} \left[\frac{B_i}{T_i (B_i + T_i)} \frac{\partial}{\partial \overline{W}_i} T_i - \frac{1}{\overline{W}_i^2 - \overline{W}_i} \right] = |\mathcal{N}_i| \left[\frac{B_i}{T_i (B_i + T_i)} (1 - P_{DCi}) \frac{T_i^2}{\overline{W}_i^2} - \frac{1}{\overline{W}_i^2 - \overline{W}_i} \right] \tag{36}$$

The third term in (21) is:

$$\frac{\partial}{\partial \overline{W}_i} \sum_{j \in \mathcal{H}_i} \log TP_k$$

$$= \sum_{j \in \mathcal{H}_i} \frac{1}{TP_j} \frac{TP_j}{1 - P_{SCj}} \frac{\partial}{\partial \overline{W}_i} (1 - P_{SCj}) \tag{37}$$

$$= \sum_{j \in \mathcal{H}_i} \frac{1}{1 - P_{SCj}} \frac{(1 - P_{SCj})}{e^{-(2D-1)S_i}} \frac{\partial}{\partial \overline{W}_i} e^{-(2D-1)S_i} \tag{38}$$

$$= \sum_{j \in \mathcal{H}_i} (-(2D-1)) \frac{\partial}{\partial \overline{W}_i} S_i \tag{39}$$

$$= \sum_{j \in \mathcal{H}_i} \frac{(2D-1) T_i}{(B_i + T_i) \overline{W}_i^2} \tag{40}$$

$$= |\mathcal{H}_i| \frac{(2D-1) T_i}{(B_i + T_i) \overline{W}_i^2} \tag{41}$$

Substituting Equations (24), (36) and (41) into (21) yields the complete expression for $\frac{\partial}{\partial \overline{W}_i} U$.

$$\frac{\partial}{\partial \overline{W}_i} U = - \frac{1}{\overline{W}_i} + \left(\frac{1}{T_i} - \frac{1}{B_i + T_i} \right) (1 - P_{DCi}) \frac{T_i^2}{\overline{W}_i^2} + |\mathcal{N}_i| \left[\frac{B_i}{T_i (B_i + T_i)} (1 - P_{DCi}) \frac{T_i^2}{\overline{W}_i^2} - \frac{1}{\overline{W}_i^2 - \overline{W}_i} \right] + |\mathcal{H}_i| \frac{(2D-1) T_i}{(B_i + T_i) \overline{W}_i^2} \tag{42}$$