A Novel MAC Protocol with Fast Collision Resolution for Wireless LANs

Younggoo Kwon, Yuguang Fang and Haniph Latchman
Department of Electrical and Computer Engineering
University of Florida
Gainesville, Florida 32611-6130
Email: ykwon@ufl.edu, fang@ece.ufl.edu, latchman@list.ufl.edu

***Abstract*— Design of efficient medium access control (MAC) protocols with both high throughput performance and high-degree of fairness performance is a major focus in distributed contention-based MAC protocol research. In this paper, we propose a novel and efficient contention-based MAC protocol for wireless local area networks, namely, the *Fast Collision Resolution* (FCR) algorithm. This algorithm is developed based on the following innovative ideas: to speed up the collision resolution, we actively redistribute the backoff timers for all active nodes; to reduce the average number of idle slots, we use smaller contention window sizes for nodes with successful packet transmissions and reduce the backoff timers exponentially fast when a fixed number of consecutive idle slots are detected. We show that the proposed FCR algorithm provides high throughput performance and low latency in wireless LANs. The extensive simulation studies show that the FCR algorithm could significantly improve the performance of the IEEE 802.11 MAC protocol if our efficient collision resolution algorithm is used and that the fairly scheduled FCR (FS-FCR) algorithm could simultaneously achieve high throughput performance and a high degree of fairness.**

I. Introduction

A good medium access control (MAC) protocol for wireless local area networks (LANs) should provide an efficient mechanism to share limited spectrum resources, together with simplicity of operation, fairness in serving all stations, and high throughput. The ideal case would be low delay under low network load and high throughput under high network load, although in reality it is usually difficult to satisfy both requirements. Therefore, a variety of MAC protocols have been developed to suit the different applications, where various tradeoff factors have been considered.

Medium access control algorithms in wireless LANs can be classified into two broad categories, namely, contention-based MAC algorithms and reservation-based MAC algorithms. Contention-based MAC algorithms are generally used in a distributed network architecture, and are suitable for bursty data traffic under low network load because of their low delay characteristics. Furthermore, the simplicity of implementation of contention-based MACs is also a desirable feature in wireless ad hoc networks where no infrastructure access points exist([2], [5], [17]). On the other hand, reservation-based MAC algorithms are mainly used by an access point in a centralized network architecture. It is well-known that reservation-based MAC algorithms can support desired QoS for real-time traffic with high degree of flexibility in providing services and work efficiently under heavy network load([8], [14]). Our focus in this paper is on distributed contention-based MAC protocols.

Distributed contention-based MAC protocol research in wireless networks started with ALOHA and slotted ALOHA in the 1970s. Later, MACA, MACAW, FAMA and DFWMAC were proposed by incorporating the carrier sense multiple access (CSMA) technique as well as the RTS and CTS handshaking mechanism for collision avoidance (CA) ([2], [9], [12] and references therein). The most popular contention-based wireless MAC protocol, CSMA/CA, has become the basis of the MAC protocol for the IEEE802.11 standard([17]). However, it is observed that if the number of active users increases, the throughput performance of the IEEE802.11 MAC protocol degrades significantly because of the excessively high collision rate. Many researchers have focused on analyzing and improving the performance of the IEEE802.11 MAC (see for example [3], [4], [5] and references therein).

To increase the throughput performance of a distributed contention-based MAC protocol, an efficient collision resolution algorithm is needed to reduce the overheads (such as packet collisions and idle backoff slots) in each contention cycle. To this end, many novel collision resolution algorithms have been proposed. For example, improved backoff algorithms can be developed to adjust the increasing and decreasing factors of the contention window size and the randomly chosen backoff values; the out-band busy-tone signaling is used to actively inform others of the busy channel status; and the contention information appended on the transmitted packets can also help in the collision resolution([2], [3], [11], [12]).

Although many innovative distributed contention-based MAC protocols have been proposed, very few MAC protocols satisfy simultaneously all desirable properties such as high throughput and good fairness while maintaining the simplicity of implementation in real wireless LANs. In this paper, we propose a new efficient distributed contention-based MAC algorithms, namely, the *Fast Collision Resolution* (FCR). We observe that the main deficiency of most distributed contention-based MAC algorithms comes from packet collisions and the wasted idle slots due to backoffs in each contention cycle. For example, in the IEEE 802.11 MAC protocol, when the number of active stations increases, there are too many stations backed off with small contention windows, hence

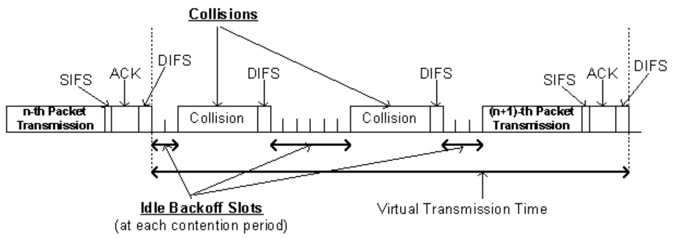

Fig. 1. Basic operations of CSMA/CA

many retransmission attempts will most likely collide again in the future, which would slow down the collision resolution. In this regard, the FCR algorithm attempts to resolve the collisions quickly by increasing the contention window sizes of both the colliding stations and the deferring stations in the contention resolution, i.e., we devise an algorithm so that all active stations will redistribute their backoff timers to avoid possible "future" collisions. To reduce the number of idle slots, the FCR algorithm gives a small idle backoff period for each station with successful packet transmission. Moreover, when a station detects a number of consecutive idle slots, it will start to reduce the backoff timer exponentially fast, compared to the linear decrease in backoff timer in the IEEE 802.11 MAC. We attempt to keep the proposed distributed contention-based MAC easily implementable in real wireless local area networks.

This paper is organized as follows. In the next section, we briefly describe the well-known IEEE 802.11 MAC protocol to set the stage for our proposed MAC protocols. Then we present, in Section III, the newly proposed fast collision resolution (FCR) algorithm and the fairly scheduled fast collision resolution (FS-FCR) algorithm. In Section IV, performance evaluations are presented, and in the final section, we present the conclusions.

II. IEEE 802.11 Medium Access Control

The most popular contention-based medium access control (MAC) protocol is the carrier sense multiple access/collision avoidance (CSMA/CA), which is widely used in the IEEE 802.11 LANs. The basic operations of the CSMA/CA algorithm are shown in Figure 1.

A packet transmission cycle consists of a successful packet transmission by a source station followed by an acknowledgment (ACK) from the destination station. General operations of the IEEE 802.11 MAC protocol are as follows (we only consider distributed coordination function (DCF) without RTS-CTS handshake for simplicity). If a station has a packet to transmit, it will check the medium status by using the carrier sensing mechanism. If the medium is idle, the transmission may proceed. If the medium is determined to be busy, the station will defer until the medium is determined to be idle for a DCF inter-frame space (DIFS) and the backoff procedure will be invoked. The station will set its backoff timer to a random backoff time based on the current contention window size (CW):

$$\text{Backoff Time (BT)} = \text{Random()} \times \text{aSlotTime} \qquad (1)$$

where Random() is an integer randomly chosen from a uniform distribution over the interval [0,CW-1].

After DIFS idle time, the station performs the backoff procedure using the carrier sensing mechanism to determine whether there is any activity during each backoff slot. If the medium is determined to be idle during a particular backoff slot, then the backoff procedure shall decrement its backoff time by a slot time ($BT_{new} = BT_{old} - aSlotTime$). If the medium is determined to be busy at any time during a slot, then the backoff procedure is suspended. After the medium is determined to be idle for DIFS period, the backoff procedure is allowed to resume. Transmission shall begin whenever the backoff timer reaches zero. After a source station transmits a packet to a destination station, if the source station receives an acknowledgment (ACK) without errors after the short inter-frame space (SIFS) idle period, the transmission procedure is determined to be successfully completed. In this case, the contention window (CW) for this source station shall be reset to the initial (minimum) value minCW. If the transmission is not successfully completed (i.e., the source station does not receive the ACK after SIFS), the contention window (CW) size shall be increased (in the IEEE 802.11 Direct Sequence Spread Spectrum (DSSS) specification, $CW = 2^{(n+5)} - 1, \quad retry\ counter\ n = 0, ..., 5$), beginning with the initial value minCW, up to the maximum value maxCW (in the IEEE 802.11 DSSS specification, minCW=31 and maxCW=1023). This process is called *binary exponential backoff (BEB)*, which resolves collisions in the contention cycle. More detailed operations can be found in ([17]).

III. Fast Collision Resolution

There are two major factors affecting the throughput performance in the IEEE 802.11 MAC protocol: transmission failures (we only consider failures due to packet collisions) and the idle slots due to backoff at each contention period, which are shown in Figure 1.

Under high traffic load (i.e., all M stations always have packets to transmit) and under some ergodicity assumption, we can obtain the following expression for the throughput (for example, based on Figure 1, we can examine one transmission cycle)([3], [5]):

$$\rho = \frac{\bar{m}}{E[N_c](E[B_c] \cdot t_s + \bar{m} + DIFS) + (E[B_c] \cdot t_s + \bar{m} + SIFS + ACK + DIFS)} \qquad (2)$$

where $E[N_c]$ is the average number of collisions in a virtual transmission time (or a virtual transmission cycle), $E[B_c]$ is the average number of idle slots resulting from backoff for each contention period, t_s is the length of a slot (i.e., aSlotTime), and $\bar{m}$ is the average packet length.

From this result, we observe that the best scenario in Figure 1 would be the following: a successful packet transmission must be followed by another packet transmission without any overheads, in which case, $E[N_c] = 0, E[B_c] = 0$, the throughput would be

$$\rho_{best} = \frac{\bar{m}}{(\bar{m} + SIFS + ACK + DIFS)} \tag{3}$$

This can be achieved only when a perfect scheduling is provided: in such a scenario, each station shall have the probability of packet transmission, $p_{trans}(i)$, at each contention period as follows:

$$p_{trans}(i) = \begin{cases} 1 & \text{if station } i \text{ transmits its packet at current} \\ & \text{contention period} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

This implies that if the current packet transmission right is assigned to station i, then only station i will transmit and all other stations will defer their packet transmissions.

Suppose that under some random backoff schemes, we could assume that the backoff timer is chosen randomly, then the probability of packet transmission for station i during the current contention period would depend on the backoff timer:

$$p_{trans}(i) = \frac{1}{(B_i + 1)} \tag{5}$$

where B_i is the backoff timer of station i. This means that if station i has the backoff timer 0 (i.e., $B_i = 0$), then its backoff time is 0 (i.e., $BT = B_i \times aSlotTime = 0$) and station i will transmit a packet immediately. Therefore, this can be interpreted to imply that station i has the probability of packet transmission of 1 at current contention period. If station i has the backoff timer ∞, then its backoff time is also ∞, which can be interpreted to mean station i has the probability of packet transmission of 0 at current contention period. From this discussion, (4) can be converted to (6):

$$B_i = \begin{cases} 0 & \text{if station } i \text{ transmits its packet at current} \\ & \text{contention period} \\ \infty & \text{otherwise} \end{cases} \tag{6}$$

Thus, we conclude that if we could develop a contention-based MAC algorithm, which assigns a backoff timer 0 to the station in transmission while assigning all other stations' backoff timers ∞ for each contention period, then we could achieve the perfect scheduling, leading to the maximum throughput. Unfortunately, such a contention-based MAC algorithm does not exist in practice. However, this does provide us the basic idea how to improve the throughput performance in the MAC protocol design. We can use the operational characteristics of the perfect scheduling to design more efficient contention-based MAC algorithm. One way to do so is to design a MAC protocol to approximate the behavior of perfect scheduling.

From (4) and (6), we can summarize the operational characteristics which can be used to achieve high throughput performance in contention-based MAC algorithms as follows:

- *Small random backoff timer for the station which has successfully transmitted a packet at current contention cycle*: This will decrease the average number of idle slots for each contention period, $E[B_c]$ in (2).
- *Large random backoff timer for stations that are deferring their packet transmissions at current contention period*: A deferring station means a station which has nonzero backoff timers. Large random backoff timers for deferring stations will decrease the collision probability at subsequent contention periods (and avoid future collisions more effectively).
- *Fast change of backoff timer for a station according to its current state: transmitting or deferring*: When a station transmits a packet successfully, its next backoff timer should be set small. The net effect of this operation is that whenever a station seizes the channel, it will utilize the medium as long as it could, this way, the medium will be spent more on useful transmissions, thus the average number of idle slots can be reduced. When a station is deferring, then its random backoff timers should be large to avoid the obvious collisions with the station with successful packet transmissions, and the net effect is that all deferring stations will give the successful station more time to finish the back-logged packets. Moreover, whenever a deferring node detects a start of busy period (either for medium contention or for a successful packet transmission), it will expand its contention window size and resets its backoff timer (similar to the backoff procedure). The net effect of this operation is to actively redistribute the backoff timers for those deferring nodes to avoid "future" collisions. When a deferring station detects the medium is idle for a fixed number of slots, it would conclude that no other stations are transmitting. To reduce the average number of idle slots ($E(B_c)$), the deferring station will reduce the backoff timers exponentially fast after every idle slot detected. Intuitively, we can see that such fast change in backoff timers will increase the probability of successful packet transmissions, consequently, the average number of collisions in a virtual transmission time, $E[N_c]$ in (2), will be decreased.

A. Fast Collision Resolution (FCR) Algorithm

As we mentioned before, the major deficiency of the IEEE 802.11 MAC protocol comes from the slow collision resolution as the number of active stations increases. An active station can be in two modes at each contention period, namely, the transmitting mode when it wins a contention and the deferring mode when it loses a contention. When a station transmits a packet, the outcome is either one of the two cases: a successful packet transmission or a collision. Therefore, a station will be in one of the following three states at each contention period: a successful packet transmission state, a collision state, and a deferring state. In most distributed contention-based MAC algorithms, there is no change in the contention window size for the deferring stations, and the backoff timer will decrease by one slot whenever an idle slot is detected. In the proposed fast collision resolution (FCR) algorithm, we will change the contention window size for the deferring stations and regenerate the backoff timers for all potential transmitting stations to actively avoid "future" potential collisions, in this way, we can resolve possible packet collisions quickly. More importantly, the proposed algorithm preserves the simplicity for implementation like the IEEE 802.11 MAC.

The FCR algorithm has the following characteristics:

1) Use much smaller initial (minimum) contention window size $minCW$ than the IEEE 802.11 MAC;
2) Use much larger maximum contention window size $maxCW$ than the IEEE 802.11 MAC;
3) Increase the contention window size of a station when it is in either collision state and deferring state;
4) Reduce the backoff timers exponentially fast when a prefixed number of consecutive idle slots are detected.

Items 1) and 4) attempt to reduce the average number of idle backoff slots for each contention period ($E[B_c]$) in (2). Items 2) and 3) are used to quickly increase the backoff timers, hence quickly decrease the probability of collisions. In item 3), the FCR algorithm has the major difference from other contention-based MAC protocols such as the IEEE 802.11 MAC. In the IEEE 802.11 MAC, the contention window size of a station is increased only when it experiences a transmission failure (i.e., a collision). In the FCR algorithm, the contention window size of a station will increase not only when it experiences a collision but also when it is in the deferring mode and senses the start of a new busy period. Therefore, all stations which have packets to transmit (including those which are deferring due to backoff) will change their contention window sizes at each contention period in the FCR algorithm.

The detailed FCR algorithm is described as follows according to the state a station is in:

1) *Backoff Procedure*: All active stations will monitor the medium. If a station senses the medium idle for a slot, then it will decrement its backoff time (BT) by a slot time, i.e., $BT_{new} = BT_{old} - aSlotTime$ (or the backoff timer is decreased by one unit in terms of slots). When its backoff timer reaches to zero, the station will transmit a packet. If there are $[(minCW + 1) \times 2 - 1]$ consecutive idle slots being detected, its backoff timer should be decreased much faster (say, exponentially fast), i.e., $BT_{new} = BT_{old}/2$ ($if\ BT_{new} < aSlotTime,\ then\ BT_{new} = 0$). For example, if a station has the backoff timer 2047, its backoff time is $BT = 2047 \times aSlotTime$, which will be decreased by a slot time at each idle slot until the backoff timer reaches 2040 (we assume that $[(minCW + 1) \times 2 - 1] = 7$ or $minCW = 3$). After that, if the idle slots continue, the backoff timer will be decreased by one half, i.e., $BT_{new} = BT_{old}/2$ at each additional idle slot until either it reaches to zero or it senses a non-idle slot, whichever comes first. As an illustration, after 7 idle slots, we will have $BT = 1020 \times aSlotTime$ on the 8*th* idle slot, $BT = 510 \times aSlotTime$ on the 9*th* idle slot, $BT = 255 \times aSlotTime$ on the 10*th* idle slot, and so on until it either reaches zero or detects a non-idle slot. Therefore, the wasted idle backoff time is guaranteed to be less than or equal to $18 \times aSlotTime$ for above scenario. The net effect is that the unnecessary wasted idle backoff time will be reduced when a station runs out of packets for transmission.
2) *Transmission Failure (Packet Collision)*: If a station notices that its packet transmission has failed possibly due to packet collision (i.e., it fails to receive an acknowledgment from the intended receiving station), the contention window size of the station will be increased and a random backoff time (BT) will be chosen, i.e., $CW = \min(maxCW, CW \times 2)$, $BT = uniform(0, CW - 1) \times aSlotTime$, where $uniform(a, b)$ indicates a number randomly drawn from the uniform distribution between a and b and CW is the current contention window size.
3) *Successful Packet Transmission*: If a station has finished a successful packet transmission, then its contention window size will be reduced to the initial (minimum) contention window size $minCW$ and a random backoff time (BT) value will be chosen accordingly, i.e., $CW = minCW$, $BT = uniform(0, CW - 1) \times aSlotTime$.
4) *Deferring State*: For a station which is in deferring state, whenever it detects the start of a new busy period, which indicates either a collision or a packet transmission in the medium, the station will increase its contention window size and pick a new random backoff time (BT) as follows: $CW = \min(maxCW, CW \times 2)$, $BT = uniform(0, CW - 1) \times aSlotTime$.

0	1	2	3	4	5	6	7	8	9	**Station Number**
1(7)	3(7)	2(7)	7(7)	2(7)	6(7)	3(7)	4(7)	1(7)	6(7)	Contention Begins
0(7) 8(15)	2(7)	1(7)	6(7)	1(7)	5(7)	2(7)	3(7)	**0**(7) 14(15)	5(7)	Collision on station 0 & 8
7(15)	1(7)	**0**(7) 4(15)	5(7)	**0**(7) 9(15)	4(7)	1(7)	2(7)	13(15)	4(7)	Collision on station 2 & 4
6(15)	**0**(7) 10(15)	3(15)	4(7)	8(15)	3(7)	**0**(7) 5(15)	1(7)	12(15)	3(7)	Collision on station 1 & 6
5(15)	9(15)	2(15)	3(7)	7(15)	2(7)	4(15)	**0**(7) 3(7)	11(15)	2(7)	Successful Packet Transmission on station 7

*** Each item indicate: Backoff Timer B_i (Contention Window Size)**

TABLE I

EXAMPLE OF IEEE 802.11 MAC WITH BINARY EXPONENTIAL BACKOFF

In the FCR algorithm, the station that has successfully transmitted a packet will have the minimum contention window size and smaller backoff timer, hence it will have a higher probability to gain access of the medium, while other stations have relatively larger contention window size and larger backoff timers. After a number of successful packet transmissions for one station, another station may win a contention and this new station will then have higher probability to gain access of the medium for a period of time.

To elaborate the operations of the FCR algorithm, we use some examples to illustrate the major difference between the IEEE 802.11 MAC and FCR algorithm. Table I shows an example of the IEEE 802.11 MAC operations with the contention window size $CW = 2^{(n+3)} - 1,\ \ retry\ counter\ n = 0, ..., 7$ (i.e., minCW=7 and maxCW=1023). In this example, there are 10 active stations contending for the use of the medium based on the IEEE 802.11 MAC. When the contention begins (i.e., the medium is determined to be idle for DIFS period by the carrier sensing mechanism), each station

0	1	2	3	4	5	6	7	8	9	Station Number

1(3)
1(7)
	0(3)
	3(7)
		2(3)
		2(7)
			1(3)
			7(7)
				2(3)
				2(7)
					2(3)
					6(7)
						3(3)
						3(7)
							3(3)
							4(7)
								1(3)
								1(7)
									0(3)
									6(7)
										Collision on 1 & 9

0(7)
8(15)
	2(7)
	10(15)
		1(7)
		2(15)
			6(7)
			1(15)
				1(7)
				12(15)
					5(7)
					4(15)
						2(7)
						15(15)
							3(7)
							6(15)
								0(7)
								14(15)
									5(7)
									3(15)
										Collision on 0 & 8

7(15)
22(31)
	9(15)
	18(31)
		1(15)
		28(31)
			0(15)
			1(3)
				11(15)
				5(31)
					3(15)
					17(31)
						14(15)
						11(31)
							5(15)
							9(31)
								13(15)
								14(31)
									2(15)
									23(31)
										Success on 3

21(31)
40(63)
	17(31)
	9(63)
		27(31)
		38(63)
			0(3)
			3(3)
				4(31)
				58(63)
					16(31)
					24(63)
						10(31)
						17(63)
							8(31)
							20(63)
								13(31)
								44(63)
									22(31)
									1(63)
										Success on 3

39(63)
100(127)
	8(64)
	55(127)
		37(63)
		29(127)
			2(3)
			5(7)
				57(63)
				111(127)
					23(63)
					46(127)
						16(63)
						81(127)
							19(63)
							30(127)
								43(63)
								9(127)
									0(63)
									1(3)
										Success on 9

99(127)
67(255)
	54(127)
	29(255)
		28(127)
		189(255)
			4(7)
			11(15)
				110(127)
				55(255)
					45(127)
					210(255)
						80(127)
						160(255)
							29(127)
							240(255)
								8(127)
								120(255)
									0(3)
									2(3)
										Success on 9

* Each item indicate: Backoff Timer B_i (Contention Window Size)

TABLE II

EXAMPLE OF FAST COLLISION RESOLUTION ALGORITHM

performs the backoff procedure with its random backoff time (BT) determined from the initial contention window range [0, 7] (hence $BT = uniform[0,\ 7] \times aSlotTime$). When a station detects the current slot idle, it will decrement its backoff time by a slot time $BT_{new} = BT_{old} - aSlotTime$ (i.e., the backoff timer is decreased by one unit). After one idle slot, the backoff timers of stations 0 and 8 reach zero, thus in the following slot, both station 0 and station 8 will transmit their packets at the same time and a collision will occur. The backoff procedures of all deferring stations are suspended and will resume after the medium is determined to be idle for DIFS period (i.e., next contention period). After stations 0 and 8 notice that their packet transmissions fail, their contention window sizes will be increased to 15 and their backoff timers will be chosen in the range of [0, 15] randomly. When a new DIFS period is detected, stations 2 and 4 transmit packets after one idle slot and a collision occurs. Stations 1 and 6 transmit packets and a collision occurs in the following contention period. After that, when the next DIFS period is detected, station 7 has a successful packet transmission. In the whole contention cycle (the time period starting with the end of a successful packet transmission and ending with the start of the next successful packet transmission), there have been three consecutive collisions before one successful packet transmission. We observe in Table I that most contention window sizes chosen for the backoffs are not big enough to avoid future packet collisions. Since the IEEE 802.11 MAC cannot provide the proper contention window size as the number of active stations increases, collisions are not resolved quickly, which leads to poor throughput performance.

Table II shows an example for the FCR algorithm with the contention window size $CW = 2^{(n+2)} - 1,\ \ retry\ counter\ n = 0, ..., 9$ (i.e., minCW=3 and maxCW=2047). In Table II, stations 1 and 9 collide in the first contention period. Stations 1 and 9 then increase their contention window sizes to 7 and pick up their backoff timers in the range of [0, 7] randomly. All deferring stations also increase their contention window sizes to 7 and pick up the new backoff timers in the range of [0, 7] randomly. In the second contention period, stations 0 and 8 collide and will repeat the same procedure. In the third contention period, station 3 transmits a packet successfully. We observe in Table II that most contention window sizes of the deferring stations are increased quickly, so the FCR algorithm resolves the contentions very quickly, which results in significantly lower collision probability during each contention period in the future.

In Table I and Table II, we can clearly see the major differences in operations between the IEEE 802.11 MAC and the FCR algorithm. To put it briefly, the high throughput of the FCR algorithm comes from: the small backoff time for the station that transmits a packet at current contention period (this reduces the wasted idle slots), the large backoff time for the stations which are deferred for packet transmissions (this reduces the collision probability), and faster change of backoff timers according to the current state: transmitting or deferring. This means that the FCR algorithm satisfies well the required condition for high throughput performance which is shown in (6).

B. Fairly Scheduled Fast Collision Resolution Algorithm (FS-FCR)

Fairness is an important issue in MAC protocol design for wireless local area networks. The IEEE 802.11 MAC exhibits inherent unfairness characteristics([21], [25], [27]). FCR makes things worse because the deferring nodes will tend to defer their transmissions further by expanding their contention windows upon detecting any start of a busy period before the backoff timers expire. However, with proper provisioning in the FCR algorithm, we can address the fairness issue while maintaining high throughput performance of FCR algorithm. The idea is to modify the self-clocked fair queueing (SCFQ) algorithm([13]) and incorporate it into FCR algorithm. We combine these two algorithms and dynamically assign the successive transmission period of the FCR algorithm by using the modified SCFQ algorithm. We call this new algorithm the fairly scheduled FCR (FS-FCR) algorithm. The SCFQ algorithm has been used to address the fairness issue for IEEE 802.11 WLANs by Vaidya et. al. ([27]). While the approach proposed by Vaidya et. al. is packet-by-packet based and controls the backoff timers, our approach is based on multiple successive packet transmissions (i.e., dynamically controlling the maximum successive transmission period). The basic operations of the fairly scheduled FCR (FS-FCR) algorithm are described as follows:

1) Each arriving packet to the queue of a station is tagged with a service tag before it is placed in the queue.
2) When the k-th packet of station i, P_i^k, arrives at the queue of the station, a service tag F_i^k is assigned as follows:

$$F_i^k = \max\{v(a_i^k), F_i^{k-1}\} + \frac{L_i^k}{\phi_i}$$

where $v(a_i^k)$ is the virtual time at the time instance of a_i^k, a_i^k is the real time when packet P_i^k arrives, L_i^k is the size of packet P_i^k, and ϕ_i is the weight of flow i.
3) The virtual time $v(t)$ is updated whenever there is a successful packet transmission. The virtual time is set to the service tag of that packet just successfully transmitted. The virtual time $v(t)$ *approximately* represents

the normalized fair amount of packet transmissions that each station should have performed (because a packet with the smallest service tag shall not be guaranteed to be served first in distributed systems). Once a busy period is over, i.e., when all stations do not have any packets to transmit, the virtual time is reset to zero.

4) Whenever a new station acquires the medium for packet transmissions, the maximum successive transmission limit (i.e., the successive transmission time period) of the station i, $T_{PkTrans,i}$, is determined by the difference between the virtual time $v(t)$ and the service tag F_i^k at the front of the packet flow at station i. If the service tag of station i is much smaller than the current virtual time, then its maximum successive transmission limit is assigned large enough to reduce the discrepancy between the current virtual time and the service tag at the front of flow i. If the service tag of station i is close to or larger than the current virtual time, then its maximum successive transmission limit is assigned to the minimum or small value to avoid increasing the discrepancy between the current virtual time and the service tag at the front of the packet flow of station i.

   An example for assigning the maximum successive transmission limit is:

$$T_{PkTrans,i} = g[v(t) - F_i^k]$$

   where

$$g[x] = \begin{cases} 20 \times t_s, & x \le (-1000 \times t_s) \\ 40 \times t_s, & (-1000 \times t_s) < x \le (-500 \times t_s) \\ 60 \times t_s, & (-500 \times t_s) < x \le (0 \times t_s) \\ 400 \times t_s, & (0 \times t_s) < x \le (500 \times t_s) \\ 1000 \times t_s, & (500 \times t_s) < x \le (1000 \times t_s) \\ 2000 \times t_s, & (1000 \times t_s) < x \le (2000 \times t_s) \\ 3000 \times t_s, & (2000 \times t_s) < x \le (3000 \times t_s) \\ 4000 \times t_s, & (3000 \times t_s) < x \le (4000 \times t_s) \\ 5000 \times t_s, & x > (4000 \times t_s) \end{cases}$$

   where t_s is the $aSlotTime$.

5) The FS-FCR algorithm uses the same operations of the FCR algorithm, except that, if a station reaches its maximum successive transmission limit in its packet transmission period, the station will set its contention window size to the maximum value of $maxCW$. This will give other stations higher probabilities to transmit their packets at next contention period. Since the wasted idle slots are limited less than 18 (see Section III.A), the overheads caused by the idle backoff slots will be small even after a station has finished its packet transmission period and does not have any packets to transmit.

IV. Performance Evaluation

In this section, we present the simulation studies for the proposed fast collision resolution (FCR) algorithms and the IEEE 802.11 MAC protocol using DSSS specification. The parameters used in the simulations are shown in Table III, which are based on the IEEE 802.11 network configurations([17]).

We assume that the best-effort data packets are always available at all stations. In the simulations, the packet lengths for the best-effort data packets are geometrically distributed with parameter q([5]):

$$P[PacketLength = i\ slots] = q^{i-1}(1-q), \quad i \ge 1.$$

Thus, the average transmission time for a packet (the average packet length) is given by:

$$\bar{m} = t_s/(1-q) \qquad (\mu sec)$$

where t_s is the slot time, i.e., $t_s = aSlotTime$.

Parameter	Value
SIFS	10 μsec
DIFS	50 μsec
A slot time	20 μsec
aPreambleLength	144 bits
aPLCPHeaderLength	48 bits
Bit rate	2 Mbps

TABLE III
Network Configurations

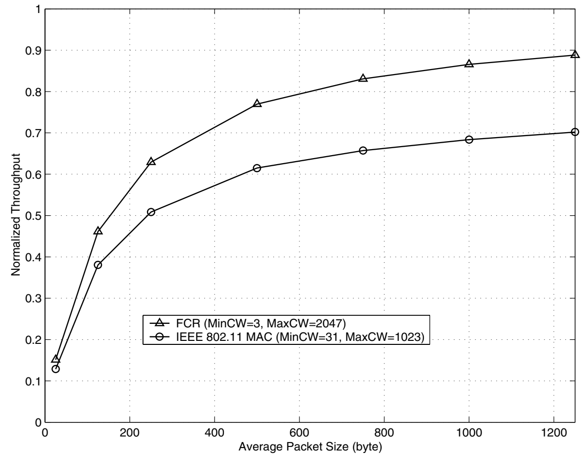

Fig. 2. Throughput for 10 BE data stations wireless LAN

A. Simulation Results for FCR Algorithm

Figures 2, 3 and 4 show the throughput results of the IEEE 802.11 MAC and FCR for 10, 50, and 100 contending stations, where the average transmission time for a packet (i.e., the average packet length) changes from 100 μsec (25 bytes) to 5000 μsec (1250 bytes). The IEEE 802.11 MAC algorithm shows very poor throughput performance as the number of stations increases. The main reason is that the probability of collisions becomes higher as the number of stations becomes larger. In the FCR algorithm, all stations except the one with successful packet transmission will increase their contention window size whenever the system has either a successful packet transmission or has a collision. This means all stations can quickly obtain the proper contention window size to prevent future collisions, consequently the probability of collisions will be decreased to quite small values. At the same time, a station with a successful packet transmission has the minimum contention window size of 3, which is much

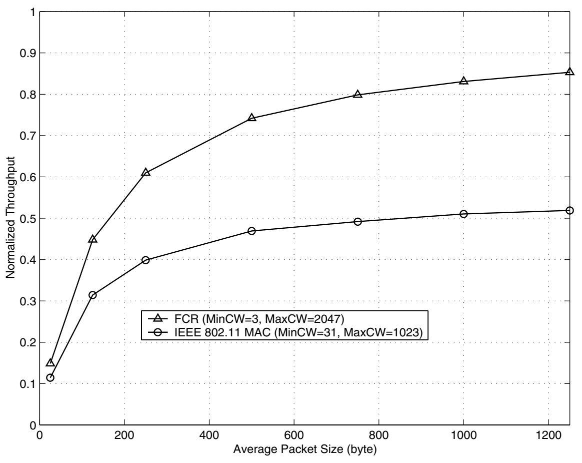

Fig. 3. Throughput for 50 BE data stations wireless LAN

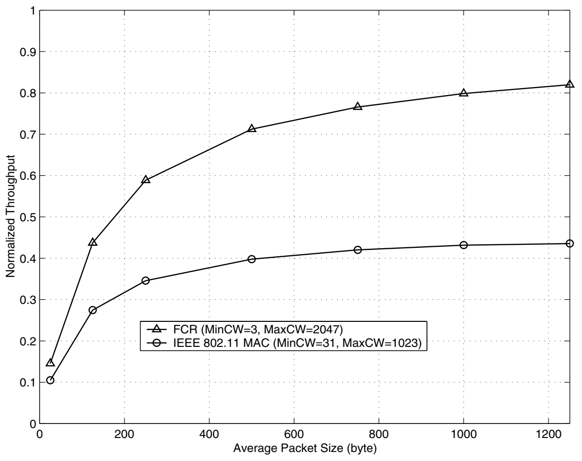

Fig. 4. Throughput for 100 BE data stations wireless LAN

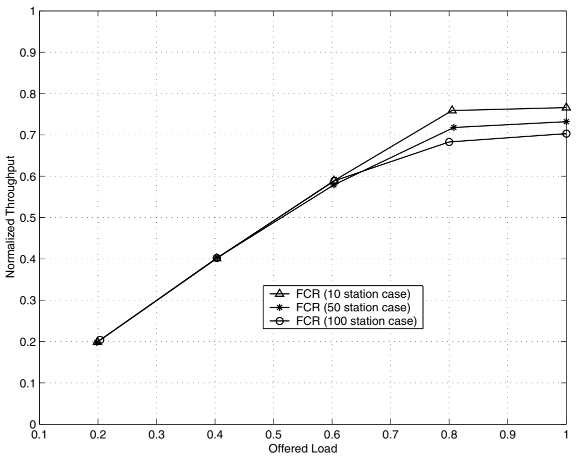

Fig. 5. Throughput vs. offered load for FCR

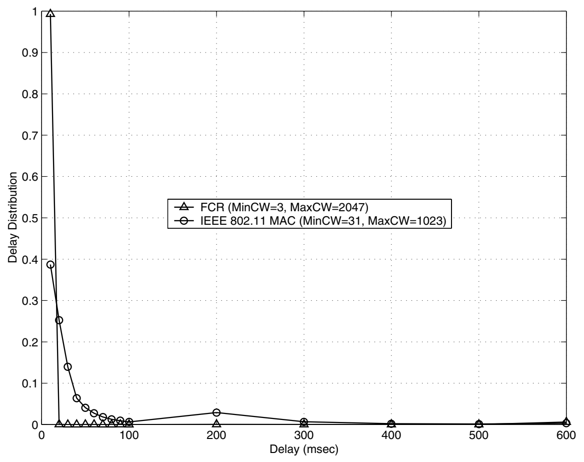

Fig. 6. Delay distribution for 10 stations wireless LAN

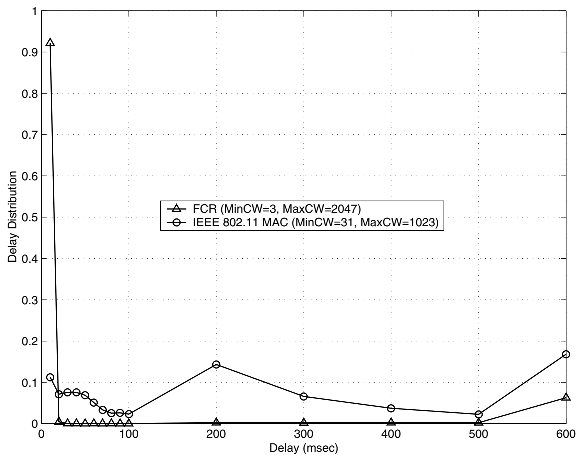

Fig. 7. Delay distribution for 100 stations wireless LAN

smaller than the minimum contention window size in the IEEE 802.11 MAC algorithm (minCW=31). This will reduce the wasted medium idle time to a much smaller value when compared to the IEEE 802.11 MAC algorithm. In Figures 2, 3 and 4, we can see that the FCR algorithm significantly improve the throughput performance over the IEEE 802.11 MAC algorithm. Moreover, the throughput performance of the FCR algorithm are not severely degraded as the number of stations increases because of the highly efficient collision resolution strategy.

Figure 5 shows the throughput vs. offered load of FCR algorithm for 10, 50, 100 stations wireless LAN with the average transmission time for a packet (i.e., the average packet length) of 2000 μsec (500 bytes). We use a traffic generator with Poisson distribution to provide each offered load in this simulation. From Figure 5, we can see that the FCR algorithm also performs very efficiently under light load conditions while providing high throughput as network load increases, and the number of stations hardly affects the performance of the FCR algorithm.

We also analyze for the packet delay for the IEEE 802.11

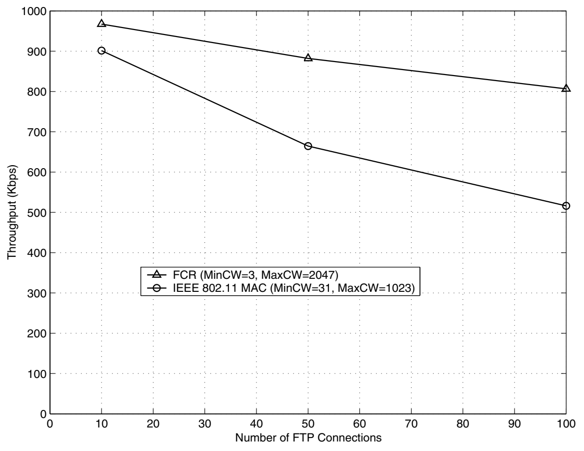

Fig. 8. TCP throughput results for FCR algorithm for FTP traffic

MAC algorithm and the FCR algorithm with the average transmission time for a packet (i.e., the average packet length) of 2000 μsec (500 bytes). The packet delay means the time period from the time when a packet arrives at the front of packet flow of a station to the time it is successfully transmitted to the intended receiving station. The packet transmission time is not included, i.e., the packet delay is the time delay completely caused by the efficiency of each MAC algorithm. Figures 6 and 7 show the packet delay distributions for the IEEE 802.11 MAC algorithm and the FCR algorithm for 10 and 100 stations wireless LANs. We have not imposed any limitation on the number of retries in this simulation for simplicity. In Figure 6, the FCR algorithm transmits 99% of all packets successfully within 10 msec while the remaining 1% packets spread over 10 msec to over 600 msec in delay. However, the IEEE 802.11 MAC transmits 39% packets within 10 msec, 25% packets in the range from 10 msec to 20 msec, 13% packets in the range from 20msec to 30 msec, and so on. In Figure 7, the FCR algorithm transmits 92% of all packets successfully within 10 msec, while the IEEE 802.11 MAC transmits only 11% packets within 10 msec, 8% packets in the range from 10 msec to 20 msec, 8.5% packets in the range from 20 msec to 30 msec, and so on. In the simulation results for the packet delay, it is clear that the FCR algorithm transmits most packets successfully within pretty short time, while the IEEE 802.11 MAC transmits packets in much longer time due to collisions, which indeed shows that the FCR algorithm does resolve collision much faster than the IEEE 802.11 MAC algorithm does.

We also ran extensive simulations on TCP connections over the FCR algorithm and the IEEE 802.11 MAC. Figure 8 shows one result for the TCP performance of the FCR algorithm obtained from the Glomosim network simulator([1]). The throughput result is shown for various numbers of FTP connections(10, 50, and 100). The channel bandwidth is 2 Mbps and all FTP connections continuously send data from source stations to destination stations with packets of 1460 bytes, and the simulation time is 100 seconds. In Figure 8, we observe

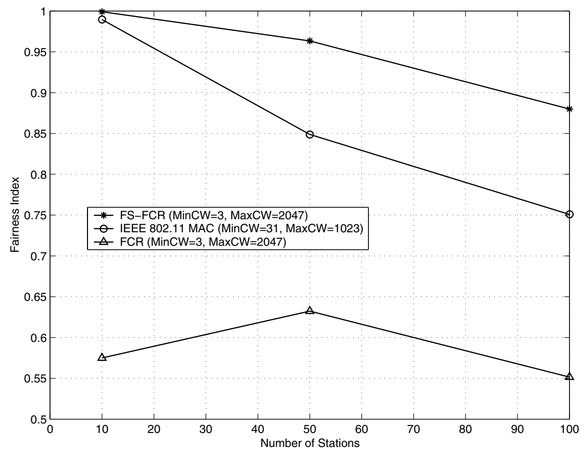

Fig. 9. Fair index for 10 sec simulation

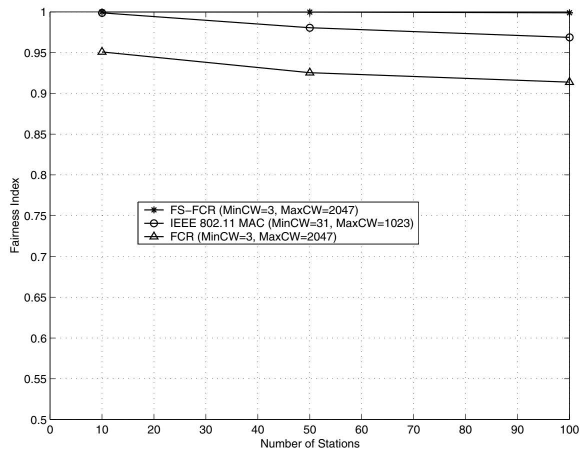

Fig. 10. Fair index for 100 sec simulation

that the FCR algorithm improves the TCP performance 10-50 % compared to the IEEE 802.11 MAC does as the number of FTP connections increases, which implies that the efficient collision resolution scheme of the FCR algorithm can also significantly improve the performance at higher layers.

B. Simulation Results for FS-FCR Algorithm

In the FS-FCR algorithm, the maximum transmission period ($T_{PkTrans}$) is controlled by the modified SCFQ algorithm to provide a high degree of fairness. Figures 9 and 10 show the results of the fairness index of FS-FCR, FCR, and IEEE802.11 MAC algorithms. The average transmission time for a packet (i.e., the average packet length) of 2000 μsec (500 bytes) is used and the simulations are run for 10 and 100 seconds. We use the fairness index defined by Jain([19]) to evaluate the degree of fairness for each algorithm. This fairness index is defined as

$$FairnessIndex = \frac{(\sum_i T_i/\phi_i)^2}{n \cdot \sum_i (T_i/\phi_i)^2} \tag{7}$$

where n is the number of flows, T_i is the throughput of flow i, ϕ_i is the weight of the flow i (we assume all stations

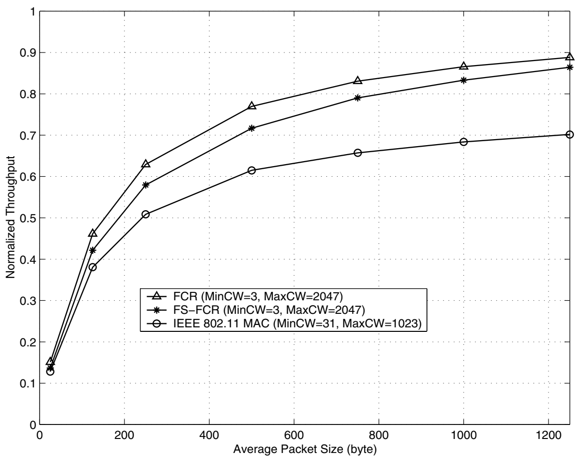

Fig. 11. Throughput for 10 BE data station wireless LAN

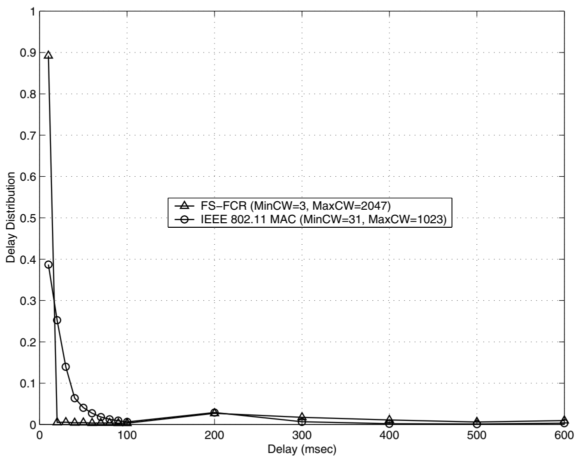

Fig. 13. Delay Distribution for 10 stations wireless LAN

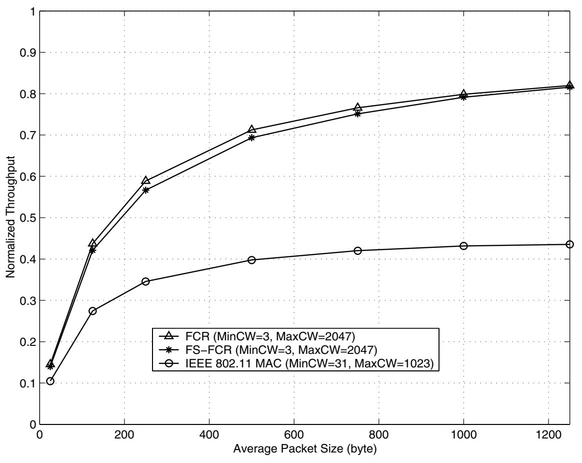

Fig. 12. Throughput for 100 BE data stations wireless LAN

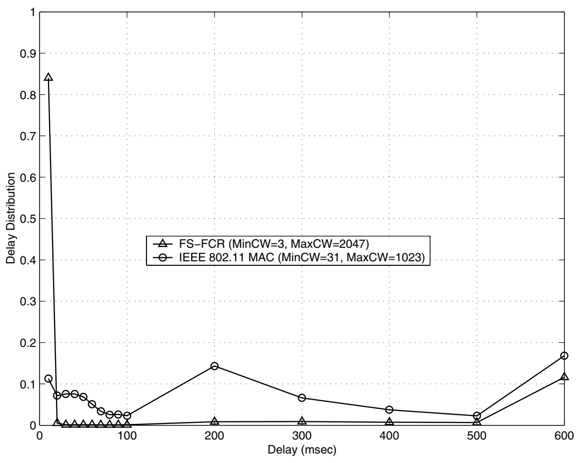

Fig. 14. Delay Distribution for 100 stations wireless LAN

have the same weight in simulations). From Cauchy-Schwartz inequality, we obtain $FairnessIndex \leq 1$, and the equality holds if and only if all T_i/ϕ_i $(i = 1, 2, \ldots, n)$ are equal. Thus, the intuition behind this index is that the higher the fairness index (i.e., closer to 1), the better in terms of fairness.

From Figures 9 and 10, we observe that the FS-FCR shows the best fairness performance. For 10 seconds simulations, the FS-FCR algorithm provides a high degree of fairness for 10 stations wireless LANs, and the fairness index is degraded slightly as the number of stations increases to 50 and 100. The FCR algorithm shows poor fairness performance, which is worse than the IEEE 802.11 MAC algorithm in the 10 seconds simulations as we expect. In Figure 10, the fairness performance results of all algorithms are improved because the simulation time is long enough (100 seconds) to give sufficient opportunities for all stations to transmit. According to the simulation results for fairness index, we can conclude that the FS-FCR algorithm significantly improves the fairness performance of the IEEE 802.11 MAC and the FCR algorithm.

Figures 11 and 12 present the throughput results for the three protocols (FCR, FS-FCR, IEEE 802.11 MAC) for 10 and 100 contending stations wireless LANs, where the average transmission time for a packet (i.e., the average packet length) changes from 100 μsec (25 bytes) to 5000 μsec (1250 bytes), and the simulation time is 100 seconds. We can see that the consideration of the SCFQ algorithm into the FCR algorithm causes only slight throughput degradation. Figure 11 shows that the throughput for the FS-FCR algorithm is much higher than that for the IEEE 802.11 MAC algorithm, and yet it is pretty close to that for the FCR algorithm. This is much more evident for 100 stations case as shown in Figure 12.

Figures 13 and 14 show the packet delay distribution for the IEEE 802.11 MAC algorithm and the FS-FCR algorithm for 10 and 100 stations wireless LANs with the average transmission time for a packet (i.e., the average packet length) of 2000 μsec (500 bytes). In Figure 13, the FS-FCR algorithm delivers 90% of all packets within 10 msec and the remaining 10% packets are delivered in the delay range from 10 msec to over 600 msec. However, the IEEE 802.11 MAC delivers only 39% packets within 10 msec, 25% packets in the range of [10 msec - 20 msec], 13% packets in the range of [20 msec -

30 msec]. In Figure 14, the FS-FCR algorithm delivers 85% of all packets within 10 msec packet delay, while the IEEE 802.11 MAC has much longer packet delay.

From the simulation study, we observe that the FS-FCR algorithm achieves much higher degree of fairness than the IEEE 802.11 MAC algorithm while still preserving the high throughput. The good performance of the FS-FCR algorithm comes from the following factors: efficient collision resolution algorithm of the FCR algorithm and fair scheduling algorithm (SCFQ algorithm).

V. Conclusions

In this paper, we propose a new contention-based medium access control algorithm. The FCR algorithm can achieve high throughput performance while preserving the implementation simplicity in wireless local area networks. In the FCR algorithm, each station changes the contention window size upon both successful packet transmissions and collisions (i.e., upon detecting a start of busy period) for all active stations in order to redistribute the backoff timers to actively avoid potential future collisions. Due to this operation, each station can more quickly resolve collisions when there are a large number of active stations in the wireless LANs. Other ideas we incorporate in the FCR are to use much smaller minimum contention window size comparing to IEEE 802.11 MAC and fast decreasing backoff timers after detecting a fixed number of idle slots. These changes reduce the average number of idle slots in each contention period, which contributes to the throughput improvement. To provide a high degree of fairness, we propose the fairly scheduled FCR algorithm, which dynamically assigns the successive transmission period of the FCR algorithm. Extensive simulation studies for throughput, delay distribution and fairness have demonstrated that the FCR algorithm gives significant throughput improvement compared to that for the IEEE802.11 MAC algorithm and the fairly scheduled FCR algorithm achieves both high throughput and high-degree of fairness simultaneously.

Acknowledgment

This work was supported in part by the ONR under Young Investigator Award N000140210464 and under grant N000140210554, and the NSF Faculty Early Career Development Award under grant ANI-0093241.

References

[1] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla, "GloMoSim: A Scalable Network Simulation Environment," UCLA Computer Science Department Technical Report 990027, May 1999.

[2] V. Bharghavan , "MACAW: A Media Access Protocol for Wireless LAN's," *SIGCOMM'94*, pp.212-225, London, England, Aug. 1994.

[3] V. Bharghvan, "Performance evaluation of algorithms for wireless medium access," *IEEE International Computer Performance and Dependability Symposium IPDS'98*, pp.142-149, 1998.

[4] G. Bianchi, "Performance Analysis of the IEEE802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, Vol.18, No.3, PP.535-547 Mar. 2000.

[5] F. Cali, M. Conti, and E. Gregori, "Dynamic Tuning of the IEEE 802.11 Protocol to Achieve a Theoretical Throughput Limit," *IEEE/ACM Trans. on Networking*, vol. 8, NO. 6, pp.785-799, Dec. 2000.

[6] J. Chen, K. M. Sivalingam, P. Agrawal, and R.Acharya, "Scheduling Multimedia Services in a Low-Power MAC for Wireless and Mobile ATM Networks," *IEEE Trans. on Multimedia*, Vol.1, NO.2, pp.187-201, June 1999.

[7] A. Banchs, X. Perez, M. Radimirsch, and H. J. Stuttgen, "Service differentiation extensions for elastic and real-time traffic in 802.11 wireless LAN," *IEEE Workshop on High Performance Switching and Routing*, pp.245-249, 2001.

[8] A. Chandra, V. Gummalla, and J. O. Limb, "Wireless Medium Access Control Protocols," *IEEE Communications Surveys*, Second Quarter 2000.

[9] B. P. Crow, I. Widjaja, J. G. Kim, and P. T. Sakai, "IEEE 802.11 Wireless Local Area Networks," *IEEE Communications Magazine* Vol.35, pp.116-126, Sep. 1997.

[10] J. Deng and R. S. Chang, "A Priority Scheme for IEEE 802.11 DCF Access Method," *IEICE Trans. Commun.*, Vol.E82-B, NO.1, Jan. 1999.

[11] HIPERLAN Type 2 Standard, *ETSI*, 2000.

[12] C. Fullmer and J. Garcia-Luna-Aceves, "Floor acquisition multiple access (FAMA) for packet-ratio networks," *Proc. SIGCOMM'95*, pp.262-273, Cambridge, MA.

[13] S. J. Golestani, "A Self-Clocked Fair Queueing for Broadband Applications," *IEEE INFOCOM '94*, pp.636-646, April 1994.

[14] D. J. Goodman, R. A. Valenzuela, K. T. Gayliard, and B. Ramamurthi, "Packet Reservation Multiple Access for Local Wireless Communications," *IEEE Transactions on Communications*, vol.37, no.8, pp.885-890, Aug. 1989.

[15] P. Goyal, H. M. Vin, and H. Cheng, "Start-Time Fair Queueing: A Scheduling Algorithm for Integrated Services Packet Switching Networks," *IEEE/ACM Trans. on Networking*, Vol.5, NO.5, pp.690-704, Oct. 1997.

[16] D. J. Goodman and S. X. Wei, "Efficiency of Packet Reservation Multiple Access," *IEEE Trans. on Vehicular Technology*, Vol.40, NO.1, pp.170-176, Feb. 1991.

[17] IEEE 802.11 Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, *IEEE*, 1997.

[18] IEEE 802.11 Status of Project IEEE 802.11e, http://grouper.ieee.org/groups/802/11/Reports/tge_update.htm

[19] R. Jain, A. Durresi, and G. Babic, "Throughput Fairness Index: An Explanation," *ATM Forum/99-0045*, Feb. 1999.

[20] K. Kim, S. Shin, and K. Kim, "A novel MAC scheme for prioritized services in IEEE 802.11a wireless LAN," *ATM (ICATM 2001) and High Speed Intelligent Internet Symposium, Joint 4th IEEE International Conference*, pp.196-199, 2001.

[21] C. E. Koksal, H. Kassab, and H. Balakrishnan, "An Analysis of Short-Term Fairness in Wireless Media Access Protocols," *ACM SIGMETRICS 2000*, Santa Clara, CA, June 2000.

[22] Y. Kwok and V. K. N. Lau, "A Quantitative Comparison of Multiple Access Control Protocols for Wireless ATM," *IEEE Trans. on Vehicular Technology*, Vol.50, NO.3, pp.796-815, May, 2001.

[23] A. Muir and J. J. Garcia-Luna-Aceves, "Group allocation multiple access in single-channel wireless LANs," *Proc. Communication Networks and Distributed Systems Modeling and Simulation Conference*, Phoenix, AZ, 1997.

[24] A. K. Parekh and R. G. Gallager, "A generalized processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case," *IEEE/ACM Trans. on Networking*, Vol. 1. No. 3, pp.344-357, June 1993.

[25] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC Protocol Work Well in Multihop Ad Hoc Networks?" *IEEE Communications Magazine*, June 2001.

[26] J. L. Sobrinho and A. S. Krishnakumar, "Quality-of-Service in Ad Hoc Carrier Sense Multiple Access Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pp.1353-1368, Aug. 1999.

[27] N. H. Vaidya, P. Bahl, and S. Gupta, "Distributed Fair Scheduling in a Wireless LAN," *Proc. Mobicom'2000*, Boston, MA, USA, Aug. 2000.

[28] M. Veeraraghavan, N. Cocker, and T. Moors, "Support of voice services in IEEE 802.11 wireless LANs," *Proc. of IEEE INFOCOM'2001*, pp.488-497, Vol.1, 2001.