cMAC: A Centralized MAC Protocol for High Speed Wireless LANs

Shuaiyuan Zhou and Zhenghao Zhang
Computer Science Department
Florida State University
Tallahassee, FL 32306, USA

***Abstract*—We propose cMAC, a centralized, polling-based protocol for wireless LANs. With cMAC, the access point (AP) sets nodes into the polling mode if they have backlogged data and polls them during the polling period. The AP also periodically opens the contention period to allow nodes with newly arrived data to send data and at the same time announce their queue states. The main feature of cMAC is that it aggressively sets nodes into the polling mode to reduce collision and overhead. We design an algorithm to determine the length of the contention period according to the maximum likelihood estimation of the number of competing nodes. We also show that cMAC is robust against message losses. We test cMAC using ns2 simulator and the results show (1) cMAC achieves significant gain , e.g., 56% with 50 nodes, over 802.11 DCF in throughput with lower average packet delay; (2) cMAC can handle uplink and downlink traffic simultaneously and (3) cMAC is backward compatible with legacy 802.11 nodes.**

I. INTRODUCTION

Wireless LANs offer convenient access to the Internet. The 802.11 protocol, i.e., Wi-Fi, has been widely deployed. One of the key issues in wireless LAN is the Medium Access Control (MAC) protocol, which determines the rules of transmission for nodes to avoid collision. The 802.11 DCF adopts a simple, randomized strategy: when the medium is free, each node uses a random number as its backoff counter and the one with the smallest number transmits first. The random backoff process introduces overhead and cannot completely remove collision.

In this paper, we take a drastically different approach to design a more efficient MAC protocol for wireless LANs. We note that in many cases, the wireless LAN consists of a single Access Point (AP) and nodes associated with it. The AP can naturally act as the central controller of the network and a centralized protocol can be applied which should yield better performance than randomized protocols because the central controller can enforce a collision-free transmission schedule at minimum overhead. In light of this, we propose *cMAC*, a centralized, polling-based protocol. With cMAC, the AP keeps track of nodes with non-empty queues, and polls such nodes round-robin in the *polling period*. Because the AP only polls nodes with non-empty queues, every poll is guaranteed a hit. The AP also periodically opens the *contention period* during which nodes with newly arrived data can compete for medium access to transmit data and announce their queue states. The main feature of cMAC is that it aggressively sets nodes in the polling mode: once a node with non-empty queues is known to the AP, it basically does not compete for medium access during the contention period, which minimizes the number of competing nodes and the probability of collision. The polling message from the AP can be piggybacked with the ACK packet; also, the queue state of a node can be piggybacked with the data packet. Therefore, the polling protocol incurs virtually no additional overhead.

The design of cMAC faces the following challenges. First, the AP cannot stay in the polling mode forever and must periodically break from polling to allow nodes to transmit if they are currently not in the polling list but have new data. It is straightforward to determine the length of the polling period, i.e., after every node has been polled or after a timeout; however, it is challenging to determine the length of the contention period because the AP does not know the number of active nodes not in the polling list. If the contention period is too short, a node may incur long delay, i.e., have to wait for several contention periods before transmits; if too long, the efficiency of the protocol obviously reduces. To this end, we design a method which estimates the number of active nodes based on observation of the number of idle time slots according to the maximum likelihood estimation. Second, the protocol should be robust against message losses. We rigorously argue that cMAC can handle all types of message losses. We tested the cMAC with the ns2 simulator [12], and the results show that cMAC achieves significantly better performance than 802.11 DCF in throughput with lower average packet delay. The simulation also shows that cMAC is backward compatible with legacy 802.11 nodes.

The rest of the paper is organized as follows. Section II describes the cMAC protocol. Section III evaluates the cMAC protocol. Section IV discusses related works. Section V concludes this paper.

II. THE cMAC PROTOCOL

In this section, we discuss the cMAC protocol in details.

A. *The cMAC protocol*

In the cMAC protocol, the AP maintains a *polling list* which is the list of the nodes with non-empty queues. The AP has three modes: the *uplink contention mode*, the *uplink polling mode*, and the *downlink transmission mode*. A node is an *idle node* if it has no data to send; otherwise an *active node*. An active node has two modes, the *polling mode* and the *contention mode*, and is called a polling node or a contention node, respectively.

The AP's modes are explained in the following.

The uplink contention mode: When powers up, the AP enters this mode and waits for uplink transmission. When gets an uplink packet, if the node has a non-empty queue, the AP adds this node to its polling list and sets a field in the ACK which will set this node in the polling mode. The AP monitors the number of idle time slots elapsed since the medium was first free till the time the medium is busy again. If this number is less than a threshold, it stays in the contention mode. Otherwise it leaves for the polling mode if it has nodes to poll, or the downlink transmission mode if it has no node to poll but has downlink data to send, or stays in the contention mode if it has neither nodes to poll nor data to send.

The uplink polling mode: In this mode, the AP polls the nodes in the polling list round-robin. The polling message is piggybacked with the ACK for the last correctly received packet. The duration in the ACK is set long enough to cover the polling of the next packet. If all nodes in the list have been polled once, it leaves this mode. If the polling period is longer than a threshold, it also leaves this mode. When leaving this mode, the AP will first enter the downlink transmission mode if it has downlink packets; otherwise it leaves for the uplink contention mode.

The downlink transmission mode: In this mode, the AP transmits downlink packets back to back, waits only DIFS in between. If a packet is sent but ACK not received, it retransmits the packet until the maximum number of retransmission is reached. The amount of time the AP stays in this mode is determined by the desired uplink/downlink ratio. It will then leave for the uplink contention mode.

The node's modes are explained in the following.

The contention mode: When powers up, a node enters this mode. If it gets a packet from the application, it follows the 802.11 DCF to compete for channel access. A node always piggybacks its queue state with the data packet it sends. When a received ACK indicates leaving the contention mode, a node will leave for the polling mode.

The polling mode: In this mode, a node waits for polling messages from the AP by checking certain header fields in the ACK. If polled, it will send a packet without backing off for any time slot. It will not retransmit if no ACK is received for this packet. Whenever a node hears an ACK from the AP, it will generate a random number from the initial contention window, then add it with half of the initial contention window, as the backoff counter. If the backoff counter reaches 0, it will send a packet, and will retransmit according to 802.11 DCF if no ACK is received. For the last packet in the queue, if ACK is received, it will leave for the contention mode immediately; otherwise, it will leave for the contention mode when receives new data from the application.

B. *Determining the Length of the Contention Period*

As mentioned earlier, the core question of the protocol is to determine whether to continue the contention in the uplink contention mode. This decision clearly should be based on the number of contention nodes and the ideal length of the contention period should allow each of such nodes to transmit once. The challenge is that the AP does not know the number of contention nodes. IdleSense [11] pointed out an interesting observation that the number of idle time slots between consecutive transmissions is a good indicator of the number of active nodes in the network: with more active nodes, the number of idle time slots is likely to be smaller because it is the smallest number chosen among the nodes. With IdleSense, nodes observe the number of idle time slots and adjust their contention window size accordingly to approach the desired number of idle slots. cMAC also relies on the observation of the number of idle time slots to determine the length of the contention period. However, our solution is different from IdleSense because the objectives are different: IdleSense is a distributed protocol runs at every node to determine the optimal contention window size while our algorithm runs at the AP to determine whether to continue the contention. Consequently, the methods are different: IdleSense uses AIMD to adjust the contention window size, while our algorithm makes the decision based on the *maximum likelihood estimation* of the number of contention nodes. Basically, we stop the contention if the estimate indicates that there is no contention node.

We say the contention period consists of one or multiple *rounds* of transmission, where each round consists of the idle time slots followed by a transmission. At the end of each round, the AP can determine to continue contention, or to leave for other modes. The first round is round 1 and for any round i, we use X_i to denote the number of idle time slots and N_i the number of contention nodes at the beginning of this contention round. Both X_i and N_i are random variables. We denote the size of the initial contention window as W. Typically, W is an even number, therefore for simplicity, W is assumed to be even in this paper but our result also applies to odd W as well with minor modifications. Let $W' = W - \sum_{j=1}^{i-1} x_j$ and call it the *effective contention window size* of round i. We basically have to determine whether to continue contention based on the observed values of $\{X_1, X_2, \ldots, X_i\}$. Our rule is somewhat surprisingly simple: *continue contention if* $X_i < \frac{W'}{2}$. Because of its simplicity, this rule is very easy to implement which improves the robustness of the protocol.

We explain the derivation of this rule in the following. As the exact analysis is intractable, we make the following two simplifying assumptions in our derivation: (1) the contention nodes pick the random backoff counter at the beginning of the contention period and (2) there is no collision in the contention period. Our simulations show that the rules derived based on these assumptions can achieve desirable performance. We begin with the first round.

1) *The First Round:* In statistics, $P(X_1 = x|N_1 = n)$ is the *likelihood* of having n contention nodes if there are x idle time slots. The maximum likelihood estimation of N_1 is a value n that maximizes the likelihood. Therefore, for any given observation x, if

$$P(X_1 = x|N_1 = 1) < P(X_1 = x|N_1 = n)$$

for some $n > 1$, the contention should continue. We show that

Theorem 1: If $x < \frac{W}{2}$, there exists n such that $P(X_1 = x|N_1 = 1) < P(X_1 = x|N_1 = n)$. If $X \geq \frac{W}{2}$, then $P(X_1 = x|N_1 = 1) > P(X_1 = x|N_1 = n)$ for any $n > 1$.

Proof: We note that the probability that a single node picked a backoff counter no less than x is clearly $1 - \frac{x}{W}$, where $0 \leq x \leq W - 1$. As we assume all nodes pick the backoff counter at the beginning of the contention period, the probability that at least one node picks a backoff counter less than x, when there are n contention nodes, is $1 - (1 - \frac{x}{W})^n$, and thus the probability that the minimum number picked by n nodes is exactly x is

$$\begin{aligned} P(X = x|N = n) &= [1 - (1 - \frac{x+1}{W})^n] - [1 - (1 - \frac{x}{W})^n] \\ &= (1 - \frac{x}{W})^n - (1 - \frac{x+1}{W})^n \end{aligned}$$

To see the first half of the theorem, note that $P(X = x|N = 1) = \frac{1}{W}$ for all x. Consider a function $f(y) = (1 - \frac{y}{W})^2 - (1 - \frac{y+1}{W})^2$ when y takes real values in $(0, W)$. It can be shown that $f'(y) < 0$ and $f(\frac{W-1}{2}) = \frac{1}{W}$. As a result, $P(X = x|N = 1) < P(X = x|N = 2)$ for $x \in [0, \frac{W}{2} - 1]$. Therefore, the first half of the theorem is established.

For the second half, we first note that if $x \in [\frac{W}{2}, W - 1]$, $P(X = x|N = 1) > P(X = x|N = 2)$. In the following, we show that $P(X = x|N = 2) > P(X = x|N = n)$ for any $n > 2$. Let $a = (1 - \frac{x}{W})$, it is basically to show that

$$a^n - (a - \frac{1}{W})^n \geq a^{n+1} - (a - \frac{1}{W})^{n+1}$$

for $n \geq 2$ and $\frac{1}{W} \leq a \leq \frac{1}{2}$. We note that

$$\begin{aligned} & a^n - (a - \frac{1}{W})^n \geq a^{n+1} - (a - \frac{1}{W})^{n+1} \\ \Leftrightarrow \quad & a^n(1 - a) \geq (a - \frac{1}{W})^n(1 - a + \frac{1}{W}) \\ \Leftrightarrow \quad & 1 \geq (1 - \frac{1}{aW})^n[1 + \frac{1}{(1-a)W}] \end{aligned}$$

while

$$(1 - \frac{1}{aW})^n[1 + \frac{1}{(1-a)W}] = \frac{(1 - \frac{1}{aW})^n}{[1 - \frac{1}{(1-a)W}]}[1 - \frac{1}{((1-a)W)^2}]$$

and

$$\frac{(1 - \frac{1}{aW})^n}{[1 - \frac{1}{(1-a)W}]} \leq 1$$

because $(1 - \frac{1}{aW}) \leq [1 - \frac{1}{(1-a)W}]$ when $\frac{1}{W} \leq a \leq \frac{1}{2}$. ■

2) *Subsequent Rounds:* If the AP decides to have multiple rounds of contention, according to the maximum likelihood criterion, there should be another round of contention if

$$\begin{aligned} & P(X_1 = x_1, X_2 = x_2, \ldots, X_i = x_i|N_i = 1) < \\ & P(X_1 = x_1, X_2 = x_2, \ldots, X_i = x_i|N_i = n) \end{aligned}$$

for some $n > 1$. We show that

Theorem 2: The contention should continue after round i according to the maximum likelihood criterion if $x_i < \frac{W'}{2}$ where $W' = W - \sum_{j=1}^{i-1} x_j$.

Proof: As we assume there is no collision, the contention nodes transmitted in this contention period till round i will not transmit again because they have either entered the polling mode or have transmitted all data. The remaining contention nodes must have picked random numbers no less than $\sum_{j=1}^{i-1} x_j$, because otherwise they would have transmitted earlier. Given this knowledge, their initial backoff counters are uniformly distributed in $[\sum_{j=1}^{i-1} x_j, W - 1]$. As their backoff counters have been deducted by $\sum_{j=1}^{i-1} x_j$, equivalently, at this moment, their backoff counters are random number uniformly distributed in $[0, W' - 1]$ where $W' = W - \sum_{j=1}^{i-1} x_j$. We may regard W' as the equivalent contention window size. The theorem immediately follows applying Theorem 1. ■

C. *Robustness of cMAC*

We analyze the robustness of cMAC against packet losses.

1) *Coping with Mismatches of Node Modes:* The AP maintains the polling list and sets the node in the polling mode or contention mode with its ACKs. As packets can be lost, this polling list may not be the same as the set of nodes in the polling mode. This mismatch happens at times when node should transit from one mode to another, which is analyzed in the following.

Case 1. Consider a node in the polling mode, and the transition is when it sent the last packet in its queue.

Case 1.1 If this packet is received correctly by the AP but the ACK is lost, the node will consider itself still in the polling mode but the AP will consider it in the contention mode. cMAC addresses this problem by allowing a node, after sending the last data packet with polling, to switch to the contention mode whenever new data is received. Note that if the node has no new data, staying in the polling mode does not cause any problem because the AP has got the data the node transmitted. When new data is received, the node starts contention which allows it to compete for medium. Note that if the protocol does not allow the node to switch to contention mode, it will be stuck in the polling mode forever and lose opportunities to transmit.

Case 1.2 In case the last data packet is not received by the AP, from the node's point of view, it did not receive the ACK for the last packet and will remain in the polling mode. This is correct because the AP will consider the node in the polling mode and will poll it at some time. If the node gets new data before the AP polls it, according to the protocol, it will switch to the contention mode, while the AP considers it in the polling mode. This mismatch will introduce one more contention node and increase the level of contention, but should occur with low probability and does not cause node to lose opportunities to transmit.

Case 2. Consider a node in the contention mode and transmitting the first packet while having backlog.

Case 2.1 If the packet is received correctly but the ACK is lost, the node will consider itself in the contention mode but the AP will consider it in the polling mode. This has the same effect as Case 1.2 when the node receives new data.

Case 2.2 If the packet is not received, the node will consider itself in the contention mode, so will the AP. No mismatch in this case.

2) Operating Modes of the AP: When the AP is in the uplink contention mode, it does not send data packets. Both the contention nodes and the polling nodes compete for medium access. The network functions similarly to the 802.11 DCF, except that the polling nodes compete at a lower priority because they pick backoff counters from $[W/2, 3W/2-1]$ and every ACK from the AP resets the counter. In addition, the level of contention should gradually reduce because whenever a contention node sends a packet to the AP, if it still has backlog, it will enter the polling mode. If the AP wishes to leave the contention mode, it can wait until the first data packet is received and initiate polling with the ACK for the packet. As long as there is an active node, this packet will be received eventually. It could happen that the AP stops the contention mode while some contention node has not transmitted yet. This node can wait for the next round in which it will have a high probability to transmit successfully because its backoff counter has been deducted in the previous contention round and is likely to be smaller than those of other nodes.

When the AP is in the uplink polling mode, nodes in the contention mode will not transmit because the AP sets the duration in the ACK packet to forbid any node from transmission for a packet period except the node being polled. A node that was not polled may transmit, and cause collision, only if it did not receive the duration and did not detect the busy medium. If the polling message is not received correctly by the node being polled, it will not transmit. Even in this case, however, all nodes in the network will, at some point of time, start to decrement their backoff counters and some node will eventually transmit and the AP may resume polling with the ACK for this packet. If the data packet from a node is not received correctly by the AP, either due to collision or due to random loss, the AP does not send ACK and cannot poll the next node. Again, some node will start to transmit a packet eventually and the AP can resume polling with the ACK.

When the AP is in the downlink transmission mode, it continuously transmits downlink packets and a node transmits only if it did not receive the duration field and did not detect the busy medium. If the AP's data packet is not received correctly by the node or if the ACK is not received correctly by the AP, the AP may retransmit the packet, as in typical link layer protocols.

D. *Fairness*

cMAC relies on centralized control. As the AP uses round-robin to poll the nodes in the polling list, such nodes should have fair access to the medium against each other. The contention nodes compete in the contention period according to the 802.11 DCF and have the same level of fairness against each other provided by the 802.11 DCF. Between the contention nodes and polling nodes, note that if the number of polling nodes is not large, during a polling period and contention period, each node is expected to transmit once, therefore fairness is maintained. If the number of polling nodes is large such that not all nodes can be polled in one polling period, the contention nodes may have higher priorities. However, a contention node should enter the polling list after its transmission if it has more packets to send. Therefore, the contention nodes that gain advantage are those with only one packet. Such nodes should account for a small percentage of the traffic in most cases.

E. *Co-existence with Legacy 802.11 Nodes*

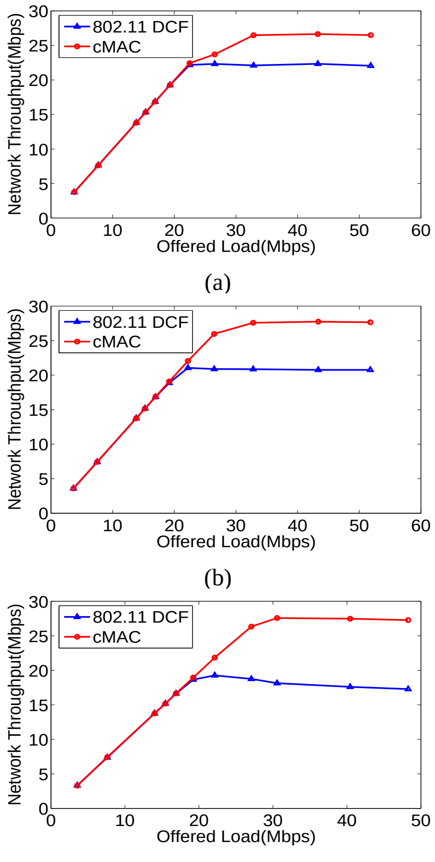
One of the nice features of cMAC is that it can co-exist with legacy 802.11 nodes. The only new type of message the protocol introduces is the polling message, which can be always piggybacked with ACK from the AP. The ACK packet to the cMAC nodes can be sent as a control frame using a different format from the ACK packet to the 802.11 nodes. With the duration field set appropriately, the 802.11 nodes will backoff and not interfere with the polling nodes. When in the contention period, 802.11 nodes will compete in the same manner as other cMAC contention nodes. Unlike the cMAC nodes, they will not abstain from the contention when the first packet is received. However, it is unlikely to be able to get a second transmission opportunity in the same contention period because it has to generate a new backoff counter which is likely larger than the number of idle time slots the AP waits before switching to the polling mode. In this sense, fairness is roughly achieved because the cMAC nodes also get one opportunity to send.

III. EVALUATIONS

We evaluate the performance of our cMAC protocol with the ns2 simulator. In our simulation, there are one AP and up to 50 nodes. We set the data rate to be 54 Mbps. To simulate normal operating conditions where the loss ratio is typically not high, we modify the loss mechanism in the ns2 simulator such that the lost ratios of both ACK and data packets are in the range of $[0, 0.1]$ and are set proportional to the receiving signal power. We set the maximum length of a polling period to be 5 ms to maintain a reasonable delay for nodes not currently in the polling table.

A. *Pure Uplink Traffic*

As cMAC mainly solves the problem of the uplink transmission, in the first set of simulations, we focus on the uplink performance with no downlink traffic.



(a)

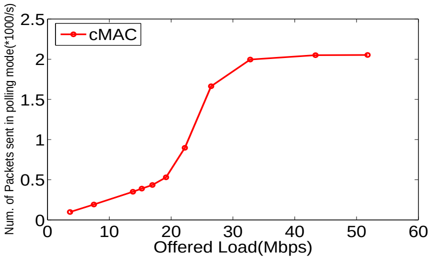
(b)

(c)

Fig. 1. Network throughput. (a) 10 nodes. (b) 20 nodes. (c) 50 nodes.

1) Throughput Comparison and Analysis: We measured the network throughput of 802.11 DCF and cMAC with one AP and 10, 20 and 50 nodes and show the results in Fig. 1. We can see that although two protocols have close performance at low load, as the load increases, the throughput of cMAC is much higher. The maximum gains are roughly 20%, 33% and 56% with 10, 20 and 50 nodes respectively. We achieve such gain in the high load regime because the majority of packets are transmitted in response to polling which is more efficient than contention. The gain is higher with more nodes because the contention in 802.11 DCF is more intense with more nodes.

To verify the source of the gain, for the 20 nodes scenario, we measured the number of packets that are transmitted in the polling mode and show the result in Fig. 2. It can be seen that more packets are transmitted in polling mode when traffic load is higher. We also counted the number of packet collisions per second in 802.11 DCF as well as in cMAC and show the result in Fig. 3. For 802.11 DCF, we see more collisions at higher traffic load. For cMAC, as the traffic load increases, the number of packet collisions first increases, then decreases, then stays at a low level. This is because at low traffic load the nodes mainly run in the contention mode; as the load increases, more and more nodes switch to the polling mode which reduces the contention and the collision.



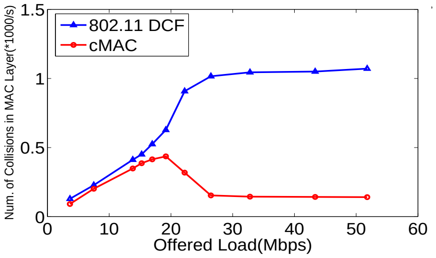


Fig. 3. Packet collision count.

2) Estimation of Contention Nodes: A key aspect of cMAC is the estimation of the number of contention nodes. We obtained the number of contention nodes in the network at the beginning of each polling period denoted as N. Note that the AP believes the number of contention node is 0 at the beginning of each polling period, therefore the value of N represents the estimation accuracy. The data was collected for the 20-node case with different traffic load, and the results are listed in Table. 1. We can see that for over 80.6% of the time, our estimation is accurate. The estimation is more accurate at higher traffic load, because the majority of active nodes are already in AP's polling list.

3) Packet Delay: We show in Fig. 4 the average delay as a function of traffic load when the network has 20 nodes. We can see that cMAC achieves smaller delay than 802.11 DCF.

B. Handling Downlink Traffic

In the second set of simulations, we introduced both uplink and downlink traffic and measured the throughput and average packet delay at various percentages of downlink traffic. The

Load (Mbps)	19.2	22.2	26.5	32.8	43.4
$N = 0$	80.6%	84.4%	90.0%	97.0%	98.6%
$N = 1$	6.45%	6.68%	7.60%	1.77%	0.85%
$N = 2$	4.04%	3.56%	1.05%	0.45%	0.12%
$N \geq 3$	8.91%	5.35%	1.34%	0.78%	0.43%

TABLE 1
ACCURACY OF ESTIMATION.

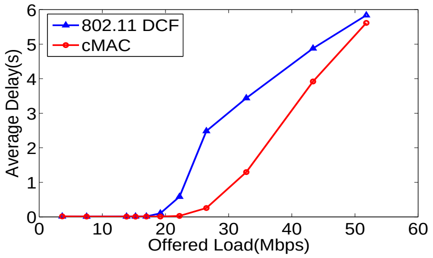


Fig. 4. Average delay of 20 nodes.

Downlink Traffic Percentage	Uplink Throu. (Mbps)	Downlink Throu. (Mbps)	Uplink Delay (ms)	Downlink Delay (ms)
20%	18.92	4.57	9.25	6.18
40%	15.26	9.15	8.52	7.07
60%	10.55	13.72	8.22	7.99
80%	4.79	17.98	8.91	9.03

TABLE 2
THROUGHPUT AND DELAY AT VARIOUS TRAFFIC LOAD.

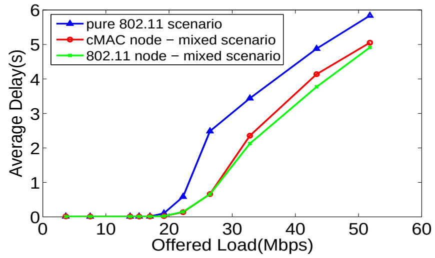


Fig. 5. Average delay of cMAC nodes and 802.11 nodes.

overall traffic load was set to be 25 Mbps, which is reasonably high but not saturating the network, allowing us to make observations based on packet delay. The results in Table. 2 show that the uplink/downlink throughputs are proportional to the offered loads and the uplink/downlink delays are almost the same. This proves that cMAC is capable of supporting both uplink and downlink traffic simultaneously.

C. Backward Compatibility

To show the backward compatibility with 802.11 nodes, in the third set of simulations, we used 20 nodes among which 10 nodes run cMAC and the rest ran 802.11 DCF. Fig. 5 shows the average packet delay of cMAC nodes as well as 802.11 nodes in this mixed scenario, along with the packet delay in a pure 802.11 scenario in which all 20 nodes ran 802.11 DCF. We see that the average delay of cMAC nodes and 802.11 nodes in the mixed scenario are similar, implying that they received similar services and were treated fairly by the network. Note that the packet delay in the pure 802.11 scenario is higher than those in the mixed scenario because cMAC reduces contention.

IV. RELATED PROTOCOLS

In this section we highlight the major differences between cMAC and other related protocols.

802.11 PCF. 802.11 PCF [1] supports polling but can be inefficient as the AP may poll nodes with no traffic, while with cMAC the AP only polls nodes with traffic. In addition, 802.11 PCF has one contention-free period in one beacon interval which is typically 100ms. With cMAC, the AP may initiate polling more frequently, hence improving the flexibility.

802.11e HCCA. 802.11e [2] provides HCCA which can initiate polling at any time, similar to cMAC. The key difference is that cMAC aggressively sets nodes into the polling mode and discourages such nodes from contention, which reduces the contention in the network, while HCCA does not support polling mode and contention mode of the nodes. cMAC also adopts a novel algorithm to determine the duration of the contention period, which is not available in HCCA.

802.11 DCF enhancements. There has been much research effort studying the performance enhancement for 802.11 DCF. For example, [3], [4] propose reducing collision by choosing backoff counters intelligently. Our cMAC is different from such works in that the polling mechanism brings further collision reduction and performance improvement. Many studies are based on new physical layers [5], [6], [7], [8]. All these works require specific hardware support while our cMAC can run on commodity wireless cards. In [9], [10], the number of competing nodes is estimated based on collision probability. cMAC's estimate is based on the number of idle time slots which is more readily observable because collision may be a rare event after polling is employed.

IdleSense. As discussed previously, IdleSense [11] is a distributed protocol using AIMD to adjust contention window size while cMAC is centralized and relies on maximum likelihood estimation to determine whether to continue contention.

V. CONCLUSION

We proposed cMAC, a centralized, polling-based protocol for wireless LANs in this paper. The key feature of cMAC is that the AP aggressively sets nodes with backlogged data into the polling mode and such nodes will basically not compete for medium access, which significantly improves the network performance by reducing the level of contention. The AP also opens up contention period for nodes with new data to transmit packets and announce queue states. We designed a simple algorithm for the AP to determine whether to continue contention based on the maximum likelihood estimation of the number of contention nodes, and also argued that cMAC is robust again packet losses. We simulated cMAC in ns2 simulator and the results show significant performance gain over 802.11 DCF and backward compatibility with legacy 802.11 ndoes. Our future works include extending cMAC to support multiple data rates.

REFERENCES

[1] IEEE Computer Society LAN MAN Standards Committee, *IEEE Standard 802.11, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 1999*.

[2] IEEE 802.11e, ”*Wireless LAN Medium Access Control (MAC) Enhancements for Quality of Service (QoS)*,” 802.11e draft 8.0, 2004.

[3] M. Ma and Y. Yang, “A Novel Contention-Based MAC Protocol with Channel Reservation for Wireless LANs,” *IEEE Transactions on Wireless Communications*, vol. 7, no. 10, pp. 3748-3758, October 2008.

[4] B. Li, W. Tang, H. Zhou, and H. Zhang, ”m-DIBCR: MAC Protocol with Multiple-Step Distributed In-Band Channel Reservation,” *IEEE Communication Letters*, vol. 12, no. 1, pp. 23-25, January 2008.

[5] S.Rayanchu, A. Mishra, D. Agrawal, S. Saha, and S. Banerjee, “Diagnosing Wireless Packet Losses in 802.11: Separating Collision from Weak Signal,” *IEEE INFOCOM*, 2008.

[6] J. Peng, L. Cheng, and B. Sikdar, “A Wireless MAC Protocol with Collision Detection,” *IEEE Transactions on Mobile Computing*, vol. 6, no. 12, December 2007.

[7] S. Sen, R. R. Choudhury, and S. Nelakuditi, “CSMA/CN: Carrier Sense Multiple Access with Collision Notification,” *Proc. of ACM MOBICOM 2010*, pp. 25-36, September 2010.

[8] J. Manweiler, N. Santhapuri, S. Sen, R. R. Choudhury, S. Nelakuditi, and K. Munagala, “Order matters: transmission reordering in wireless networks,” *ACM MOBICOM*, 2009.

[9] G. Bianchi and I. Tinnirello, ”Kalman Filter Estimation of the Number of Competing Terminals in an IEEE 802.11 Network,” *Proc. of IEEE INFOCOM 2003*, pp. 844-852, March 2003.

[10] A. L. Toledo, T. Vercauteren, and X. Wang, ”Adaptive Optimization of IEEE 802.11 DCF Based on Bayesian Estimation of the Number of Competing Terminals,” *IEEE Transactions on Mobile Computing*, vol. 5, no. 9, pp. 1283-1296, September 2006.

[11] M. Heusse, F. Rousseau, R. Guilier and A. Duda, “Idle Sense: An Optimal Access Method for High Throughput and Fairness in Rate Diverse Wireless LANs,” *Proc. of ACM SIGCOMM 2005*, vol. 35, pp. 121-132, August 2005.

[12] Network Simulator 2, http://www.isi.edu/nsnam/ns/.