Avoiding Collision with Hidden Nodes in IEEE 802.11 Wireless Networks

Jinkyu Lee, *Student Member, IEEE*, and Ikjun Yeom

***Abstract*—In wireless networks, collision is a major factor of performance degradation. In this letter, we propose a scheme for reducing collision in IEEE 802.11 networks. Each node can avoid collision by maintaining a disjoint set of time slots for transmission. Through simulation, we show that the proposed scheme is effective to reduce collision even in the presence of hidden nodes.**

***Index Terms*—IEEE 802.11 DCF, avoiding collision, hidden nodes.**

I. Introduction

The IEEE 802.11 Distributed Coordination Function (DCF) [1] has become a popular technology for various wireless networks due to its effectiveness in reducing collisions with a simple and decentralized fashion. To reduce collisions, it employs a collision avoidance mechanism called Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), in which each node randomizes its transmission time by choosing a random number and waiting for it before transmitting its packet. Even though the IEEE 802.11 DCF reduces collisions significantly compared to CSMA/CD based MAC protocols, it still suffers from a significant number of collisions, especially in the presence of a large number of nodes in a network.

There have been several researches to solve these problems regarding the random number selection in [2], [3], [4], [5], [6]. Among them, Early Backoff Announcement (EBA) in [2] has been paid attention due to its simple and effective way to reduce collision. In EBA, each node piggybacks the random backoff number for the next packet in the current transmission to avoid multiple nodes to select the same number. One basic assumption of EBA is that the piggybacked backoff number of a node is delivered to other nodes safely, and this makes it hard to apply EBA for topologies with hidden nodes.

In this letter, we propose a novel scheme for reducing collision in IEEE 802.11 networks. In this scheme, each node monitors the link, and chooses an idle slot for its transmission. Once it successfully transmits its packet, it keeps the slot for a while to transmit its subsequent packets. To improve link utilization, a node may keep a set of several slots. As long as the set of slots is maintained disjointly, a node can access the link without collision. The main advantage of this scheme over EBA is that it does not need to explicitly inform slot numbers to others since each node individually senses the link to avoid collision. Hence, the proposed scheme performs well in the presence of hidden nodes since a node can usually detect transmissions of hidden nodes even though it is not possible to decode them. In the rest of this letter, we present the scheme in detail, and evaluate performance of it compared with the legacy IEEE 802.11 and EBA.

Manuscript received April 28, 2009. The associate editor coordinating the review of this letter and approving it for publication was S. Pierre.

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2007-313-D00670).

J. Lee is with the Division of Computer Science, Korea Advanced Institute of Science and Technology, Daejeon, South Korea (e-mail: jinkyu@cps.kaist.ac.kr).

I. Yeom (corresponding author) is with the Department of Computer Engineering, Sungkyunkwan University, Suwon, South Korea (e-mail: ikjun@skku.edu).

Digital Object Identifier 10.1109/LCOMM.2009.090975

II. Proposed Scheme

A. Basic operations

The basic idea of the proposed scheme is that a node carefully chooses unique slots for its transmission instead of random selection in the legacy IEEE 802.11 DCF. To apply this basic idea, time is counted in a slotted manner, and this time slot number increases by one for each slot time with modulo m operation[1]. Each node has a disjoint set of slots, $\{R_i\}$, in the range from 0 to $m-1$, and transmits its packet only when the current slot number is corresponding to the one of its slots. Here note that each node may count its own slot number, and it does not need to be synchronized for maintaining the disjoint sets. For fairness and high utilization, we configure the number of slots for each node as $\lfloor \frac{m}{n} \rfloor$ slots, where n is the number of nodes in the network. First, we assume that we know n, and we will present how to estimate n later.

A node classifies time slots into k-state ($0 \leq k \leq MAX$) according to the number of successful transmissions in a slot. Initially, all slots are set to 0-state. Once a node chooses a slot (assumed to be not occupied by other nodes) and successes to transmit its packet in the slot, the slot is set to 1-state. For each subsequent success of transmission in the slot, the node increments the state of the slot by one up to MAX. For each fail of transmission in the slot, it decrements the state of the slot. If k becomes zero, then the node releases the slot and finds another slot. A node is regarded to have a slot when $k \geq 1$ for the slot. With this state based slot management, we can avoid frequent occupation and release of a slot. However, if we set MAX too high, slot management may not be responsive to topology changes. Through a number of simulations, we have observed that $MAX = 2$ or 3 shows best performance. In this letter, we set $MAX = 2$. The basic operations of the proposed scheme is summarized as follows:

• *Initialization:* If a new node joins a network, the node monitors the link and picks $\lfloor \frac{m}{n} \rfloor$ number of (assumed to be

[1]The value, m, is the maximum number of nodes in a network, and it is a pre-defined constant. Practically, we can let m conservatively large. *e.g.*, 256.

empty) candidate slots. Initially, the states of the slots are set to zero. Then, it attempts to transmit its packets in the candidate slots, and increments the state of the slot by one for a successful transmission. If it fails, it chooses another slot until it has $\lfloor \frac{m}{n} \rfloor$ number of k-state slots ($k \geq 1$).

• *Detecting a new node:* If a new node joins a network, and n increases, each node in the network keeps the $\lfloor \frac{m}{n} \rfloor$ largest state of slots (where n is the current number of nodes), and releases the rest of slots for the new node. Here note that the existing nodes do not need to inform which slots they release to the new node. The new node finds its slots via the *Initialization* process in the above.

• *Detecting left of a node:* If n decreases, in order to maintain link utilization, each node in a network finds more slots to have $\lfloor \frac{m}{n} \rfloor$ number of slots via the *Initialization* process.

B. Estimating the number of nodes

To count the number of nodes (n) accurately, each node may explicitly inform its existence to others when there is no hidden node. In the presence of hidden nodes, however, this active approach is not applicable since a node cannot directly communicate with hidden nodes. Here, we devise a passive monitoring based mechanism to estimate n in the presence of hidden nodes. The mechanism is based on overhearing and the observed collision rate. Basically, each node overhears others' transmissions to estimate the number of nodes, initially, and this number is periodically updated. However, the estimated number based on overhearing is often inaccurate due to hidden nodes and incorrect overhearing. Suppose that n is the actual number of nodes, and n_0 is the observed initial number of nodes. Then, consider the followings:

• If n_0 is over-estimated ($n_0 > n$), then $\lfloor \frac{m}{n_0} \rfloor n < m$, which means that there are some idle (unoccupied) slots. Since a small portion of idle slots does not degrade the link utilization seriously, we just keep n_0 for n. In Section III, we confirm that the link utilization is not sensitive to an over-estimated number of nodes.

• If n_0 is under-underestimated ($n_0 < n$), then $\lfloor \frac{m}{n_0} \rfloor n > m$, and some slots are occupied by more than two nodes. In this case, collision is inevitable, and the collision rate r will be

$$r = \frac{n \cdot \lfloor \frac{m}{n_0} \rfloor - m}{m} \approx \frac{n}{n_0} - 1 \quad (1)$$

Based on (1), we adjust n using measured r as follows,

$$n = n_0 \cdot (1 + r) \quad (2)$$

Note that (2) can be applied for over-estimated n_0 since r is almost zero with over-estimated n_0. We will show the accuracy of our estimation mechanism in Section III.

C. Achieving fairness

In the proposed scheme, since a node keeps its time slots for a while, fairness in the link access rate is dependent on the distribution of time slots. In the long-term, the proposed scheme can realize fairness implicitly since (a) each node has multiple randomized time slots when $n < m$, which is the most case since we set m conservatively large; and (b) in and out of nodes perturbs the distribution of time slots. To improve fairness in the short-term, we may consider the following two alternative methods:

• To change and randomize the distribution of time slots, a node may replace its slot intentionally. When a node chooses a slot, it sets a timer for the slot. After the timer is expired, it releases the slot, and acquires a new one. The advantage of this method is that it does not require message exchanges. Since acquiring a new slot causes collision, however, link utilization can be reduced depending on the timeout interval.

• To achieve fairness more rigorously, we may change time slots of each node for each round. In this case, to avoid collision while changing slots, slots are changed along with a pre-defined hopping sequence. Any sequence pattern can be used as long as (a) it is randomized for fairness; and (b) uniqueness of each slot is protected while hopping. There could be a number of such sequences, and we may simply use a pseudo random number generation function to generate a hopping sequence. This method can realize very high degree of fairness since the distribution of slots are randomized for each round. The drawback of this method is its implementation cost. For this method, rounds should be synchronized among nodes in a network in order to change time slots without collision, which requires message exchanges via piggybacking.

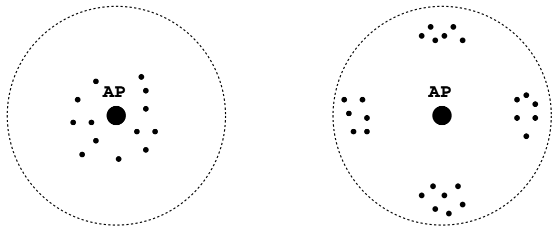
III. Performance Evaluation

In this section, we evaluate performance of the proposed scheme and compare it with the legacy IEEE 802.11 and EBA [2] through *ns-2* [7] simulation. We use the standard values for the Lucent's WaveLAN as the radio model, and a nominal radio transmission (interference) range is 250 (550) meter. Data and ACK transmission rate are 11 Mbps and 1 Mbps, respectively, and one slot time is 0.02 msec. which is corresponding to the backoff time slot in the IEEE 802.11 standard. In each run of simulation, each node sends 1 KB UDP packets with 1 msec. interval.

In simulation with EBA, we assume that EBA knows the exact number of nodes so that throughput of EBA can show an upper bound of achievable throughput when there is no hidden nodes. For our scheme, we let the scheme estimate the number of nodes by itself to observe how the scheme works without the exact number of nodes. For fairness, our scheme employs the timeout-based method instead of hopping since it is more practical without any piggybacking. The timeout interval for each time slot is randomly chosen in the range of 5 to 15 seconds to avoid synchronization.

We consider two topologies as in Fig. 1. In Fig. 1(a), there is an AP in the center, and other nodes are located within 30 - 100 meters randomly apart from the AP so that each node is within other nodes' transmission range, and does not observe any hidden node. In Fig. 1(b), there is an AP, and there are four groups of nodes. To simulate hidden nodes, each group is separated enough not to be within the transmission range of nodes in other groups.

In Fig. 2, we present the aggregated throughput of each scheme. Each result is averaged over ten runs of simulation. We also present Table I to show the numbers of successful



(a) Without hidden nodes (b) With hidden nodes

Fig. 1. Simulation topologies.

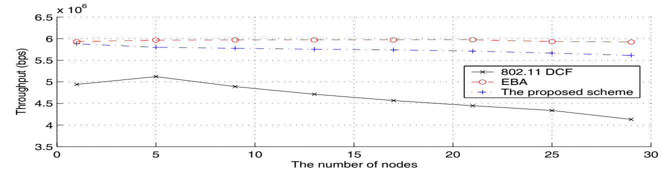
TABLE I
THE NUMBER OF SUCCESSFUL TRANSMISSIONS AND COLLISIONS

	802.11		EBA		Proposed	
	succ.	coll.	succ.	coll.	succ.	coll.
Without hidden	25436	12166	37341	0	35231	1984
With hidden	28991	7831	31290	5387	35480	1084

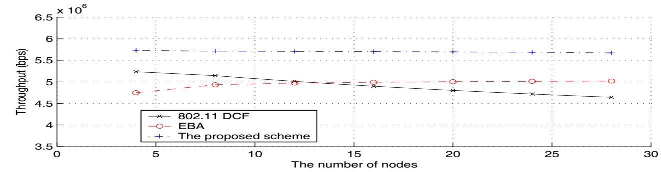
transmissions and collisions for each scheme.[2] The data in the table is collected for $n = 29$ without hidden nodes and $n = 28$ with hidden nodes. It is clear that the legacy IEEE 802.11 does not resolve collision effectively, and the aggregated throughput decreases as more nodes join the network in both topologies. Without hidden nodes, the proposed scheme realizes high throughput close to that of EBA. The gap between them is due to collision for fairness in our scheme. If we increase the timeout interval, throughput can be closer to that of EBA. In the current configuration, we observe more than 95% of Jain's faines index. With hidden nodes, EBA achieves similar throughput with the legacy IEEE 802.11 since it cannot handle hidden nodes whereas the proposed scheme maintains similar throughput with and without hidden nodes.

Now, we observe the accuracy of the node number estimation. From 10 second, a new node joins the network for each five seconds until 20 nodes join the network. In Fig. 3, we compare the actual and the estimated numbers of nodes. It is observed that the estimated numbers are quite close but mostly higher than the actual numbers. This over-estimation is due to the adjustment using Eq. (2). If we disable the adjustment, we may get a closer estimation, but under-estimation is more harmful than over-estimation due to frequent collisions. The impact of the over-estimation on throughput can be observed in Fig. 4. This figure is the result of simulation in Fig. 1(a) with n = 10, 20 and 30. We intentionally set the estimated number of nodes higher than the actual number as shown in the X-axis in the figure, and observe the change of throughput. As shown in the figure, even in an extreme case that the estimated number (30) is three times of the actual number (10), we observe only 3% loss of throughput, and this confirms that performance of the proposed scheme is not sensitive to an over-estimated number of nodes.

[2]In IEEE 802.11 and the proposed scheme, the number of collisions with hidden nodes is smaller than that without hidden nodes since the number of transmission attempts with hidden nodes is reduced due to interference.



(a) Without hidden nodes



(b) With hidden nodes

Fig. 2. Aggregated throughput.

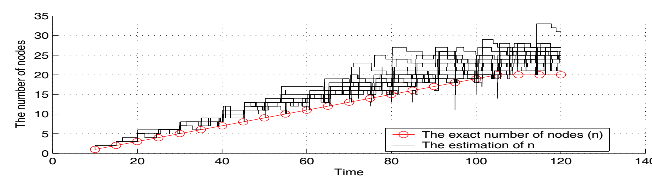


Fig. 3. The estimation of the number of nodes.

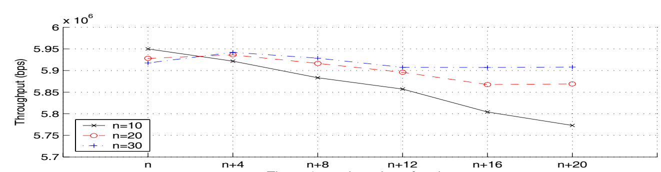


Fig. 4. Throughput change according to the estimated number of nodes.

IV. CONCLUSION

In this letter, we have proposed a scheme for avoiding collision. In the scheme, each node finds and maintains a disjoint set of time slots for transmission. We have shown that the proposed scheme is effective to reduce collision especially in the presence of hidden nodes. We have also discussed how to achieve fairness with the proposed scheme.

REFERENCES

[1] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification, IEEE Std. 802.11, 1997.
[2] J. Choi *et al.*, "EBA: an enhancement of the IEEE 802.11 DCF via distributed reservation," *IEEE Trans. Mobile Computing*, vol. 4, 2005.
[3] H. Wu *et al.*, "Performance of reliable transport protocol over ieee 802.11 wireless lan: analysis and enhancement," in *Proc. Infocom*, June 2002.
[4] Y. Kwon, Y. Fang, and H. Latchman, "A novel MAC protocol with fast collision resolution for wireless LANs," in *Proc. Infocom*, Apr. 2003.
[5] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE J. Select. Areas Commun.*, no. 3, pp. 535–547, 2000.
[6] Q. Zhao *et al.*, "A simple model for nonsaturated IEEE 802.11 DCF networks," *IEEE Commun. Lett.*, vol. 12, no. 8, pp. 563–565, Aug. 2008.
[7] "The Network Simulator - ns-2," http://www.isi.edu/nsnam/ns/.