Channel Access Throttling for Improving WLAN QoS

Bo Han*, Lusheng Ji†, Seungjoon Lee†, Robert R. Miller†, Bobby Bhattacharjee*
*Department of Computer Science, University of Maryland, College Park, MD 20742, USA
†AT&T Labs – Research, 180 Park Avenue, Florham Park, NJ 07932, USA

***Abstract*—The de facto QoS channel access method for the IEEE 802.11 Wireless LANs is the Enhanced Distributed Channel Access (EDCA) mechanism, which differentiates transmission treatments for data frames belonging to different traffic categories with four different levels of channel access priority. In this paper, we propose extending EDCA with Channel Access Throttling (CAT) for more flexible and efficient QoS support. By assigning different member stations different channel access parameters, CAT differentiates channel access priorities not between traffic categories but between member stations. Then by dynamically changing the channel access parameters of each member station based on a pre-computed schedule, CAT enables EDCA WLANs the benefits of scheduled access QoS. We also present evaluation results of CAT obtained from both simulations and experiments conducted using off-the-shelf WLAN hardware and open-source device driver. Our results show that CAT can proportionally partition channel capacity, significantly improve performance of multimedia applications, effectively achieve performance protection for admitted flows, and increase per cell VoIP call capacity by up to 41%.**

***Index Terms*—IEEE 802.11e; wireless multimedia networks; wireless LAN; QoS.**

I. Introduction

During the past decade, IEEE 802.11 based Wireless LAN (WLAN) technology has seen phenomenal growth. Virtually all laptops sold today are equipped with WLAN interfaces. WLAN is also becoming the communication technology of choice for more and more handheld appliances such as dual-mode Smart Phones, gaming pads, and media players. The types of software that communicate over WLAN also have expanded from web browsing and file downloading to multimedia applications that demand not only more bandwidth but also better Quality of Service (QoS) support.

The IEEE 802.11 standard specifies two medium access mechanisms for QoS data: the Enhanced Distributed Channel Access or EDCA, and the HCF (Hybrid Coordinator Function) Controlled Channel Access or HCCA [1]. EDCA is a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) based channel access mechanism. It provides differentiation in frame transmission treatment for frames belonging to different traffic categories. The differentiation is achieved by configuring transmissions of data frames of different traffic categories with different channel access parameters and thus differentiates their channel access priorities. On the other hand, HCCA is a pure scheduled access method which requires no carrier sensing and backing off during the controlled access phase. Instead, a scheduler residing at the Access Point (AP) fully controls accesses to medium within the Basic Service Set (BSS) and polls member stations for transmissions based on a pre-computed schedule.

While EDCA does offer service differentiation, it has certain limitations: (1) EDCA's QoS support is "soft" – it cannot guarantee parameterized service level requirements such as data rates and delay bounds; (2) EDCA service differentiation is coarse – it cannot provide service differentiation finer than the four defined traffic categories; and (3) EDCA channel access is not efficient – the contention window sizes are often too long which causes inefficient use of airtime. While most of the aforementioned issues of EDCA can be addressed by a scheduled access approach such as HCCA, virtually *none* of currently deployed WLAN devices (which are numbered in hundreds of millions) are capable of supporting HCCA easily (e.g., with only software update).

Motivated by this problem, we propose a new WLAN QoS approach called Channel Access Throttling (CAT) for improving WLAN QoS. CAT enables scheduled access-like fine granularity QoS policy using EDCA hardware. CAT is essentially an "upper MAC layer" scheduled channel access approach. Distinguished from other scheduled access approaches such as HCCA, a CAT AP does not have direct controls over medium. Instead, a CAT AP manages channel access through the same fundamental CSMA/CA with adjustable contention window mechanism as in EDCA. Thus CAT can also be viewed as an extension to EDCA. Because of this design, CAT is *compatible* with EDCA and can be readily implemented on today's EDCA hardware.

In a nutshell, CAT can function as follows. When it is time to "poll" a particular member station, the CAT AP changes the channel access parameters of its member stations so that this "polled" member station has higher channel access priority than all the other member stations. As long as the channel access priority difference is large enough, the "polled" station virtually has *exclusive* right to the channel because the other stations will be almost certainly in backoff stages. Then at the end of the "polled" period for this station, the CAT AP changes the channel access parameters of this station so that the "polled" station's channel access priority is no longer higher than its peers' priority. At the same time, the CAT AP can elevate the channel access priority for another member station to "poll" that station. In this fashion, the CAT AP changes the channel access parameters of the stations to "poll" them according to a pre-computed service schedule.

Figure 1 illustrates another example of CAT in action. Instead of elevating only one station's channel access priority relative to other stations, CAT can also elevate the channel access priorities of multiple stations at the same time. In

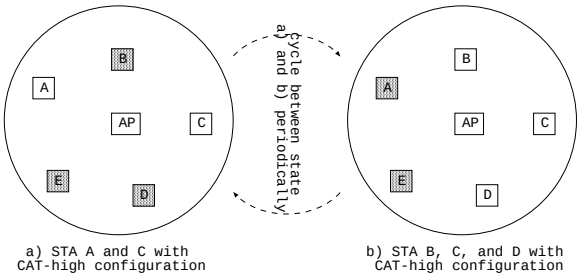

Fig. 1. In this example, CAT allows two groups to gain prioritized access to the channel at different times, where each group has multiple stations.

this example the schedule has only two phases. During its first phase stations A and C are assigned high channel access priority and during its second phase stations B, C, and D are assigned high channel access priority. Being able to "poll" multiple stations together is a distinct benefit of CAT compared to other scheduled access mechanisms, in that the stations that are polled together can still resolve contention with CSMA/CA.

Our contributions of this paper are three-fold:

- We present the concept and design of CAT which enables scheduled access-like fine granularity QoS policy using EDCA hardware.
- We demonstrate that we can implement CAT using off-the-shelf hardware and open-source device driver.
- We perform extensive testbed experiments and simulations using realistic application scenarios and demonstrate that CAT can achieve better QoS differentiation and channel access efficiency than EDCA.

The rest of this paper is organized as follows. We first give a brief introduction of the EDCA and HCCA mechanisms in Section II. In Section III, we describe the detailed design of CAT and discuss how to choose CAT parameters. In Section IV, we use simulations to evaluate the performance of CAT. We describe our implementation and testbed experiment results in Section V. After reviewing related work in Section VI, we conclude in Section VII.

II. Background

In this section, we briefly review EDCA and HCCA. In particular, we focus on the CSMA/CA channel access mechanism of EDCA, as CAT uses the same mechanism.

The core of EDCA is to differentiate channel access priorities for data frames in four different Access Categories (ACs): AC_VO (voice), AC_VI (video), AC_BE (best effort) and AC_BK (background). An IEEE 802.11 QoS station has a separate output queue and a backoff procedure called EDCA function for each of these ACs. Each AC uses a different set of channel access parameters, which essentially specifies how a station should backoff its transmission if encountering a busy channel, and thus get differentiated service. EDCA uses the following channel access parameters:

- *Arbitration InterFrame Space Number (AIFSN)* controls the duration of Arbitration InterFrame Space (AIFS): *AIFS = AIFSN* $\times$ *aSlotTime + SIFS*, where aSlotTime is the duration of a time tick in the WLAN system and the SIFS is the smallest inter-frame gap.
- *CWmin and CWmax* define the region for contention window (CW) value. The upper bound for CW is initially set to CWmin, and doubles on every transmission failure, until it reaches CWmax, after which it stays constant at CWmax. The actual CW is chosen randomly between 0 and this upper bound.
- *Transmission Opportunity Limit (TXOPLimit)* is the maximum duration for TXOP (Transmission Opportunity). During a TXOP, the owner member station may transmit multiple frames without observing the usual channel access deferring rules.

	Description	CW_min	CW_max	AIFSN	TXOPLimit
AC_VO	Voice	3	7	2	1.504 ms
AC_VI	Video	7	15	2	3.008 ms
AC_BE	Best Effort	15	1023	3	0 ms
AC_BK	Background	15	1023	7	0 ms

TABLE I
EDCA Parameters for IEEE 802.11a/g

An AP announces these parameters to its associated stations using a special Information Element in its Beacons and other management frames. The recommended default values of these parameters for IEEE 802.11a/g are summarized in Table I. For IEEE 802.11b, TXOPLimits are different (3.264 ms for voice and 6.016 ms for video), while the other values are the same.

Figure 2 illustrates the channel access mechanism for EDCA. Each backoff procedure independently calculates the total waiting time before the next transmission. First, the backoff procedure waits for AIFS duration and checks if the channel is idle for the entire AIFS period. If it is, the contention window phase begins with a backoff counter initialized to a random number between 0 and the current CW upper bound. The backoff counter will then be decreased in each following idle time slot. When a backoff counter reaches zero, the station can attempt to transmit a frame in the corresponding AC (Access Category). If backoff counters of multiple backoff procedures simultaneously become zero in the same QoS station, the highest-priority procedure wins the internal contention, and all other procedures act as if an external collision occurred. Of course, the winner procedure still needs to contend with procedures from other stations for the wireless channel. Note that due to this contention-based operation among QoS stations, EDCA cannot guarantee "hard" QoS parameters such as delay bounds and data rates. Also, a data frame's access category is determined from the ToS bits of its IP header, which is out of control by the WLAN system. Neither can EDCA provide finer service differentiation to flows with the same IP ToS bits. This makes supporting certain services (e.g., E911 calls are required to have priority over regular calls) difficult.

Compared with EDCA, HCCA uses a traffic scheduler called Hybrid Coordinator (HC) to support parameterized QoS for wireless multimedia applications. The HC always has the highest priority to access the wireless medium, because it can start transmission when the channel is sensed to be idle

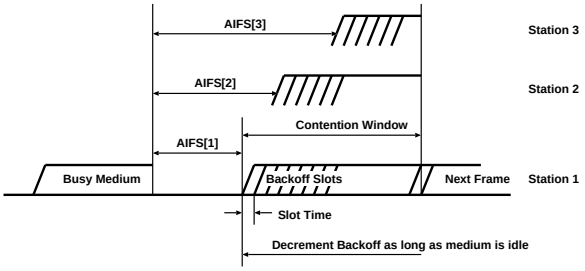

Fig. 2. EDCA Channel Access

for a very short interval (i.e., PIFS). This allows the HC to initiate a contention-free interval called Controlled Access Phase (CAP) for any station at any time. As a result, while a station may gain channel access based on EDCA rules, the HC can also initiate CAPs to give the station exclusive channel access to transmit its frames. This allows the HC to control transmissions of traffic flows more precisely to satisfy their corresponding QoS requirements.

III. Channel Access Throttling

A. Design of CAT

The key idea behind CAT is that it extends the control of EDCA's channel access differentiation from one dimension, different access category, with two additional dimensions, different member station and different time. This extension enables CAT a great deal of flexibility to satisfy the specific system design goals.

For example, certain systems (e.g., emergence response systems) can employ CAT to differentiate channel access priorities among stations. That is, if special stations are present, they always have higher channel access priority than other stations, regardless of what kind of traffic they are sending. For these systems, CAT may only employ two access priority groups: high access priority (or CAT-high) group for special stations and low access priority (or CAT-low) group for regular stations. Obviously CAT can support more than two CAT priority groups so that members of a particular priority group have better chance of accessing the channel than members of lower priority groups when members of all higher priority groups are either absent or have no traffic to send.

Nothing prevents CAT from still differentiating channel access priority based on traffic access categories because CAT is fully compatible with EDCA. Depending on specific design requirements, a CAT implementation may choose to use station identity as the primary differentiation criteria and access category as secondary, or vice versa.

Although the flexibility of CAT is easy to comprehend and applying different configurations of CAT is straightforward, the benefits of specific CAT configuration would often be better understood under the context of specific target systems. Thus in this paper we primarily focus on the simplest and the most generic implementation of CAT. That is, we will only have two access groups, CAT-high and CAT-low. We do not differentiate access categories. Additionally there is at most only one station in the CAT-high group at any given time.

By ensuring the channel access priority difference is large enough, CAT can essentially allow only the CAT-high stations to access the channel because CAT-low stations will be almost certainly in backoff stages. Then by rotating the memberships of these two access groups among the member stations, CAT provides transmission opportunities to all of its member stations according to the requirements of the admitted QoS streams.

By controlling how much time a station remains in CAT-high group, CAT can proportionally allocate bandwidth among stations, which can be translated to data rate service guarantees. By carefully planning out when within a service schedule a station becomes a CAT-high station, CAT can also support delay bound requirement for QoS streams. However, note that both are difficult to support under the original EDCA mechanism.

An added benefit of applying member station differentiation and keeping the membership of CAT-high group small is improved channel access efficiency. When CAT allows only one station entering the CAT-high group at one time, there is *no* need for contention window for this station because virtually no other station will compete for channel access with it. As a result, reducing deferral time increases channel utilization efficiency for CAT compared to the original EDCA mechanism.

CAT may manage stations entering and exiting access groups *periodically*. In this approach, a CAT AP sets up a schedule relative to a periodic reference time that is available to all member stations. For instance, the beginning of each Beacon interval, also known as the Target Beacon Transmission Time (TBTT), can be used as a reference time in practice. The time between two consecutive reference times is the service cycle period. A CAT schedule configuration contains two sets of EDCA channel access parameters, one for CAT-high group and the other for CAT-low group. The configuration also lists the starting and ending times for each station to be in CAT-high group and CAT-low group within each service cycle. After receiving the configuration, each member station needs to periodically adjust its EDCA channel access parameters to the specified values at specified times according to its membership in either CAT-high or CAT-low group.

CAT can also throttle channel access *on-demand*. In this case, the AP may announce the CAT-low parameters as the default configuration in its Beacon messages just as how a regular EDCA AP announces EDCA parameters to its associated member stations. Then, at specific times, the AP sends CAT-high channel access parameter configuration to a specific member station in a similar fashion to *polling* messages used in HCCA. However, unlike in HCCA where the polled TXOP starts immediately after the reception, in CAT the on-demand configuration may also instruct the recipient member station when to enter and exit the CAT-high group, hence solving the problem that a CAT AP may not be able to access the channel when it needs to poll a station.

STA1	STA2					
	2, [1, 3]	2, [3, 7] (AC_VO default)	2, [7, 15] (AC_VI default)	3, [15, 1023] (AC_BE default)	7, [15, 1023] (AC_BK default)	15, [1, 3]
2, [1, 3]	\	0.6370, 0.0699	0.6607, 0.0137	-	-	-
2, [3, 7]	0.0699, 0.6370	\	0.3816, 0.0603	0.3996, 0.0014	-	-
2, [7, 15]	0.0137, 0.6607	0.0603, 0.3816	\	0.2178, 0.0227	-	-
2, [15, 31]	0.0031, 0.6653	0.0132, 0.3958	0.0414, 0.2143	0.1086, 0.0886	0.1138, 0.0357	-
2, [15, 1023]	0.0001, 0.6666	0.0042, 0.3987	0.0317, 0.2161	0.1065, 0.0912	0.1135, 0.0360	-
3, [15, 1023]	-	0.0014, 0.3996	0.0227, 0.2178	\	0.1118, 0.0501	-
7, [15, 1023]	-	-	-	0.0501, 0.1118	\	-
15, [1, 3]	-	-	-	-	-	\
15, [511, 1023]	-	-	-	-	-	-

TABLE II
CHANNEL ACCESS PROBABILITIES FOR STATIONS USING TWO DIFFERENT SETS OF CAT PARAMETERS

B. Choosing CAT Parameters

In this section, we study how to choose the proper parameters for different CAT access groups. In particular, we borrow the EDCA model proposed by Zhu and Chlamtac [2]. This model is based on Bianchi's seminal work [3], which analyzes IEEE 802.11 DCF performance using a two dimensional Markov chain. Although this model makes a number of simplifying assumptions, we find it sufficient for understanding the system behavior of CAT when we assign different parameter sets to different stations.

We prefer to use AIFSN as the primary means to differentiate channel access priority because AIFSN differentiation is more effective than contention window differentiation [4]. From the model by Zhu and Chlamtac, we can obtain the channel access probability of each station, which is the probability that a member station transmits a packet in a randomly chosen "generic" time slot. The generic time slot can be either idle or busy time slot. The busy time slots may be caused by either successful transmissions or collisions. Even if the duration of busy time slot may be much longer than that of idle time slot, the model treats it as a single time slot. As a result, the channel access probability is different from channel utilization for each station. We are interested in finding out how much *separation* in channel access probability two priority groups experience, when we vary the parameter sets.

While we can use the model by Zhu and Chlamtac for more general scenarios (e.g., with more than two priority groups), for ease of illustration, we focus on a simple scenario with only two member stations: STA1 using CAT-high channel access parameters and STA2 uses CAT-low parameters. We denote the channel access parameter sets as $\{\text{AIFSN}_i$, [CWmin_i, CWmax_i]$\}$, where $i = 1, 2$ denotes each station. Table II summarizes the numerical results for multiple pairs of EDCA parameter sets, which are derived from the process outlined in the appendix. In each cell, the first number is the channel access probability for STA1, and the second number is for STA2. We use the symbol "-" when the access probability of STA2 is zero. For instance, when we use $\{2, [1, 3]\}$ for STA1, according to the model, STA2's traffic with $\{3, [15, 1023]\}$ cannot access the channel at all, traffic with $\{2, [7, 15]\}$ can access the channel with probability less than 2%, and traffic with $\{2, [3, 7]\}$ can split the channel access with STA1 at the ratio of 1:9. We also note that, the channel access probabilities of these two stations may not add up to 1 because there are also idle time slots. The result in the table illustrates that by varying the EDCA parameters, CAT can achieve different levels of service differentiation between CAT groups, where in some settings, CAT-low stations virtually never compete with CAT-high stations for channel access. Moreover, the result in Table II also provides an insight into the throughput sharing when we use CAT in the presence of legacy EDCA stations that use standard recommended parameter values.

C. Discussions

In addition to ensuring much higher access priority than the CAT-low group, the choice for channel access parameters of CAT-high group should also concern how many stations are put into the CAT-high group. Because CAT service cycle period is usually limited by the service intervals of QoS demanding applications, generally speaking, the fewer stations we put into the CAT-high group, the more CAT group membership transitions we need to iterate through all stations, and the shorter each CAT period becomes. In this case, if each transition incurs a fixed cost, the overall channel efficiency becomes lower. On the other hand, the more stations we put into the CAT-high group, the more contention these stations experience for channel access. In this case, we need to set backoff windows carefully to resolve contention efficiently.

While channel access priority separation is certainly the primary concern for choosing parameters for CAT-high and CAT-low groups, the CAT-low parameters should also address the concern of overall channel efficiency. This is because if the CAT-high group does not have data to send, the channel is idle and eventually some CAT-low station will complete its channel access deferral and begin to transmit. Since any unnecessary channel idling is a waste of radio resource, the channel access parameters for the CAT-low group should be set in such a way to avoid unnecessarily long channel access deferral.

IV. SIMULATION EXPERIMENTS

To evaluate the performance of CAT, we conducted extensive simulations. Due to space limitation, only selective results are presented in this paper. In particular in this section we present only results that we have not verified with testbed experiments, namely the results of VoIP service capacity study, which is difficult to experiment in real testbed due to its

Parameter	Value	Parameter	Value
Slot duration	9 μs	Basic transmission rate	6 Mbps
SIFS	16 μs	ACK frame length	14 bytes
PIFS	25 μs	RTP header length	12 bytes
DIFS	34 μs	UDP header length	8 bytes
PLCP header	4 μs	IP header length	20 bytes
PLCP preamble	16 μs	MAC header length	30 bytes
Beacon interval	100 ms	HCCA retry limit	1

TABLE III
PARAMETER SETTINGS USED IN THE SIMULATION

scale. Simulation results that have been verified by testbed experiments of the same scenarios will be introduced later in the next section.

A. Simulation Setup

For the simulation study, we use ns-2 simulator with an enhanced 802.11 module called *yans* [5]. Compared to the default 802.11 module in ns-2, the *yans* extension implements a more detailed physical layer model and a new MAC layer which supports both EDCA and HCCA. In particular, the *yans* extension implements a Bit Error Rate (BER) based model for simulating wireless communication which is closer to real WLAN receiver behavior and more realistic when simulating different transmission rates than the default ns-2 module. We summarize the parameters used in our simulations in Table III, in which all the values of durations and inter-frame spaces are for IEEE 802.11a. In all simulation runs, we randomly and uniformly deploy member stations in a square area centered at an AP. No rate adaptation is used.

In our simulations, we focus on the scenario where all traffic is VoIP. We simulate Constant Bit Rate (CBR) VoIP traffic generated by the G.711 voice codec. We use the same parameters as [6], with sample size being 80 bytes, sample interval at 10 ms, and delay bound of 50 ms. Each VoIP packet payload contains two samples. As a result, the VoIP packet interval is 20 ms and the MAC layer payload length is 200 bytes. The main factor that negatively impacts VoIP QoS is missing packets, which can occur due to packet losses (e.g., due to transmission errors, queue overflow) and packet expiry (i.e., packet arrival after delay bound). According to the ITU-T's E-model [7], a 5% packet missing rate is enough to reduce user satisfaction level on an otherwise perfect VoIP call to "some users dissatisfied". In our experiments, we use the 5% packet missing rate as a cut-off threshold to determine how many calls a WLAN cell can support. We increase the number of stations one by one, each of which generates a full duplex VoIP call with the AP. If a newly added call causes some already admitted calls including itself to have larger than 5% packet missing rate on either direction for more than 100 seconds, we consider that we have just exceeded the maximum number of calls the WLAN cell can support.

We use the periodic approach for CAT (see Section III-A) in our simulation experiments. Only one member station enters CAT-high group, and the duration each member station remains in CAT-high group is equal. The AP also enters CAT-high state because VoIP traffic is bi-directional and the AP needs to send VoIP packets to the member stations. To meet the 50 ms delay requirement, we have multiple service cycles (i.e., 5) within each 100 ms Beacon interval.

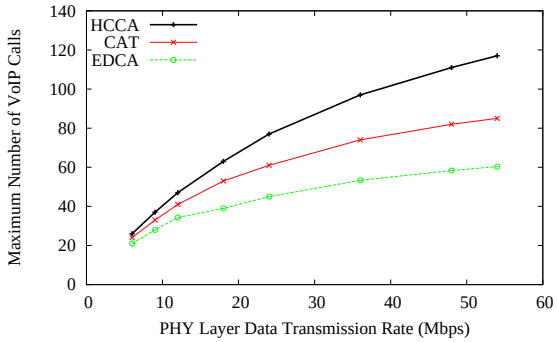

Fig. 3. Maximum number of VoIP calls supported in a WLAN cell

Since the CAT-high group has only one node at a time in our experiments, we assign to CAT-high group the minimum channel access parameters allowed in the standard (i.e., CWmin = CWmax = 0, AIFSN = 1 for AP and 2 for member station), so that we make an efficient use of channel. To improve the channel utilization further, we allow the AP to send all downstream VoIP packets in its queue in one transmission opportunity (TXOP). For this, AP uses the number of active VoIP calls to calculate the appropriate TXOPLimit to finish sending one packet for each VoIP call. We set the TXOPLimit for member stations to 0 ms, which means that they can only send out 1 packet for each TXOP. For CAT-low period, we use {AIFSN, [CWmin, CWmax]} = {15, [511, 1023]}, which sufficiently separates the channel access probability between CAT-high and CAT-low groups.

For comparison, we also simulated the same scenario using EDCA and HCCA. For HCCA, we use the reference scheduler in *yans*, which follows the description in the IEEE 802.11e document. The scheduler first chooses a Service Interval (SI), which is the interval between the start of two successive periodical polling periods. The service interval is calculated as a number that is not larger than any of the maximum service intervals of the admitted streams, which are usually the delay bounds of these streams divided by the retry limit of HCCA. In our simulations, we set the service interval to be 40 ms.

B. Results

In Figure 3, we show the maximum number of VoIP calls per cell that can be supported by CAT, EDCA, and HCCA, for different PHY layer transmission rates. Each data point represents an averaged result of 9 runs—three different random node placement scenarios each with three runs with different random seeds. As we can see, the VoIP service capacity of CAT significantly exceeds that of EDCA. For instance, when the physical layer transmission rate is 54 Mbps, a WLAN cell implementing CAT can support 85 calls, which is 41.67% higher than 60 calls an EDCA cell can support. This increase is due to several reasons. First, the downlink (AP to stations) is the major bottleneck for EDCA VoIP performance. While all downstream VoIP packets go through the AP, the EDCA configurations do not provide significant

channel access priority advantage for the AP. As a result, packets in the AP's queue do not get enough channel access opportunities. In contrast, CAT is able to allocate channel bandwidth in a finer granularity and give the AP sufficient transmission opportunity to send all downstream packets and satisfy QoS constraints. Note that there exist several other mechanisms to protect AP's traffic. For example, Pilosof et al. propose to adjust TCP receiver window size to alleviate this problem [8]. Their scheme operates above the MAC layer and works only for TCP traffic, while CAT can support wider range of applications. Another reason for CAT's superior performance is that CAT allows us to use very short contention windows for stations when they are in CAT-high group, much shorter than those used in EDCA. The time saving due to shorter channel sensing and backoff allows the WLAN cell to exchange more packets and thus support more calls.

Figure 3 shows that the VoIP capacity of CAT is still not as good as that of HCCA. This is expected for two main reasons. First, HCCA uses shorter inter-frame spacing. As CAT is compatible with EDCA devices, we did not implement such spacing in our simulation code for CAT. Secondly, HCCA piggybacks ACK whenever possible, i.e., on a reverse direction frame. We did not implement such feature for CAT again because we want CAT to be fully compatible with regular EDCA implementations.

V. Testbed Experiments

In this section, we present evaluation results of CAT using a testbed which we constructed with Commercial Off-The-Shelf (COTS) WLAN hardware. We first describe our implementation details before presenting the results.

A. Testbed Implementation

We implemented CAT by extending the open-source MadWifi wireless device driver (v0.9.4) [9]. MadWifi driver is developed for WLAN devices with Atheros chipset, which is an EDCA platform. The Atheros Hardware Abstraction layer (HAL) provides a call interface for configuring EDCA related parameters. MadWifi driver initializes station EDCA channel access parameters to the default values recommended in the standard. If a station receives a Beacon message containing EDCA parameters Information Element, its MadWifi driver will configure the Atheros hardware accordingly. We modified the driver such that a station changes EDCA parameters when triggered by a timer (periodic) or when receiving a CAT message (on-demand). The main challenge in the periodic approach is the synchronization of CAT timers among different member stations, as poor synchronization among CAT timers on different stations may cause overlapping of CAT-high periods of different stations and increase contention level during these overlapping periods. In our implementation we use the Beacon messages as timer synchronization signals and CAT operation cycle coincides with Beacon interval. Member stations repeat their CAT timer settings, i.e., when to enter and leave CAT-high and CAT-low states, for every Beacon interval. Every time a new Beacon message is received by a station, the station re-synchronizes its CAT timer with the Beacon reception.

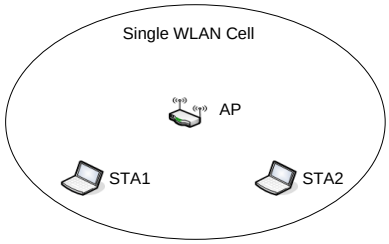

Fig. 4. Experiment Topology. We use a case where two stations are within the transmission range of each other.

For a CAT-high period, since we have only one CAT-high station in our experiments, we use the smallest possible parameter values for efficiency and set {AIFSN, [CWmin, CWmax]} = {2, [1, 1]}.[1] For a CAT-low period, we use {7, [3, 7]}. As we mentioned before, by using small CWmin and CWmax values for the CAT-low state, CAT-low stations can access the channel rapidly when the CAT-high station has no packets to send. Also in MadWifi, the channel access parameters are associated with transmission queues. When a transmission queue's channel access parameters are changed, only subsequent frames taken off this queue will be configured with the new values. To the frame that has already been de-queued and in its sensing and backoff stages, it is still configured with the previous parameter values. MadWifi driver does not offer any interface to interrupt or reset the state of such a frame. In other words any packet already in sensing and backoff stages effectively blocks the station from changing its CAT state. For this reason, using small CWmin and CWmax in our experiments also reduces the impact of such blocking.

B. Experiment Results

1) Proportional Partitioning of Wireless Channel Capacity: For this group of experiments, we use three wireless nodes—an AP and two member stations as shown in Figure 4. We place these nodes about one meter apart from each other. We run Iperf [10] on these member stations to generate CBR UDP streams at the rate of 24 Mbps. The packets are addressed to the AP. We set the Type of Service for all data packets to AC_VO category in EDCA because this is the EDCA configuration with the least access deferral and thus the highest efficiency. We set the PHY layer transmission rate for all nodes to be 24 Mbps and the transmission power 17 dBm. The Beacon interval of the AP is 100 ms. In this experiment, STA1 is in the CAT-high state during the first 70% of each Beacon interval, and STA2 is in the CAT-high state in the rest of 30%.

We show the experiment results for throughput in Figure 5. The throughput numbers are the average throughput during the corresponding experiment segment. CAT is enabled for both STA1 and STA2 at around 120s into the experiment and

[1] In the MadWifi driver, the smallest possible value of CW is 1, because it uses exponential form for this value.

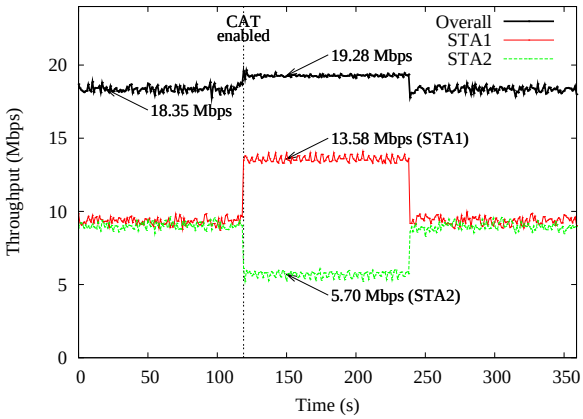

Fig. 5. Testbed Result for Partitioning of Channel Bandwidth with CAT

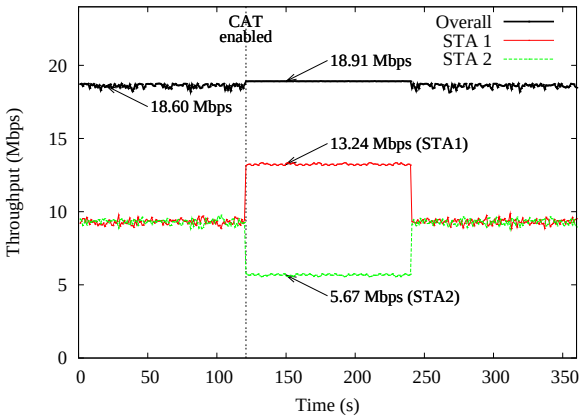

Fig. 6. Simulation Result for Partitioning of Channel Bandwidth with CAT

disabled at around 240s. As we can see, when CAT is not enabled, these two member stations share the radio resource equally because they both use EDCA and their traffic ACs are the same. However when CAT is enabled, the two stations achieve throughput close to the ratio of 7:3 (13.58 Mbps v.s. 5.70 Mbps).

Figure 5 also shows that using CAT can increase the overall throughput of the system. Without CAT the average overall throughput is about 18.35 Mbps. When CAT is enabled the average overall throughput increases to 19.28 Mbps which is very close to the maximum throughput (around 19.30 Mbps) that can be achieved by a single transmitter under the default EDCA setting for voice traffic.

We also present the simulation results for the same scenario with the same parameters in Figure 6. The high similarity between the simulated and experimented results indicats that the simulator is fairly accurate. The minor differences are mainly due to two reasons. First, in the simulator we were able to reset the channel access parameters for frames that are already de-queued from the transmission queue and in channel sensing and backoff stages. Second, in simulation clocks on different stations are perfectly synchronized and time slot boundaries are perfectly aligned. In reality such synchronization and alignment errors likely increase collision

	Frame Size	FPS	Avg Bit Rate	Max Bit Rate
V1	720 × 576	29.97	2.870 Mbps	5.399 Mbps
V2	1280 × 720	59.94	3.068 Mbps	3.933 Mbps
V3	1920 × 1080	29.97	10.269 Mbps	17.536 Mbps

TABLE IV
VIDEO CLIP PARAMETERS

	EDCA			CAT for V3		
	#1	#2	#3	#1	#2	#3
V1	∞	75	68	-	-	-
V3	12	12	12	77	77	54

TABLE V
PSNR (IN DB) FOR CLIPS V1 AND V3 STREAMED BY STA1 AND STA2

probability. And thirdly environmental reasons may have also taken a toll on throughput.

2) Service Differentiation for Video Streams: We also conducted experiments to demonstrate how to differentiate services for traffic flows of the same AC. Lacking support for admission control, EDCA accommodates all traffic flows and lets them compete for channel access. If the channel capacity is not enough to satisfy all flows, the typical outcome is that none of the flow gets satisfactory service if all flows are of the same AC. On the other hand, CAT can be used as a means for admission control by allocating admitted flows enough CAT-high time to satisfy their needs, as demonstrated by experiments in this section. The integration of CAT into admission control algorithms will be an interesting future work.

For this group of experiments, the same topology shown in Figure 4 is used. We run VLC media player v0.8.6 on STA1 and STA2 to stream video clips to the AP using RTP over UDP. Three MPEG2 video clips are used in these experiments. Their parameters are summarized in Table IV. We also run VLC media player on the AP to dump the received video streams into files. Buffering of video packets is disabled. The dumped files are then compared with the original video files to determine streaming quality, which is measured quantitatively using PSNR (Peak Signal-to-Noise Ratio). We use the Video Quality Measurement Tool[2] from the Moscow State University to calculate the PSNR. Perfect streaming which results a dumped video file identical to the source video file has a infinite PSNR. According to studies shown in [11], [12], video streaming with PSNR less than 20 dB is generally not acceptable.

In the first set of experiments STA1 and STA2 stream V1 and V3 respectively. We use channel 52 of IEEE 802.11a with data rate of 24 Mbps. The videos use AC_VI when under EDCA and the same CAT-high and CAT-low parameters as in the previous experiment when under CAT. The only difference is that we set AIFSN of CAT-low stations to 4, the smallest possible value based on the guideline for choosing CAT parameters. Each experiment was repeated three times. As Table V shows, under EDCA because the overall channel capacity is sufficient to support V1's data rate (24 Mbps

[2]http://www.compression.ru/video/index.htm

	EDCA			EDCA Trick	CAT Protecting STA1			CAT Protecting STA2		
	#1	#2	#3		#1	#2	#3	#1	#2	#3
STA1	18.23	18.16	18.12	18.23	∞	∞	∞	-	-	-
STA2	19.08	19.12	19.05	19.01	-	-	-	∞	∞	∞

TABLE VI
PSNR (IN DB) FOR CLIP V2 STREAMED BY STA1 AND STA2

transmission rate vs. 5.399 Mbps peak data rate and 2.87 Mbps average data rate), the V1 stream performs quite well. However the V3 stream is penalized severely. On the other hand, when CAT is used as a means for admission control with the V3 stream being the admitted flow, CAT is able to maintain the quality of the V3 stream at a level well above the "acceptable" range.

In the next group of experiments, we experiment with similar settings with another video stream V2. Because V2 has a rather low data rate, we reduce the WLAN transmission rate to 6 Mbps so that the bandwidth is not enough to support two streams of V2. Table VI shows the PSNRs of the same video clip (V2) streamed simultaneously by STA1 and STA2 under EDCA vs. under CAT. Under EDCA, neither stream gets satisfactory quality. Even when one stream (sent by STA1) is elevated to AC_VO, the category with the highest channel access priority under EDCA (denoted by "EDCA Trick"), the promoted stream still has poor quality , because there is not enough service differentiation between these two streams. On the other hand, when CAT is used for protecting any stream, it achieves perfect streaming quality for the protected stream.

In summary, we have demonstrated that we can implement CAT using the currently available hardware and open-source device driver. Our experiment results show that CAT can partition the network resource among member stations and provide prioritized service to selected stations when the wireless resource is insufficient to support all flows.

VI. RELATED WORK

A. Bandwidth Allocation, Admission Control and Scheduling

To address the challenges of bandwidth allocation in IEEE 802.11e wireless LANs, Xiao et al. study nine bandwidth allocation schemes for multimedia traffic [13]. Motivated by the lack of efficient schedulers for HCCA, Boggia et al. propose two feedback-based bandwidth allocation algorithms, feedback based dynamic scheduler (FBDS) and proportional-integral (PI)-FBDS, which utilize classic discrete-time feedback control theory [14]. Fallah and Alnuweiri develop a new access scheduling framework for IEEE 802.11e MAC, named controlled access phase scheduling (CAPS) [15]. Skyrianoglou et al. develop an adaptive traffic scheduling algorithm for HCCA, which allocates the available bandwidth based on the actual traffic buffered in the member stations and is very suitable for VBR traffics [16]. Cicconetti et al. propose a real-time HCCA scheduler which can separate online activities from offline activities [17]. All these works are complements to our CAT framework and can be built on top of it.

B. Analytical Models Verified by Simulation Study

Bianchi et al. provide a thorough understanding of the principles behind the QoS related operations in IEEE 802.11e EDCA [4]. They show that tuning the AIFSN parameter leads to more effective and robust operation than the tuning of CWmin parameter. Banchs and Vollero propose an analytical model for the throughput performance of an IEEE 802.11e wireless LAN, as a function of the EDCA parameters [18].

To achieve proportional bandwidth allocation, Hu and Hou design a MAC contention control (MCC) protocol [19]. They identify two parameters, the number of collisions between two consecutive successful transmissions and the number of consecutive idle slots, that are stable at the network's optimal operational state. They adjust the measured values of these two parameters to their optimal levels by changing the packet dequeuing rate from the interface queues. Nassiri et al. propose a channel access mechanism that can provide both relative proportional throughput allocation and absolute priorities in 802.11 wireless networks [20]. The method is based on the idea of the Idle Sense [21]: stations dynamically control their contention window size based on the measured average number of idle slots between transmission attempts. Compared with CAT, these methods are still contention-based mechanisms which can not significantly reduce the possible collisions.

C. Experiment Study of EDCA Parameter Tuning

To complement previous theoretical analysis and simulation studies, Banchs et al. perform an experimental study of EDCA focusing on traffic engineering and service guarantee [22]. Dangerfield et al. experimentally study the effect of background data traffic on voice calls in an IEEE 802.11e wireless LAN [23]. Ge et al. develop a multi-class model for the problem of adaptive parameter tuning in IEEE 802.11e EDCA [24]. They theoretically analyze the role of AIFSN and TXOPLimit parameters on service differentiation and show that it is desirable to only tune one set of parameters at a time. Based on the developed end-to-end VoIP transmission quality evaluation method, Narbutt and Davis experimentally evaluate the performance of EDCA to provide voice service over IEEE 802.11e wireless LANs [25]. They show that proper AIFSN and CWmin parameters can protect voice calls from background data traffic, although tuning the AIFSN parameter is more effective than tuning the CWmin parameter.

VII. CONCLUSIONS

In this paper, we propose a new wireless channel access approach called CAT for improving WLAN QoS. CAT combines the channel access mechanism of EDCA with the principle of

scheduled access. As a result, CAT can benefit EDCA devices, which are currently only capable of supporting differentiated service QoS, with scheduled access QoS like features. We have implemented CAT using off-the-shelf hardware and the open-source MadWifi device driver and performed various experiments on a testbed using our implementation. The experiment results show that not only can CAT successfully partition the wireless channel capacity proportionally, CAT can also increase channel capacity by improving channel access efficiency. We also present simulation results that demonstrate CAT can be used to improve the performance of VoIP applications and increase the per cell capacity by up to 41%.

ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for their insightful comments and Mathieu Lacage for support on the *yans* simulator. Bobby Bhattacharjee and Bo Han were supported in part by NSF award NeTS-NBD 0626636. Part of this work was done when Bo Han was a summer intern at AT&T Labs – Research.

REFERENCES

[1] IEEE Std 802.11e 2005. Local and metropolitan area networks–Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications–Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements, November 2005.

[2] H. Zhu and I. Chlamtac. Performance Analysis for IEEE 802.11e EDCF Service Differentiation. *IEEE Transactions on Wireless Communications*, 4(4):1779–1788, July 2005.

[3] G. Bianchi. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *IEEE JSAC*, 18(3):535–547, March 2000.

[4] G. Bianchi, I. Tinnirello, and L. Scalia. Understanding 802.11e Contention-Based Prioritization Mechanisms and Their Coexistence with Legacy 802.11 Stations. *IEEE Network*, 19(4):28–34, July-August 2005.

[5] M. Lacage and T. R. Henderson. Yet Another Network Simulator. In *Proceedings of the 2006 ACM Workshop on ns-2: the IP Network Simulator*, October 2006.

[6] S. Shin and H. Schulzrinne. Experimental Measurement of the Capacity for VoIP Traffic in IEEE 802.11 WLANs. In *Proceedings of INFOCOM 2007*, pages 2018–2026, May 2007.

[7] ITU-T Recommendation G.107. The E-model, a Computational Model for Use in Transmission Planning, February 2003.

[8] S. Pilosof, R. Ramjee, D. Raz, Y. Shavitt, and P. Sinha. Understanding TCP fairness over Wireless LAN. In *Proceedings of INFOCOM 2003*, pages 863–872, March-April 2003.

[9] Multiband Atheros Driver for WiFi (MadWifi). Available at http://madwifi.org/.

[10] Iperf. Available at http://dast.nlanr.net/Projects/Iperf/.

[11] N. Thomos, N. V. Boulgouris, and M. G. Strintzis. Optimized Transmission of JPEG2000 Streams over Wireless Channels. *IEEE Transactions on Image Processing*, 15(1):54–67, January 2006.

[12] D. Li. Performance Evaluation of Video Streaming over Multi-hop Wireless Local Area Networks. Master's thesis, Department of Computer Science, University of Victoria, July 2008.

[13] Y. Xiao, F. H. Li, and B. Li. Bandwidth Sharing Schemes for Multimedia Traffic in the IEEE 802.11e Contention-Based WLANs. *IEEE Transactions on Mobile Computing*, 6(7):815–831, July 2007.

[14] G. Boggia, P. Camarda, L. A. Grieco, and S. Mascolo. Feedback-Based Control for Providing Real-Time Services with the 802.11e MAC. *IEEE/ACM Transactions on Networking*, 15(2):323–333, April 2007.

[15] Y. P. Fallah and H. Alnuweiri. Hybrid Polling and Contention Access Scheduling in IEEE 802.11e WLANs. *Journal of Parallel and Distributed Computing*, 67(2):242–256, February 2007.

[16] D. Skyrianoglou, N. Passas, and A. K. Salkintzis. ARROW: An Efficient Traffic Scheduling Algorithm for IEEE 802.11e HCCA. *IEEE Transactions on Wireless Communications*, 5(12):3558–3567, December 2006.

[17] C. Cicconetti, L. Lenzini, E. Mingozzi, and G. Stea. Design and Performance Analysis of the Real-Time HCCA Scheduler for IEEE 802.11e WLANs. *Computer Networks*, 51(9):2311–2325, June 2007.

[18] A. Banchs and L. Vollero. Throughput Analysis and Optimal Configuration of 802.11e EDCA. *Computer Networks*, 50(11):1749–1768, August 2006.

[19] C. Hu and J. C. Hou. A Novel Approach to Contention Control in IEEE 802.11e–Operated WLANs. In *Proceedings of INFOCOM 2007*, pages 1190–1198, May 2007.

[20] M. Nassiri, M. Heusse, and A. Duda. A Novel Access Method for Supporting Absolute and Proportional Priorities in 802.11 WLANs. In *Proceedings of INFOCOM 2008*, pages 1382–1390, April 2008.

[21] M. Heusse, F. Rousseau, R. Guillier, and A. Duda. Idle Sense: An Optimal Access Method for High Throughput and Fairness in Rate Diverse Wireless LANs. In *Proceedings of SIGCOMM 2005*, pages 121–132, August 2005.

[22] A. Banchs, A. Azcorra, C. Garcia, and R. Cuevas. Applications and Challenges of the 802.11e EDCA Mechanism: An Experimental Study. *IEEE Network*, 19(4):52–58, July-August 2005.

[23] I. Dangerfield, D. Malone, and D. J. Leith. Experimental Evaluation of 802.11e EDCA for Enhanced Voice over WLAN Performance. In *Proceedings of WiOpt 2006*, pages 1–7, April 2006.

[24] Y. Ge, J. C. Hou, and S. Choi. An Analytic Study of Tuning Systems Parameters in IEEE 802.11e Enhanced Distributed Channel Access. *Computer Networks*, 51(8):1955–1980, June 2007.

[25] M. Narbutt and M. Davis. The Capability of the EDCA Mechanism to Support Voice Traffic in a Mixed Voice/Data Transmission over 802.11e WLANs - An Experimental Investigation. In *Proceedings of LCN 2007*, pages 463–470, October 2007.

APPENDIX

We present a simplified formula of Zhu and Chlamtac [2], which is based on a two-dimensional Markov model. We consider two-node network, where STA1 has a higher channel access priority than STA2. STA i ($i = 1, 2$) has channel access parameters AIFSN$_i$, CWmin$_i$, and CWmax$_i$. We also define δ_{AIFS} = AIFS$_2$− AIFS$_1$. In this Markov model, the backoff state transition probability for STA λ_i can be expressed as follows:

$$\lambda_1 = 1, \lambda_2 = \begin{cases} \frac{E[BW_1]-\delta_{AIFS}}{E[BW_2]}, & E[BW_1] > \delta_{AIFS} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where $E[BW_i]$ is the average backoff duration of packets transmitted by station i. (See the paper by Zhu and Chlamtac [2] for more details.) Let p_i be the conditional collision probability of station i (the probability that a packet being transmitted on the channel collides with other packets). Let $b_{i,j,k}$ be the steady-state probability of a Markov state where station i is in the j-th backoff stage with the backoff counter k. By imposing the normalization condition, we have [2]:

$$\sum_{j=0}^{m_i} \sum_{k=0}^{W_{i,j}-1} b_{i,j,k} = b_{i,0,0} \sum_{j=0}^{m_i} p_i^j (1 + \frac{W_{i,j}-1}{2\lambda_i}) = 1 \quad (2)$$

where m_i is the maximum backoff stage and $W_{i,j}$ is the backoff window size at stage j for station i. The channel access probability τ_i for station i can be calculated as:

$$\tau_i = \sum_{j=0}^{m_i} b_{i,j,0} = \frac{1 - p_i^{m_i+1}}{i - p_i} b_{i,0,0}.$$

In this simplified model, we also have $p_1 = \tau_2, p_2 = \tau_1$.