Achieving Fairness in Distributed Scheduling in Wireless Ad-Hoc Networks

Arun K. Somani and Jianwei Zhou
Dependable Computing & Networking Laboratory
Department of Electrical and Computer Engineering
Iowa State University, Ames, IA 50011
Contact E-Mail: {arun}@iastate.edu

***Abstract*— Fairness is an important design criterion for medium access control protocol in multihop wireless networks. It is a complex problem due to its many dimensions that include consideration of location-dependent contention, spatial reuse of channels, and desire to achieve fully distributed scheduling in the wireless communication systems. This paper presents a localized and fully distributed algorithm with fair scheduling in multihop wireless networks. The proposed algorithm incorporates start time fair queuing (STFQ) into the Distributed Coordination Function (DCF) in IEEE 802.11. Our algorithm accounts for the services that have already been received by the sender to adjust the backoff timer to ensure that every flow gets a fair service. We propose a simple data structure that every node (sender or receiver) needs to maintain and an update mechanism that achieves fairness. We illustrate through simulations that the proposed algorithm achieves the desired fairness.**

I. INTRODUCTION

With the spread of high performance laptops and personal digital assistants (PDA), wireless local area networks have become an emerging technology for today's computer and communication industries. The wireless medium is a broadcast medium, hence can be accessed by multiple devices at the same time. Multiple simultaneous transmissions can result in garbled data making communication impossible. Medium Access Control (MAC) protocols define rules for orderly access to the shared medium and play a crucial role in the efficient and fair sharing of scarce wireless bandwidth. Extensive research has been done in this area [1], [2], [3] and various protocols have been developed. Among them IEEE 802.11 is the most popular wireless LAN protocol that can be used by both, infrastructure-based and infrastructureless, network topologies. In this paper, we consider an infrastructureless network topology, otherwise called an ad-hoc wireless network. An ad-hoc wireless network can be either a single-hop or multi-hop network. In a single-hop ad-hoc network, every node in the network can directly receive transmissions from every other node. The size of such a network is smaller than the transmission range of a node. In a multi-hop network, the network size is typically larger than the transmission range of a node. Hence, all the nodes in the network cannot be reached directly in a single hop.

Fair scheduling in ad-hoc networks has received extensive attention in single-hop ad-hoc wireless [4], [5], [6], [7]. This problem in multihop wireless networks remains largely unaddressed because of the complex and unique issues specific to these networks such as, location-dependent contention, spatial channel reuse, and fully distributed scheduling. In this paper, we propose a new approach to achieve fair scheduling in multihop wireless ad-hoc networks.

The paper is organized as follows: The network model, preliminary concepts, and the design issues are presented in Section II. We present a concise introduction to the IEEE 802.11 standard in Section III. Section IV describes the prior work in the area of fair scheduling in multi-hop wireless networks. In Section V, our fairness definition and a detailed description of our distributed algorithms are presented. We also show through a simple example the difference in performance between our approach and the protocols proposed earlier in the literature. Section VI discusses the issues related to the practical implementation of our protocol. We present the results of our algorithm in Section VII and conclude the paper in Section VIII.

II. NETWORK MODEL AND SOME PRELIMINARIES

A. Network Model

In this paper, we study a packet-switched multihop wireless network in which the wireless bandwidth is shared among multiple contending users. Transmissions are locally broadcasted and only those receivers that lie within the transmission range of the sender can receive the transmission. We assume that the channel is error-free, hence any error in the transmission is due to the collision of packets. We also assume that the packets are of fixed size. We discuss some fault tolerance issues in VI.

B. Preliminaries

Definition 1. A transmission from one node to another is called flow. Flows f_1 and f_2 are said to conflict with each other if packets from these two flows cannot be scheduled for transmission simultaneously. Two flows are said to be conflict-free if they do not conflict with each other.

Definition 2. A flow contention graph is defined as $G = (V, E)$, where V denotes the set of all flows and an edge (f_i, f_j) belongs to E if and only if flows f_i and f_j conflict with each other.

Fig. 1 shows an example of network topology and its corresponding flow contention graph. In this figure, flows f_0 and f_1 are active only in region 1 (left most circle), flow f_2

is active in region 2 (middle circle), and flow f_3 is active in region 3 (right most circle). Region 2 overlaps with regions 1 and 3, and due to location of source and destination of flow f_2, it experiences contention with all the other flows.

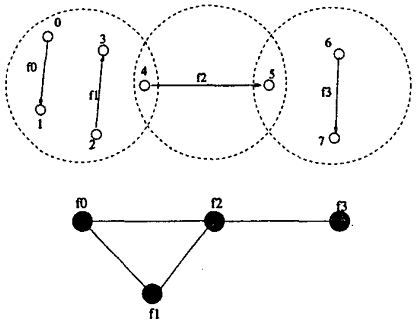

Fig. 1. Node graph vs flow contention graph

C. Design Issues

Fair packet scheduling in ad-hoc wireless networks is an important issue due to several problems that are unique to these networks. Some of the important challenges in the design of fair packet scheduling algorithms in these networks are discussed in this section.

1) Location-dependent contention and spatial reuse: Since wireless transmissions are locally broadcast, collisions and contention for the shared medium are location-dependent. Consider the example in Fig. 1. Flow f_0, f_1 and f_2 contend for the channel with each other. Therefore, when f_0 transmits, f_1 and f_2 should restrain from transmission. Similarly, flows f_2 and f_3 contend for transmission. However, flows f_0 and f_3 do not contend for the channel. Hence, each flow has a different contending flow set depending on its location.

Location-dependent contention also allows for spatial reuse of bandwidth. Specifically, any two flows that are not interfering with each other can potentially transmit data packets over the physical channel simultaneously. Consider the example of Fig. 1 again. Flows f_0 and f_3 are not contending with each other, therefore they can transmit concurrently resulting in a spatial reuse of the channel bandwidth.

2) Conflict of fairness and channel utilization: One unique nature of multihop wireless networks is that multiple flows may transmit simultaneously and the transmission of a flow in a region has an impact on which other flows can transmit in the rest of network. This nature leads to a conflict between achieving fairness and maximizing aggregate channel utilization. For example, consider the network shown in Fig. 1. A maximum aggregate channel utilization of $2C$, where C denotes the capacity of the wireless channel, can be achieved starving flow f_2. However, if flow f_2 receives non-zero channel allocation, then the aggregate channel utilization would be less than $2C$.

3) Fully distributed scheduling: In a multihop wireless network, contending flows may originate from different sending nodes. Therefore, every node needs to implement a local scheduler for its transmitting flows. As ad-hoc networks are infrastructureless, there is no centralized arbitration to access the channel. Hence, co-ordination among the nodes is required to share the bandwidth in a fair manner. Consider Fig. 1 again. The senders $0, 2, 4$, and 6 do not know the packet-level flow information at the other nodes. Hence, solution to fair scheduling in wireless ad-hoc networks by nature needs a fully distributed algorithm.

D. Start-time Fair Queuing

Extensive research has been carried out on fair queuing algorithms to achieve a fair allocation of bandwidth on a shared link. Fair queuing algorithms in literature typically attempt to approximate the Generalized Processor Sharing (GPS) discipline [5]. Start-time Fair Queuing (SFQ) is one among them. In SFQ, two tags, a start tag denoted by $S(i,k,t)$ and a finish tag denoted by $F(i,k,t)$, are associated with each packet, where i denotes the flow number; k denotes the round number; and t denotes the time. Packets are scheduled in the increasing order of the start tags of the packets. Furthermore, at time t, $v(t)$ is defined as the start tag of the packet in service at time t. The working of SFQ is as follows:

1) When a packet arrives at time t,
   if the connection is active, $S(i,k,t) = F(i,k-1,t)$;
   if the connection is inactive, $S(i,k,t) = v(t)$.
2) $v(t)$ is computed as:
   if there is a packet in service, $v(t) = S(i,k,t)$;
   if the system is idle, $v(t) = max_{i,k}F(i,k,t)$
3) The finish time is computed as:
$$F(i,k,t) = S(i,k,t) + \frac{P_i}{w_i} \quad (1)$$
   where P_i is the packet size of flow i and w_i is its weight.
4) Service is in terms of increasing start numbers.

The following example explains the working of SFQ. Suppose there are two flows. Packets from flow 1 arrive at time 0 and 100- with packet sizes as 100 and 99 respectively. Packets from flow 2 arrives at 0+ with packet size as 100. The start and finish times of each packet are:
$S(1,0,0) = 0, \quad F(1,0,0) = 100;$
$S(1,1,0) = 100, \quad F(1,1,0) = 199;$
$S(2,0,0) = 0, \quad F(2,1,0) = 100.$

Hence the transmission order is: packet 1 of flow 1, packet1 of flow 2, and packet 2 of flow 1.

III. Introduction to IEEE 802.11

The IEEE 802.11 specification addresses the Physical (PHY) and Medium Access Control (MAC) layers. At the PHY layer, IEEE 802.11 defines three physical characteristics for wireless local area networks: diffused infrared, direct sequence spread spectrum (DSSS), and frequency hopping spread spectrum (FHSS).

The 802.11 MAC layer, supported by an underlying PHY layer, is concerned primarily with the rules for accessing the wireless medium. Two network architectures are defined: infrastructure-based network and ad-hoc network. An infrastructure-based network is one in which communication between wireless clients are controlled through a centralized controller that is attached to a wired network. The transition of data from the wireless to the wired medium is via an Access Point. The coverage area is defined by an Access Point (AP) and its associated wireless clients, and together with all the devices form a Basic Service Set.

An ad-hoc network is an architecture that is used to support mutual communication between wireless clients. Typically created spontaneously, an ad-hoc network does not support access to wired networks, and does not need an AP to be part of the network.

The MAC architecture in the IEEE 802.11 comprises of two functions: Distributed coordination function (DCF) and Point coordination function (PCF). The PCF resides over DCF in the protocol stack. DCF is the fundamental access method used to support asynchronous data transfer on a best-effort basis and is supported by all stations. While only DCF is used in the ad-hoc networks, the infrastructure-based networks can either employ DCF only or both DCF and PCF. The following section gives an introduction to the DCF in some detail.

A. *DCF in IEEE 802.11*

DCF consists of a four-way handshake: RTS-CTS-DATA-ACK. The exchange is illustrated in Fig. 2.

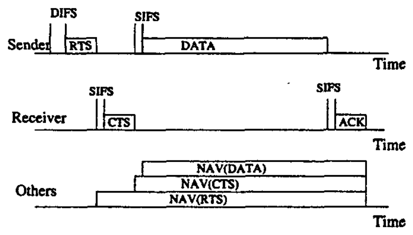

Fig. 2. Four-way handshaking in DCF

When a node (sender) has data to transmit, it waits for a random wait period (backoff timer) watching the channel to be free. The node then tries to acquire the channel by sending an RTS packet. The receiver responds with a CTS packet indicating that it is ready to receive the data. The sender then completes the packet transmission. If this packet is received without errors, the receiver responds with an ACK packet for acknowledgment. If an ACK packet is not received, the packet is assumed to be lost and retransmitted. If the RTS fails, the node attempts to resolve the collision by doubling the wait period. This contention resolution method is called binary exponential backoff (BEB). In order to give preference to a station trying to send an ACK, different waiting intervals are employed. The waiting time for sending an RTS packet is defined by Distributed Inter-Frame Space (DIFS) interval and that for sending an ACK packet is defined by Short Inter-Frame Space (SIFS) interval SIFS is lower in value than DIFS, hence the station sending an ACK attempts transmission before a station attempts to send data, and hence takes the higher priority. In addition to sensing the physical channel, virtual carrier sensing is achieved by using time fields in the packets, which indicates to the other nodes the duration of the current transmission. This time field is called the Network Allocation Vector (NAV) field. All nodes that hear the RTS or CTS message backoff for a duration specified by NAV before sensing the channel again.

IV. Prior Work

In this section, we describe the previous research in the area of fair-scheduling listing the various protocols and highlighting their merits and demerits.

A. *Protocols For LAN*

In [8], Vaidya, Bahl, and Gupta propose a distributed fair scheduling (DFS) for wireless LANs, where every node can listen to every other node. They employ self-clocked fair queuing (SCFQ) to calculate the backoff time and incorporate it with DCF in the IEEE 802.11. This method greatly improves fairness for a wireless LAN. However, it cannot be directly used for multi-hop wireless networks.

B. *Global Information Based Algorithms*

In [10], Lu et al. propose a model to achieve trade-off between efficiency of channel utilization and fairness. This model provides fairness through fair queuing in the basic channel and uses the Maximum Independent Set algorithm to support spatial reuse. If a flow f has a weight of w_f, then their algorithm can guarantee a fraction of $\frac{w_f}{\sum_{j \in B(t)} w_j}$ of the bandwidth, where $B(t)$ is the set of backlogged flows in the entire network at time t.

In [7], Luo and Lu propose another scheme to ensure fairness and maximize channel reuse subject to the fairness constraint. The idealized ad-hoc fair queuing algorithm works as follows. It simulates Weighted Fair Queuing (WFQ) to assign two tags for each arriving packet: a start tag and a finish tag. The scheduler maintains a look-ahead window of ρ packets to make scheduling decisions for each slot, with ρ being set as $\max_{f \in F} L_p/r_f$, where L_p is tehpacket length and r_f is the weight of the flow. For all the packets with a start tag in the range of $[V(t), V(t) + \rho]$, an adaptive coloring algorithm is applied to partition flow packets into $m(t)$ disjoint sets such that within each set concurrent transmissions are possible. Their distributed algorithm employs a weighted round robin approach. The duration of a round is set as $\alpha = (1 + \varepsilon)\Delta$, where Δ is the maximum flow degree. Each flow contends with backoff $\Delta - d$ minislots (d is the flow's degree). This algorithm degrades when flows with different weights or varying packet sizes are considered. Even with equal flow weights and identical packet length applications, the channel utilization can further be improved.

A common drawback of the above two schemes is that they require every node to maintain an up-to-date information of all the flows in the network. Hence, these approaches are not scalable.

In an ongoing research work of Vaidya and Bahl [4], they identify the difficulties in defining fairness in multi-hop networks. They define a Generalized Resource Sharing (GRS) algorithm that requires further investigation as it includes sorting of flows which requires global information.

C. *Local Information Based Algorithms*

In [9], Luo and Lu propose a scheme that only needs local information. The basic idea of this scheme is that each node maintains a table to keep the start time tag of the flows of its neighbors. The backoff of a flow f at a node is set based on the number of neighbor flows whose start tag timer is smaller the start tag timer of f. Three models are proposed by Luo and Lu: Maximize-Local-Min Fair Queuing (MLM-FQ), Enhanced-Maximize-Local-Min Fair Queuing (EMLM-FQ) and Bounded-Fair-Maximize-Local-Min Fair Queuing (BFEMLM-FQ). MLM-FQ transmits flows with local minimum service tags, i.e., a node with the local minimum service tag among all flows in its table transmits immediately. EMLM-FQ uses the backoff mechanism to increase spatial reuse. BFEMLM-FQ used a sliding window as an option to limit the unfairness. Each node maintains an upper bound δ for flow unfairness. When the service tag of a flow reaches the threshold δ, the flow is restrained from transmission temporarily.

The models in [9] are practical and scalable than those in [10] and [7] in that they need only local information. MLM-FQ guarantees fairness however with a reduced throughput. EMLM-FQ improves MLM-FQ's throughput, however might starve some of the flows.

V. Our Approach

The objective of our approach is to develop a distributed fairness algorithm that is simple to implement and uses local information for its operation. It must improve over the shortcomings of previous approaches discussed earlier.

A. *Fairness Technique*

In the literature, the definition of fairness for multihop wireless networks is the same as that of wireline networks: $\frac{P_i(t_0,t_1)}{P_j(t_0,t_1)} = \frac{w_i}{w_j}$, where $P_k(t_0,t_1)$ is the service flow k receives and w_k is the weight of flow k for $k = i$ and j. The weight is defined as the fraction of service that a flow must receive and it is a predefined value.

This definition is not appropriate for multihop wireless networks because of the unique nature of such networks as mentioned in section II and in [4]. We, therefore, use the following guidelines for fairness in such networks, which we believe are more desirable as it maximizes the service to the most starved flow.

G1. Each flow should receive its fair share as per constraints.

G2. Any flow receiving more than its fair share as per *G1*, without affecting other flows is acceptable.

G3. Fairness can mean achieving less than 100% throughput.

This is like Max-Min [12] with additional step, *G3*. Together we refer to it as *SZD-fairness*.

B. *A Distributed Algorithm*

To achieve our fairness objective, each flow must have its fair share in transmitting packets. At the same time spatial reuse must also be allowed. We employ SFQ to acknowledge each flow's service turn and use backoff scheme to allow spatial reuse. To achieve weighted round robin algorithm in a distributed fashion, we introduce the concept of a "round" that keeps track of where a flow is, in comparison to the other flows in terms of its transmission. If a flow is ahead, it must slow down and if it is behind it must pick up in transmission. Specifically, each flow maintains a table (discussed in detail in Section VI) to keep track of its own and its neighboring flows' round numbers. If the round number of a flow is ahead of its neighbors', it adjusts its backoff timer to reduce its chances to transmit and vice-versa. We also give higher priority to those flows that have higher contention because they are more likely to be starved. Such flows are called the *intersection flows*. To achieve this, we notice that the node degrees of the intersection flows are always higher than the other flows in the flow contention graph. Thus in our proposed algorithm, we take into account the number of neighbors a flow has to adjust its backoff time when there is a contention for a channel as described below.

C. *Working of Our Algorithm*

1) Each flow maintains a table which includes *flow ID* and *round number* as table entry for both itself and its neighboring flows. Each flow also maintains a variable called *count*, which is a variable that records the number of packets that have been transmitted. The count is used to implement weights of a flow. Thus, in each round, its initial value is set to the weight of the flow. Once a packet is transmitted, the count is decremented by one. When the *count* reaches 0, the flow is said to have reached the end of a round and its count value is reset to its weight. Thus in each *round*, a flow transmits a number of packets corresponding to its weight.
2) The backoff time for a flow is determined by the difference of the finish tag and the start tag (by SFQ) and the number of neighbors the flow has, expressed as:

$$backoff = (Finish\ tag - Start\ tag) * scalefactor \tag{2}$$

where *scale-factor* is used to keep the initial backoff value less than a chosen constant to ensure some degree of efficiency in terms of channel utilization. It is a constant value and is the same for all nodes.
3) Whenever a flow transmits a packet, it decrements its *count* value by 1. If the value of count is zero, the round number is incremented and the count value is reset to the weight of the flow. Along with transmission of a

data packet, the round number is also transmitted to the neighbor flows.
4) All neighbors of a flow receive this information and update the round number of the transmitting flow.
5) After finishing its transmission, each flow calculates the difference of its round number with the minimum value in its table, that is, $(R - R_{min})$. The following expression is used to obtain the backoff timer for a flow.

```
If $(R - R_{min}) \leq k_1$
        $backoff = backoff * (1+(R-R_{min})/k_1);$
If $(R - R_{min}) > k_1$
        $backoff = backoff * 2;$
```

Here, the factor of 2 acts as a limit to guarantee the throughput, however, it may be changed based on experience. The constant k_1 is chosen to control the performance (including fairness and throughput) and its effect is studied by simulations.
6) For collision resolution, we use the method as proposed in the IEEE 802.11. When a collision occurs, the new backoff time is computed as follows:

```
temp = 2* backoff;
if (temp > collision-threshold)
        temp = collision-threshold;
new backoff= random value between [0, temp];
```

In our simulations, we set the threshold value as 800.

D. Comparison

In this section we give a simple example showing how our algorithm performs better than those in [8] and [9] for the network topology as shown in Fig. 1.

Suppose the initial start time of each flow is: $f_0 = 0$, $f_1 = 1$, $f_2 = 2$ and $f_3 = 3$. All the four flows have the same fixed packet size and same weight. Assume that the transmission time of each packet is 100 (much larger than its backoff timer), and $\frac{P_i}{w_i} = 10$.

The scheme in [8] works as follows:

1) At time 0, backoff of flow f_0 is 0, therefore, it starts transmission. Flows f_1 and f_2 sense that the channel is busy and wait for the channel to become idle. Flow f_3 senses an idle channel and decrements its backoff timer one by one.
2) At time 3, flow f_3 starts transmission. Flow f_0 continues transmission and flows f_1 and f_2 continue their wait.
3) At time 100, flow f_0 finishes its transmission and resets its backoff time as 10. Flow f_3 keeps transmitting. Flows f_0 and f_1 sense an idle channel while flow f_2 senses a busy channel. Thus flow f_1 decrements its backoff to 0, flow f_0 decrements its backoff to 9 and flow f_2 keeps its backoff as 2.
4) At time 101, flow f_1 starts transmission.
5) At time 103, flow f_3 finishes its transmission and resets its backoff as 10. Since flow f_3 senses the channel is idle, it decrements its backoff timer again.
6) At time 113 flow f_3 starts its second transmission.
7) After flow f_1 finishes its first transmission, both flow f_0 and f_1 can sense an idle channel and decrement their backoff timers.
8) At time 210, flow f_0 starts its second transmission.

The outcome of the above procedure is detailed in Fig. 3.

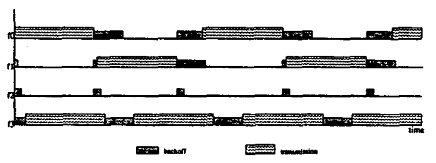

Fig. 3. Channel allocation with time for scheme in [8].

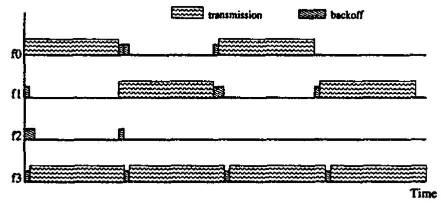

Fig. 4. Channel allocation with time for scheme in [9].

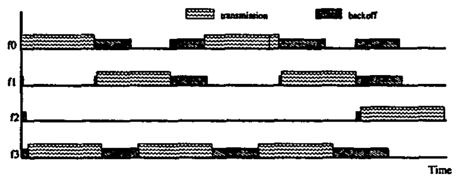

Fig. 5. Channel allocation with time for our scheme

Going through a similar procedure, for the algorithm in [9], the result is illustrated in Fig. 4 and for our algorithm it is depicted in Fig. 5.

As shown in Fig. 3 and Fig. 4, f_2 receives no service. Whenever f_2 is ready to transmit, the channel is occupied by f_0, f_1, or f_3. But in our algorithm, f_2 does get the chance to transmit packets.

VI. Implementation Considerations

To implement our proposed scheme in practice, we need to consider the following issues.

A. Determination of a flow's backoff value

We calculate flow i's backoff timer B_i according to the flow's start tag and finish tag by combining Eqns. (1) and (2) as:

$$B_i = \frac{P_i}{w_i} * scalefactor.$$

To reduce the possibility of collisions, we further randomize the B_i value as follows:

$$B_i = \rho * B_i.$$

where ρ is a random variable with mean 1. In our simulations, ρ is uniformly distributed in [0.9, 1.1].

B. Maintaining table information at the sender and receiver

Each flow maintains a table that contains information about the flow and its neighboring flows. In practice, the table needs to be implemented at each node (both the sender and the receiver) for the flow. The flow ID in the table entry represented by $(senderID, receiverID)$.

C. Round number propagation and update

As the round number increases with the transmission of packets, the number of bytes allocated for transmitting the round number becomes an issue. We transmit the information regarding the round number to all neighboring nodes as part of a data packet in order to limit the overhead of our algorithm. To make this overhead as small as possible, we suggest that the sender transmits whether its round number is updated or not. To accomplish this goal only one bit is needed to communicate the information. If the bit is set as 1, it means its round number is changed and both its receiver and its other neighbor nodes may increment the flow's round number by 1. In the current frame format of the IEEE 802.11 (shown in Fig. 6), the subtype of Frame Control Field in DATA frame has 4 bits (B4 to B7) and the fourth bit (B7) is reserved. We propose to use this bit for the above purpose of informing the receiver and neighbors if the round number is updated or not. Dor faulttolerance purpose, we may have to adopt a more robust mechanism to update round number. Instead of tranmsmitting one bit, we may chhose to transfer more than one bits (that could represent the least l signifcant bits of round number, for example).

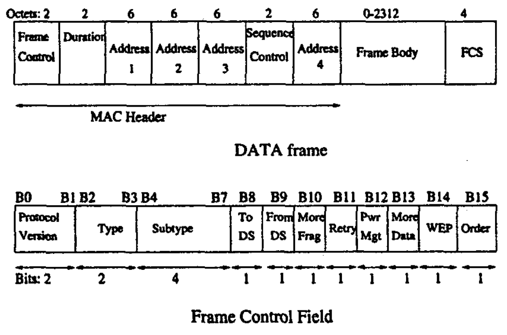

Fig. 6. DATA frame and its frame control field in IEEE 802.11

It is also necessary to allocate one bit in ACK frame because some flows may only be a neighbor of the flow's receiver and therefore ACK can propagate the updated information of the round number. We will discuss the ACK frame in detail in the following section. The proposed ACK frame format is given in Fig. 7. $B0$ in DR of the ACK frame is used to inform its neighbors about the updated information of the flow's round number.

D. Communication between the sender and receiver

In our algorithm the difference between a flow and its neighbors needs to be calculated to decide the backoff time for the flow. The flow's and its neighbors' round numbers are stored at the flow's end nodes. To decide the difference between the flow's round number and its neighbors' round number, we need communication of the round number between the flow's sender and receiver. In a large network, this is too much information to communicate. To avoid this, we find that the difference between the flow's round number and the minimum round number among the other flows in the table can be reached by simply obtaining the largest differences in its sender's or its receiver's table.

To transmit this difference, we propose two changes to the ACK frame in the IEEE 802.11, as shown in Fig. 7. First, the source address (TA) needs to be included in the ACK frame to specify which source is being ACKed. This is mainly used to inform its neighbor nodes which flow it has acknowledged. TA will occupy six bytes. Secondly, one additional byte(DR) is needed to hold the update information of the flow's round ($B0$ as mentioned in above) and the difference of the flow's round number with the minimum one in its table ($B1$ to $B7$). Note that seven bits can only carry a maximum value of 128 which is sufficient for our solution as we only care about the difference, which is about 5(the value of constant k_1 in our implementation).

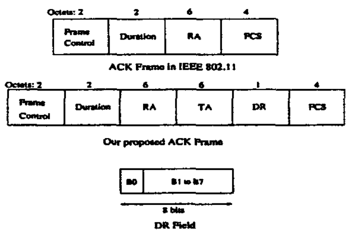

Fig. 7. Original ACK frame in IEEE 802.11 and our ACK Frame

E. Summary

To summarize, we propose modifications to only DATA and ACK packets to realize our algorithm. Since these two packets are reliable, we are able to communicate all the information for the algorithm to function correctly. In addition, each node also maintains a table that contains the following information.

$$(sender,\ receiver,\ round\ number,\ updated)$$

The sender informs the receiver if its round number has changed (1 is changed and 0 is not changed) in the DATA packet. The receiver informs the sender about the difference of its round number with the minimum one of other flows and the flow's round number is changed or not in ACK packet. After receiving ACK, the sender first calculates the difference of its round number with the minimum one in its table, then compare them and uses the maximum of the numbers to calculate the backoff. The neighbor nodes of the receivers update the flow's round number according to the updated information in ACK frame.

VII. Performance of Our Algorithm

In this section, we present the result of the simulations we performed in order to evaluate our proposed algorithm. We

TABLE I
SIMULATION PARAMETERS

Name	Value
channel bandwidth	2Mbps
radio transmission range	250 meters
rate of CBR	1000Kbps
data package size	512 bytes
simulation time	1000s
routing protocol	DSR
max contention window size	800

compare our algorithm with the IEEE 802.11 MAC protocol. Our algorithm is implemented in the ns-2 simulator [2]. Specifically, we modified the original implementation of IEEE 802.11 according to our proposal.

A. Simulation parameters and evaluation criteria

The important parameters used in the simulation are listed in Table I.

In our simulation we choose Dynamic Source Routing (DSR) as the routing protocol. As presented by simulations in [11] DSR outperforms other routing protocols for multihop wireless networks such as, Destination-Sequence Distance Vector (DSDV) and Temporally-Ordered Routing Algorithm (TORA).

In our analysis, we use fairness index and throughput as two criteria to evaluate the protocols. Throughput is defined as the number of packets transmitted during a specific duration. Fairness index (FI) is expressed as follows [12].

$$FI = \frac{(\sum_f T_f/(C \times r_f))^2}{number\ of\ flows * \sum_f (T_f/(C \times r_f))^2}$$

In this expression, T_f denotes throughput of flow f (in Mbps), C denotes the bandwidth of the wireless medium and r_f denotes the proportion of service it can get according to SZD-fairness.

The meaning of the expression is as follows.

Let r'_f denote the ratio of bandwidth used by a flow. $r'_f = \frac{T}{C}$. Let r_f denote the fraction of bandwidth allocation according to the SZD-fairness. The metric $X_f = \frac{r'_f}{r_f}$ denotes the ratio of the achieved bandwidth fraction to that defined by the SZD-fairness. Let M and V denote the mean and the variance of X_f for a set of "F" flows. Then, the fairness ratio is defined as:
$FR = \frac{M^2}{V}$.

For any medium access algorithm that achieves fairness, the mean value is expected to be close to 1 with a small variance. Hence, the larger the value of FR for a MAC protocol, the better the fairness guarantee is.

The factor r_f for each flow is computed based on its contention and weight for contending flows according to SZD-fairness as follow:

```
1) Initially $k = 1$
2) $r_i^0 = \frac{w_i}{\sum w_j}$, where $w_j$ is the weight of flow which is in contention with flow $i$
3) $F_{c_i}^0 = \sum_{c_i} r_i^0$
4) $\tilde{r}^k = \min_{c_i \in C^k} \frac{1}{\sum_{\{f_i | f_i \in c_i \setminus c_s\}} w_i}(1 - F_{c_i}^{k-1})$
5) $r_i^k = \begin{cases} r_i^{k-1} + r^k \cdot w_i & \text{for } f_i \in c_i \setminus c_s \\ r_i^{k-1} & \text{for } f_i \in c_s \end{cases}$
6) $F_{c_i}^k = \sum_{c_i} r_i^k$, that is, compute the allocation of each clique
7) $C^k = C^{k-1} \setminus c_s$
8) $k = k + 1$, go to 2
```

where

$F \rightarrow$	allocated bandwidth
$r_i \rightarrow$	each flow's allocation
$C \rightarrow$	set of cliques c_i
$w_i \rightarrow$	weight of flow i
$c_s \rightarrow$	a saturated clique i

Consider the example of Fig. 1 and assume that the weight of all flows is 1. First, the basic allocation is calculated as following:
$r_0^0 = \frac{1}{3}, r_1^0 = \frac{1}{3}, r_2^0 = \frac{1}{4}, r_3^0 = \frac{1}{2}$.
Then the minimum increment is computed as $\tilde{r}^1 = \frac{1}{24}$.
Thus $r_0^1 = \frac{1}{3} + \frac{1}{24} = \frac{3}{8}, r_1^1 = \frac{1}{3} + \frac{1}{24} = \frac{3}{8}, r_3^1 = \frac{1}{2} + \frac{1}{24} = \frac{13}{24}$.
Next the minimum increment is computed again as $\tilde{r}^2 = \frac{5}{24}$ and $r_3^2 = \frac{13}{24} + \frac{5}{24} = \frac{3}{4}$.
Therefore the service each flow can receive by SZD-fairness is: $r_0 = r_1 = \frac{3}{8}, r_2 = \frac{1}{4}, r_3 = \frac{3}{4}$.

B. Comparison with IEEE 802.11

We studied two different examples, though small but represntative to carry out a comparison study. In the following set of experiments, all weights are set to 1 since IEEE 802.11 cannot be used for variable weight cases. We use $scalefactor = 0.2$, $k1 = 5$ and $collision - threshold = 800$. The effect of these three parameters to the performance of our algorithm is studied in detail in [13].

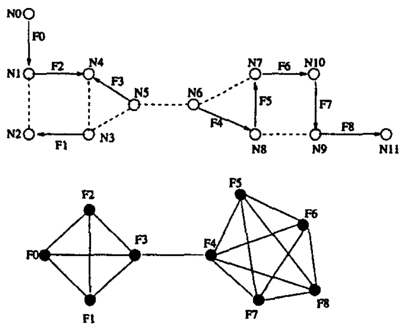

Fig. 8. Example 1's node graph and flow graph

1) Examples: Example 1 shows nine flows in two cliques as shown in Fig. 8 and the simulation results are listed in Table II with weight = 1 for all flows for 802.11 and SZD and different weights and corresponding weighted SZD. Using IEEE 802.11, f_2 and f_8 are the most preferred and f_3 receives much more service than f_4. Our scheme ensures fair allocation among them: f_2 receives almost the same service as f_0 and f_1; f_8 receives similar service as f_6 and f_7; and f_3 receives a little more service than that of f_4.

TABLE II
THROUGHPUT OF EXAMPLE 1

Flow	802.11	SZD	Weight	WSZD
0	65849	51594	2	44545
1	60139	46946	2	56942
2	103145	43642	3	93879
3	57531	27028	2	25629
4	27913	26680	1	15486
5	29780	44892	1	34087
6	40520	50377	2	88657
7	48531	54121	1	46724
8	163644	56599	1	59248
Total	597052	401879	-	465197
Percentage	100%	67%	-	78%
Fairness index	0.71	0.95	-	0.91

Example 2, with 10 flows, is very complex as shown in Fig. 9. By using IEEE 802.11, f_7 receives very little service and f_5 and f_6 receives very high service. Our scheme adjusts them and the result seems fairer than that of IEEE 802.11.

TABLE III
THROUGHPUT OF EXAMPLE 2

Flow	802.11	SZD	weight	WSZD
0	48629	45913	2	48767
1	13458	20909	3	53536
2	32974	11556	2	26153
3	51520	45313	1	22865
4	7226	17198	3	24334
5	74774	27321	2	36588
6	77176	27069	1	22094
7	1493	10432	1	9203
8	4043	8244	1	3650
9	21020	14961	1	18507
Total	332313	228916	-	265697
Percent	100%	69%	-	80%
Fairness index	0.67	0.90	-	0.78

Note that the throughput of Example 2 was computed under scalefactor = 0.3. From the two tables, we see that the throughput of cases with weight > 1 is higher than that of cases with weight = 1. This is because of the backoff timer calculation we adopted in our scheme. In cases with weight = 1, after each transmission, it is possible to increase the backoff timer; while for cases with weight > 1, the backoff timer may increase only when the weight is decremented to 0. Therefore the average backoff of the second case is smaller than that of the first case. We also find that the fairness index is much higher than IEEE 802.11 with all weights equal to 1.0 though it is a little lower than that of cases with weight = 1 using our algorithm.

VIII. CONCLUSION

Ad-hoc wireless networks do not need any infrastructure. Hence, ad-hoc wireless networks are being used in more and more applications, such as in civilian environment (meeting rooms, sports stadiums, taxi cab network etc.), military environments, and emergency operations. Besides throughput, fairness is another critical problem for well-behaved users of

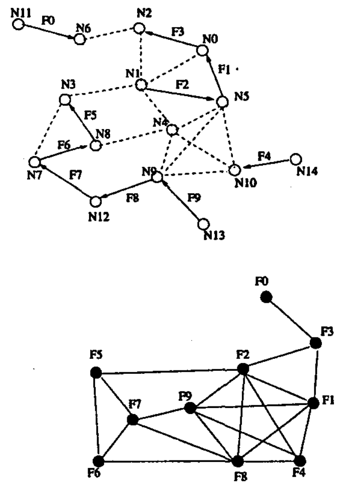

Fig. 9. Example 2's node graph and flow graph

these kind of networks. Because of its own unique characteristic such as location-dependent contention, conflict between fairness and throughput and fully distributed nature, it is a very challenging task to overcome.

In this paper, we have proposed a fairness definition for wireless ad-hoc networks. We have also developed a distributed algorithm to meet the defined fairness by maintaining only the local information. We illustrate the performance of our proposed algorithm with simulation results and comparisons with other approaches proposed in the literature.

REFERENCES

[1] V. Bharghavan, A. Demers, S Shenker and L. Zhang, *MACAW:A Media Access Protocol for Wireless LANs*, Proc. ACM SIGCOMM '94, vol. 24, no. 4, 1994.
[2] IEEE, *IEEE std 802.11 - wireless LAN medium access control (MAC) and physical layer (PHY) specifications*, 1997.
[3] A. Chandra, V. Gummalla and J. Limb, *Wireless Medium Access Control Protocols*, IEEE Communications Surveys, Second Quarter 2000.
[4] N. Vaidya, P. Bahl, *Fair Scheduling in Broadcast Environment*, Technical Report, MSR-TR-99-61, August 1999.
[5] H. Zhang, *Service Disciplines For Guaranteed Performance Service in Packet-Switching Networks* , Proceedings of the IEEE, 83(10), Oct 1995.
[6] S. Lu, V.Bharghavan and R. Srikant, *Fair scheduling in wireless packet networks*, IEEE/ACM Trans. Networking, August 1999.
[7] H. Luo and S. Lu, *A Topology-Independent Fair Queuing Model in Ad Hoc Wireless Networks*, IEEE ICNP 2000.
[8] N. Vaidya, P. Bahl and S. Gupta, *Distributed Fair Scheduling in a Wireless LAN*, ACM Mobicom 2000, Boston, MA, August 2000.
[9] H. Luo, P. Medvedev, J. Cheng and S.Lu, *A Self-Coordinating Approach to Distributed Fair Queuing in Ad Hoc Wireless Networks*, IEEE INFOCOM'01, 2001.
[10] H. Luo, S. Lu and V. Bharghavan, *A New Model for Packet Scheduling in Multihop Wireless Networks*, ACM Mobicom 2000, Boston, MA, August 2000.
[11] J. Broch, D. Maltz, D. Johnson, Y. Hu and J. Jetcheva, *A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols*, Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking, Oct. 1998.
[12] R. Jain, G. Babic, B. Nagendra, and C. Lam *Fairness, call establishment latency and other performance metrics*, Tech. Rep. ATM Forum/96-1173, ATM Forum Document, August 1996.
[13] Jianwei Zhou *A new localized approach to distributed fair scheduling in multihop wireless networks*, Masters Thesis, Iowa State University, Summer, 2001.

The reserach was supported by Nicholas Professorship at Iowa State University.