Rethinking Wireless MAC Architecture for Quality of Service Support – Design and Implementation

Ajit Warrier and Injong Rhee
Department of Computer Science
North Carolina State University

***Abstract*—Existing wireless MAC standards are known to have fairness problems. In this paper, we ask a more fundamental question: should fairness be a property of medium access, or in other words, are we placing the blame on the right place? We argue that fairness is a property of applications and must be implemented by applications. Medium access control should simply provide mechanisms for application-defined fairness policies to be implemented effectively and efficiently. We argue that these mechanisms must provide separate control knobs for two main functions of MAC, namely, fairness control and contention resolution, in order to enable their independent evolution. Furthermore, they must be lightweight and their behaviors must be predictable to enable efficient and consistent implementations of fairness policies on top of these mechanisms. Existing MAC standards do not follow these guidelines. In this paper, we design a new MAC protocol, called *Siren*, adapted from an existing MAC protocol using these guidelines, and implement Siren in a real multi-hop wireless network. The efficacy of our design is demonstrated by implementing on top of Siren various popular fairness policies such as static priority, fair time sharing, proportional rate allocation, earliest deadline first (EDF) and proportional fairness and measuring their performance in our network testbed.**

I. Introduction

One of the frequently cited problems of current wireless networks is related to lack of fairness. The reported fairness problems of MAC include, to name a few, starvation [24], [11], [29], priority inversion [30], inequitable allocation of bandwidth [29], lack of QoS support [18] and multi-rate LAN unfairness [14]. The research community has come up with several solutions [24], [30], [18] to such fairness problems. The common thread among such solutions is that they often require altering the MAC protocol significantly to achieve the authors' notion of "fairness", in effect embedding a fairness policy into the MAC. Clearly, this makes them "point solutions", wherein the proposed changes to the MAC for one solution are incompatible with those for other solutions. The underlying reason for this is the *MAC-centric* notion of fairness prevalent in the research community today.

Evidence for this can also be seen in the design of several existing QoS-enhanced MAC protocols such as IEEE 802.11e and HIPERLAN/1 [5]. These MAC protocols either explicitly or implicitly implement some notions of fairness defined by their designers. For instance, IEEE 802.11e implicitly supports a fairness notion similar to proportional rate allocation among different priority classes where the exact allocation among the classes are hard to tract due to interaction with contention resolution (CR) and window backoffs, and HIPERLAN/1 explicitly supports a policy similar to EDF (earliest deadline first). The solutions to fix the fairness problems of existing MACs (especially IEEE 802.11) have the same problems. For instance, [15], [13], [28], [31] embed their solutions into MAC directly so that MAC is tuned to support only the notion of fairness that the authors define.

Independent of whether such policies are adopted by designers explicitly or purely by chance due to some design and implementation artifacts, a MAC protocol embedded with a predefined notion of fairness makes it very difficult to support diverse fairness policies that application utility functions dictate. Furthermore, an embedded fairness policy which suits some applications may be unfair for different applications. These two observations, namely (a) the ever-growing list of point solutions to enforce different fairness policies requiring MAC protocol changes and (b) the difficulty in enforcing a fairness policy in a MAC which already has an embedded notion of fairness motivate us to take an altogether different approach to the problem of MAC architecture design. In this paper, we outline important design principles for a flexible and efficient MAC architecture designed to support application-level fairness policies; whatever those policies may be. In order to demonstrate that these principles are realizable and effective using off-the-shelf radios, we propose and implement a new MAC protocol called *Siren* on the MicaZ sensor radio chip interface. We then show the versatility of Siren by implementing various fairness policies including static priority, fair time sharing, EDF and proportional fairness on top of Siren in our network testbed. In particular, the proportional fairness in multi-hop networks has never been implemented in a real network. We provide its first practical implementation.

Our main contributions in this paper are:

1) Identifying design principles for a flexible and efficient MAC architecture.
2) Proposing of a MAC architecture, Siren, which follows these principles.
3) Developing algorithms for the implementation of several different fairness policies on top of Siren, demonstrating its flexibility and efficiency.
4) Implementing Siren and these fairness policies on a conventional CSMA-based radio – the CC2420 and experimental results on a 30 node multi-hop sensor testbed.

The rest of the paper is structured as follows: Section II

present the design principles, Section III presents Siren. In Section IV, we propose algorithms for several fairness policies for implementation on top of Siren. In Section V, we describe the implementation of Siren on the CC2420 [2] radio. In Section VI, we study the efficacy of our implementation on a real multi-hop wireless sensor network testbed comprising of 30 Micaz sensor nodes. Section 6 discusses related work and Sections VII and VIII present related work and concludes the paper.

II. Design Principles

1) An Application-centric Fairness Approach: We perceive fairness to be a property of applications leaving the MAC to do what it does best, i.e. medium access. Fairness and efficiency are in the eyes of beholders. They are properties strictly governed by the utility of application. It is the responsibility of the application (and not the MAC) to define the type of fairness that it needs (e.g. temporal fairness, proportional fairness, max-min fairness, etc.) and then enforce this notion of fairness upon the MAC. This *application-centric* notion of fairness follows directly from the well-known system design principle to separate mechanisms from policies. Since fairness policies are application-defined, MAC must simply implement mechanisms in which application policies can be efficiently implemented. We view that this decoupling between fairness policy and fairness mechanism to be the key principle in designing MAC.

2) Decoupling Priority Resolution From Contention Resolution: There could be many different ways to construct mechanisms for implementing various fairness policies. A common technique is to (1) assign priorities to data packets, (2) manipulate these priorities dynamically to implement a specific policy dictated by applications, and (3) ensure channel access to be ordered in terms of the priorities. Several schemes [19], [18], [25], [28] follow this approach where fairness mechanisms are supported by *priority resolution* (PR) – the mechanisms to implement the prioritized access of packets. They differ mainly in the ways that CR is combined with PR. Unfortunately many of these protocols allow their contention resolution mechanisms to depend on fairness control mechanisms (in this case, PR). This coupling of CR and PR mechanisms hinders independent evolution of each mechanism. There exists a large body of work to improve CR alone independently from PR. If PR uses a specific way that CR is implemented, then such improvement work cannot be easily applied. For example, DWOP [19] uses RTS/CTS and acknowledgment for sharing of common knowledge on the priorities of outstanding packets among competing nodes. BTPS [30] also uses RTS/CTS and multiple channels. Thus, if a new and improved medium access scheme does not use RTS/CTS, acknowledgment or CSMA/CA types of CR, then the schemes devised in [19], [25], [28] are difficult to apply. Sometimes the use of a different medium access mechanism than RTS/CTS is dictated by upper layers such as routing. Good examples are ExOR [8] and networking coding [20] that use strictly broadcast not involving RTS/CTS. These protocols cannot be implemented along with some fairness policies of applications if PR requires use of RTS/CTS.

The conventional coupling of PR and CR also degrades the modularity of the system by causing problems in one domain to creep into the other, interfering with their correct operation. For instance, IEEE 802.11e's CR mechanism specifies that the backoff timer of a station needs to be paused when it detects activity on the channel, and then resumed again when the channel is idle. Thus, it is possible that a lower priority station may capture the channel causing priority inversion [30]. Therefore, the decoupling of PR and CR is essential also for the modularity of the system.

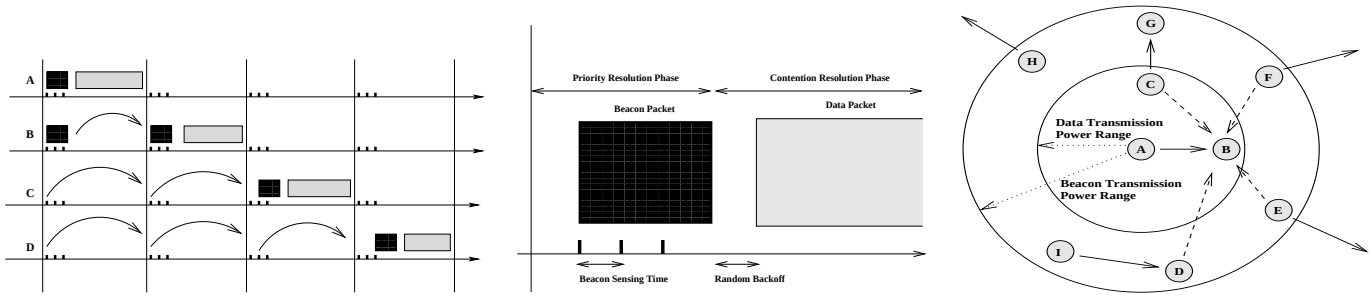
3) Lightweight and robust priority resolution: The implementation of PR in multi-hop wireless networks may require sharing common knowledge about the priority of outstanding packets among nodes within an interference range. In some protocols [19], [18], this is achieved by piggybacking packet priorities in each packet transmission including RTS/CTS and acknowledgment. However priority information may become stale or lost due to several reasons. First, packet priorities may change dynamically even at the time-scale of medium access, requiring nodes to perform an RTS/CTS exchange every time the packet priority changes (the CTS is necessary to inform nodes two hops away). Second, priority information could be lost due to collisions and interference. Stale or lost priority information may lead to deadlock in these protocols. PR must rely on only (if not at all) a minimal amount of information sharing which can transpire in the time scale of medium access and such information sharing must be robust to channel interference and noise.

III. The Siren MAC Architecture

In Siren, time is slotted, and all nodes within a two-hop range are synchronized at the slot boundaries. In each slot packets queued at a node for transmission undergo three phases in sequence: 1) A priority assignment phase in which the head-of-line (HOL) packet is assigned a discrete priority i, $0 \leq i < P$, where P is the number of *priority levels* supported, and 0 is the highest priority. 2) A PR phase where nodes with backlogged packets contend for the medium based on their HOL packet priorities. At the end of the PR phase, only the nodes with the highest priority HOL packets are left for contention in the CR phase 3) A CR phase during which nodes with the same (highest) priority[1] contend for the medium on an equal basis.

By clearly separating the priority assignment phase from the CR and PR phase, and keeping it independent of the MAC, Siren fulfills the first design principle of allowing the application to decide the fairness policy. The specific details of the priority assignment phase depend on the application's fairness requirements, and we give algorithms for priority assignment for different fairness policies in Section IV. Also, by separating the CR and PR phase, Siren fulfills the second

[1]For brevity, we will abuse terminology and refer to "nodes with HOL packet priority i" as "nodes with priority i".



(A) (B) (C)

Fig. 1. (A) shows the working of Siren for nodes A, B, C and D with priorities 1,1,2 and 3 respectively. Both A and B put beacons on the channel after one CCA time. C and D sense the beacon after two CCA times, and hence they postpone their transmissions to the next time slot. Both A and B enter the CR phase and will contend for the channel. A picks a shorter backoff and successfully transmits a packet, while B senses this packet and defers its transmission to the next slot. In the third time slot, C transmits beacon after two CCA times which is sensed by D after three CCA times. Hence D defers its transmission, while C transmits its packet. Finally D transmits its beacon and packet in the fourth time slot. (B) shows the expanded view of one time slot which clearly shows the PR and CR phases. (C) shows how node A uses a higher beacon transmission power to block out potential interfering nodes for receiver B. C is a one-hop node whose transmission to G would interfere with both A's transmission and B's reception. F and E are two-hop nodes whose transmission would affect B's reception. I's transmission to D would not affect A or B, however D's acknowledgment to I would interfere with B's reception. Node H's transmission would not affect A or B, but it will get blocked out. This is an instance of the exposed terminal due to Siren's beacon transmission.

principle of decoupling priority resolution and contention resolution. In addition, the design of Siren also makes it highly efficient compared to other QoS MACs like 802.11e as we shall see later in this section. We now look at the CR and PR phases in more detail.

A. Priority Resolution in Siren

The priority resolution phase of Siren is inspired from that in EY-NPMA and its variants [5], [27], in which nodes rely on short pulses of energy or *beacons* to inform their neighbors about the priority of their outstanding packet. Hence, the overhead incurred in the PR phase of EY-NPMA is the time for transmission of the beacon times the number of priority levels supported. Clearly, smaller beacons incur less overhead. Early radios (e.g. CC1000 [1]) support bit streaming which allows transmission of extremely small beacons. However the new generation of packetizing radios (e.g. CC2420 [2], most commercial 802.11 radios) does not support bit streaming and allows only valid packets with standard compatible headers to be transmitted. This limitation can significantly increase the overhead of the priority resolution phase if the minimum valid packet size supported by a standard is large. Siren overcomes this limitation by the use of *beacon sensing*. The basic idea is that since a node does not need to completely receive a beacon to be aware of a higher priority node in its vicinity, it only needs to detect the presence of the beacon on the channel through CSMA. We now discuss this concept in more detail.

1) Beacon Sensing: A node with a priority i waits for i *CCA (clear channel assessment) times* which is the minimum time to sense a (beacon) transmission from a neighbor. At the end of the i-th CCA time, if the channel is idle, the node transmits the beacon and enters the CR phase. However, if it senses any beacon before i CCA times, then it defers its own transmission to the next time slot. If there exist many nodes with the same priority, they all transmit beacons simultaneously and will enter the CR phase together. As illustrated in Figure 1 (A), the overhead for the PR phase in Siren is the CCA time multiplied by the number of priority levels plus one beacon transmission time. Since the CCA time for most radios is much smaller than the minimum valid packet transmission time, beacon sensing results in significant overhead savings.

To quantify the overhead saving achieved by beacon sensing we compare the overhead of the PR phase in EY-NPMA and Siren normalized by the total time for transmission of one data packet (with different data packet sizes and 10 priority levels) in Figure 2. We look at two radio environments: Wireless-LAN (WLAN) and ZigBee [4]. ZigBee radios have a link speed of 250Kbps (resulting in a per-byte transmission time or "byte-time" of 32μs, CCA time of 320μs and minimum packet transmission time of 512μs (there is a detailed discussion of these numbers in Section V-A), while WLAN radios have a link speed of 11Mbps, CCA time of 15μs, and minimum packet transmission time of 184μs. As a comparison we also plot the overhead for an 802.11e-based PR scheme where priority resolution is achieved by performing backoffs over non-overlapping backoff windows [7] i.e. $DIFS_i = DIFS_{i-1} + CW_{i-1}$, where CW_i is the backoff window for priority level i. Here the overhead for priority resolution is the backoff period times the number of priority levels. We assume the conventional backoff period of 32 byte-times; 1.024ms for ZigBee and 480μs for WLAN. Siren's beacon sensing saves about 20% overhead compared to EY-NPMA in ZigBee radios, while the saving goes up to 100% for WLAN radios. This is because the CCA time is relatively closer to the minimum packet transmission time for ZigBee radios compared to WLAN radios. Interestingly, 802.11e with non-overlapped backoff periods incurs almost 100-150% more overhead compared to Siren. This is because in 802.11e the PR and CR schemes are interlocked with each other, and hence each additional priority level incurs the overhead of one additional backoff period while in Siren, adding an additional priority level requires only an additional CCA time.

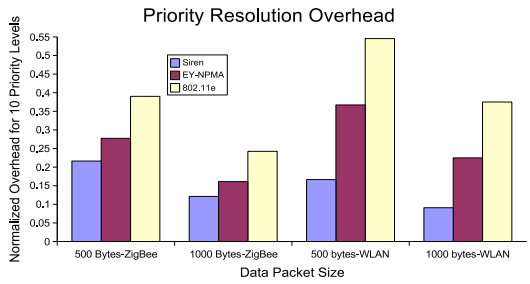


Fig. 2. Normalized priority resolution overhead for Siren, EY-NPMA and 802.11e with non-overlapped backoff periods with different data packet sizes.

2) Beacon Transmission Power: In Siren, a node uses beaconing to inform its neighbors about its priority. However, hidden terminals located at two hops away from the node, and at one hop away from its intended receiver may not be able to sense the beacon and may interfere with packet reception at the receiver [24], [11], as illustrated in Figure 1 (C). This problem is not handled in EY-NPMA and IEEE 802.11e (note that RTS/CTS does not enforce PR across two-hop ranges).

Siren's approach for this problem is highly efficient; it uses a higher transmission power for beacon transmissions compared to data transmissions to preserve the correctness of PR across two hops. The transmission power for beacons should be enough to reach all potential interfering nodes for A's intended receiver. In this paper we set the beacon transmission power of all nodes to be roughly three times that of their data transmission power (Section V-C) for a simplified implementation. This is decided based on the following reasoning. Assuming free-space propagation, where the transmitted signal drops by square root of distance, quadrupling transmission power for beacons corresponds to doubling distance, hence ensuring that the beacon is received by all two-hop interferers. However, since beacon packets need to be only sensed, not received, we find through experiment that simply using three times the data transmission power works for most cases.

Using the same beacon transmission power for all nodes in the network is a conservative approach which may cause the beacon transmissions of some nodes to reach farther away from interfering nodes and shut down non-interfering nodes from transmission leading to loss of network capacity. This problem is called the *exposed terminal* problem. An example is shown in Figure 1 (C). Node H will be shut down by A's beacon, even though its transmission will not interfere with A's transmission to B. Recent work has established that it is possible to detect the interference relations between nodes on a run-time basis [32]. In addition, transmission power can also be tuned so that transmissions reach only a set of interfering nodes [21]. Hence it is possible to tune the beacon transmission power on a per-node basis, so that it reaches the set of interfering nodes and no farther. This will not eliminate all cases of exposed terminals, but will definitely result in improvement in network capacity. However, we do not explore this option in this paper, and leave it for future work. Our experimental results presented in Section VI are obtained without this tuning.

3) Time Synchronization: Siren does not require global clock synchronization. The protocol works as long as the sender is synchronized with all potential interferers (both one and two-hop nodes). Hence conventional distributed clock synchronization algorithms which depend on hop-by-hop clock synchronization [10], [22] are appropriate for Siren. We use FTSP [22] for our testbed.

B. Contention Resolution in Siren

The nodes that survive the PR phase compete on an equal basis for channel accesses in the CR phase. Siren uses CSMA/CA with fixed window backoffs for CR. Nodes take a random backoff and sense the channel at the end of the backoff period. If a node senses a carrier, then it simply goes to the next slot where it runs the PR phase again. This scheme provides equal chances to all competing nodes in the CR phase. We choose this CR mechanism because it is very simple to implement; other more efficient CSMA schemes can also be incorporated [15], [17]. This is an advantage of Siren as it allows any CR mechanism to be incorporated without modifying the PR mechanism.

IV. Implementing Fairness Policies

In this section, we demonstrate how Siren allows effective and efficient implementations of various fairness and QoS policies. We consider the following single-hop and multi-hop fairness and QoS policies: static priority, EDF, proportional rate allocation, and proportional fairness. The core advantage of Siren is that these policies can be implemented at the application without any modification in MAC. Below, we present our implementation; since the implementation of static priority is straightforward, we omit its description.

A. Proportional Rate Allocation and Time Fairness

Consider the problem of proportional throughput allocation in a WLAN [12]. Let there be N nodes in a WLAN which require service differentiation to the effect that their obtained throughputs are in the ratio $r_1 : r_2 : \ldots r_N$. A related problem is maintaining *temporal fairness* (where all nodes receive equal channel time) in multi-rate WLANs. Suppose N stations access the channel with different transmission rates $R_1, R_2, \ldots R_N$ (due to varying channel quality) in a WLAN where all nodes can hear each other. IEEE 802.11b has been shown to be highly "unfair" in such an environment [14]. A solution is to to assign equal time-shares among competing nodes on the channel. Existing work [15], [13], [28], [31] implements this solution by directly modifying MAC. Our solution does not require any modification of Siren. The temporal fairness problem can also be framed as a proportional rate allocation problem in the following manner: if the N nodes access the channel so that their obtained throughputs are proportional to their transmission rates, i.e. $R_1 : R_2 : \ldots R_N$, then the time spent by each node on the channel is equalized.

We propose the following algorithm for achieving *approximate proportional rate allocation*. Let there be P priority levels available. A station i, $1 \leq i \leq N$, sets the priority p_i of its outgoing packets by executing a maximum of $P-1$ Bernoulli trials with probability $\phi(i) = \frac{r_i}{\sum_{i=1}^{N} r_i}$. If it wins in trial m, $1 \leq m \leq (P-1)$, it sets $p_i = m$. If it runs $P-1$ trials without success, then it sets $p_i = P$.

We omit formal proof to save space. Instead, we provide an informal discussion. To see why this algorithm gives the required rate allocation, consider a single time slot. If only a single node wins in the first Bernoulli trial, then it sends a beacon with priority 1, and grabs the channel to transmit its data packet. This occurs with probability $\sum_{i=1}^{N} \phi(i)(1-\phi(i))^{N-1}$. Now, if more than one node wins the first Bernoulli trial, then all such nodes simultaneously transmit beacons and enter the CR phase. As the CR phase in Siren gives equal opportunity to such nodes (as all are in one hop), then the medium access probabilities among such nodes do not follow the required ratios. However, note that the probabilities $\phi(i)$ are normalized to 1. Hence, on average, the number of nodes winning the first (or any) Bernoulli trial is close to 1. Hence the probability of medium access for a node i approximates the Bernoulli probability $\phi(i)$. Since the Bernoulli probabilities are normalized based on the required proportional rate allocation, the medium access probability of nodes winning the first Bernoulli trial approximates the required ratios.

Now consider the case that no node wins the first Bernoulli trial – this happens with probability $\prod_{i=1}^{N}(1-\phi(i))$. Such nodes try the Bernoulli trial again and by the same logic as above their access pattern also approximates the required ratios. This process is repeated for $P-1$ trials. All nodes which do not win in any of the $P-1$ Bernoulli trials transmit beacons with priority P. Clearly the medium access pattern between this set of nodes will not follow the required ratio, but will be equally distributed. However, the probability of all nodes not winning their $P-1$ Bernoulli trials is given by $(\prod_{i=1}^{N}(1-\phi(i)))^{P-1}$. This probability quickly falls down to 0 with increase in P. Hence the more number of priority levels available to the system, the closer is the approximation to the desired ratios.

B. EDF Scheduling in Multi-hop Networks

EDF scheduling can be implemented as described by Kanodia et al. [18]. Every packet at its source is stamped with a TTL (time-to-live) which is its desired end-to-end delay bound. At every hop, just before the transmission of this packet, its TTL is decremented by the amount of time spent in the queue. The TTL is directly mapped to a priority level at each hop. This allows nodes with backlogged packets which have a very low TTL to access the channel quickly and flush such packets. We assign the priority p_i of an outgoing packet i as follows. Let D_{max} be the maximum deadline value that a real time packet can have and P be the number of priority levels. Suppose that a packet i has a deadline d_i. Then $p-i$ is set to d_i/Δ where $\Delta = D_{max}/P$.

C. Proportional Fairness in Multi-hop Networks

Chen et al. [9] propose a joint design of congestion control, routing, and scheduling to maximize aggregate utility in a wireless network. The algorithm is implemented as follows.

Each node maintains a per-flow queue. A node decides which per-flow queue it services next and where to route a packet from this queue based on a *price* which is computed by taking a queue size difference between that node and its neighbors. More precisely, let l_i^k and l_j^k be the per-flow queue length for flow k at nodes i and j. For each flow k, node i computes price C_{ij}^k between itself and each of its neighbors j as follows: $C_{ij}^k \leftarrow l_i^k - l_j^k$. C_{ij}^k indicates whether node i should route packets belonging to flow k over node j. Intuitively, the largest value of C_{ij}^k hints that node i should route packets for flow k over the neighbor j with the smallest queue length l_j^k. If multiple per-destination queues at node i are backlogged, node i first schedules a packet of flow k and forwards this packet to its neighbor j so that the differential price C_{ij}^k is maximized for all flows k and all neighbors j. The rationale behind this is to give to congested nodes a prioritized access to the channel so that they can drain their queues faster (a large queue may cause a large price). The original algorithm in [9] require solving distributed maximal matching over two hops for every packet transmission. This is very time consuming. Instead, we approximate maximal matching by mapping C_{ij}^k of the packet to a priority level and using Siren to resolve priorities among competing neighbors. Note that with a queue size of Δ packets per flow, the value of C_{ij}^k may vary between $-(\Delta-1)$ and Δ, hence requiring 2Δ priority levels for a one-to-one mapping. The packet is then transmitted by Siren using this priority level. This approximation does not guarantee proportional fairness, but it provides a practical implementation that still achieves a high aggregate utility of the system, which is defined to be $\sum log(x)$ where x is a per-flow throughput. This is shown in Section VI.

V. MICAZ IMPLEMENTATION

A. CC2420 Transceiver – Overview

The MicaZ [3] sensor node contains the CC2420 [2] transceiver which is a ZigBee [4] compatible radio. It has a link speed of 250Kbps, resulting in a per-byte transmission time of 32us. It supports 8 different transmission power levels ranging from a minimum of -25 dBm ($< 10\mu$W) to a maximum of 0 dBm (1 mW). It contains two buffers of size 128 bytes each for outgoing and incoming packets respectively. Packets placed in the outgoing buffer need to be *valid* – they must strictly adhere to the ZigBee packet format, else they are dropped.

The ZigBee [4] standard specifies the minimum PHY layer header of 5 bytes (4 bytes preamble and 1 byte synchronization), a MAC layer header of 9 bytes and 2 bytes of CRC. This implies that the minimum packet size with no payload in CC2420 is 16 bytes. Also, the standard specifies that the maximum packet size (including PHY, MAC headers, CRC and payload) is restricted to 128 bytes.

B. Beacon Transmission

The minimum packet size limitation of 16 bytes corresponds to a beacon transmission time of $16 \times 32 = 512\mu$s. By means of experiments, we found out that CC2420 requires sensing the medium for 10 byte-times $= 320\mu$s to sense activity on the channel. Hence we use a beacon sensing time of 320μs for our implementation. Unless otherwise specified, we use 6 priority levels for Siren incurring an overhead of $320 \times 6 + 512$ μs = 2.4ms per packet transmission for priority resolution.

C. Beacon/Data Transmission Power

We use -5 dBm (316 μW) transmit power for data packets, and 0 dBm (1 mW) for beacons.

D. Time Synchronization

We use the FTSP [22] algorithm for time synchronization. In our multi-hop testbed comprising of 30 MicaZ sensor nodes, the FTSP algorithm takes 4 minutes (with time synch messages broadcast every 10 seconds) for all nodes to synchronize. All data transmissions begin 4 minutes after the start of the experiment. Since time synchronization messages are sent as broadcast they are prone to loss due to collisions with data packets. This makes it difficult to maintain time synchronization under high network load. To prevent this problem, synchronization messages are sent with a priority level 0.

E. Large Packet Size Emulation

The CC2420's maximum packet size limitation of 128 bytes is too small relative to the overhead caused by beacon transmissions. To overcome this limitation, we maintain a *virtual packet* at the MAC layer in which the transmission of larger packets is emulated. For every data packet of size 128 bytes received from the upper layers, the MAC driver retransmits the same packet X number of times. The receiver counts the number of CRC-valid copies of a packet it receives. If it receives X copies of the same packet, it will send an ACK, else it will silently discard all copies of the packet. The sender, after sending X copies, waits for an ACK. If it does not receive an ACK, it will signal a failure to the upper layers, and try again. This emulates closely the behavior of a packet of size $128 \times X$ bytes. In our experiment, we set X to 3 for an emulated packet size of 384 bytes.

VI. Testbed Results

A. Experimental Methodology

Our sensor network testbed consists of 30 MicaZ sensor nodes distributed in labs and student offices in our computer science building, as shown in Figure 3. We compare the performance of Siren (implemented as described in Section V) with the following MAC protocols:

(a) Randomized Medium Access (CSMA): As the base case of a MAC protocol which does not support priority resolution, we consider B-MAC [23], which is the default MAC protocol for MicaZ sensor nodes under the TinyOS environment. Medium access in B-MAC is similar to that in 802.11, with the exception that it does not perform exponential backoffs. Just before packet transmission, a node performs a backoff over a fixed window. At the end of the backoff, the node transmits the packet if the channel is idle, else it takes a backoff again. We will refer to this MAC as *CSMA-DATA/ACK* for the rest of this section.

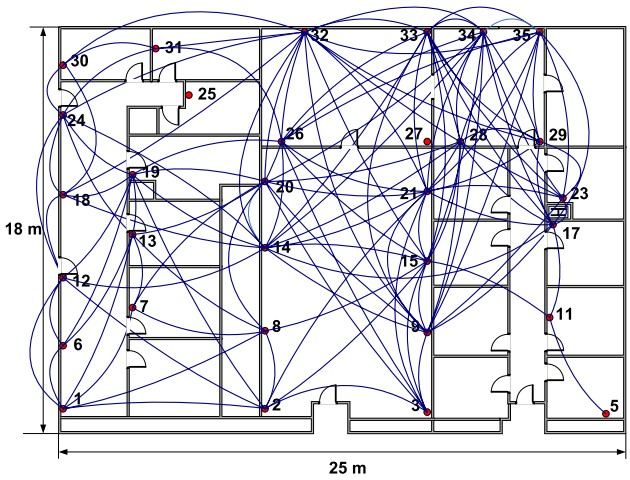


Fig. 3. Above figure shows the 35 node mote testbed distributed across one floor of the MRC building. Lines between nodes indicate connectivity achieved with data transmission power of -5 dBm.

B-MAC does not support the use of RTS/CTS. For a fair comparison, and as an additional performance baseline, we implement RTS/CTS for B-MAC and use it in our experiments, in addition to the regular DATA/ACK mode of access. We denote it as *CSMA-RTS/CTS*.

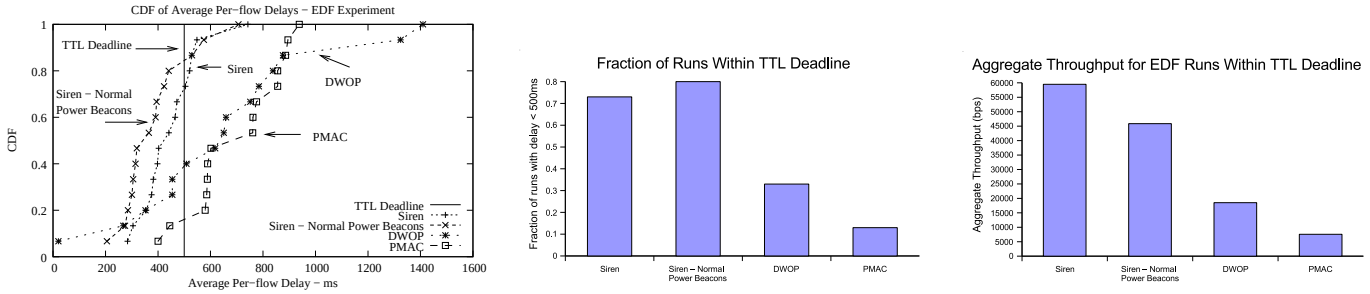
(b) CSMA with Differentiated Backoffs: We emulate the priority resolution of 802.11e and its variants using differentiated backoffs in B-MAC. Our implementation assigns non-overlapped backoff periods to 6 priority levels, allowing absolute service differentiation and denote it as *PMAC*.

(c) DWOP: We implement DWOP on top of B-MAC with RTS/CTS as described in the original paper [19], referring to it as *DWOP* in this section.

B. EDF Scheduling

We implement EDF scheduling as described in Section IV-B. Each source node stamps a TTL of $500ms$ on its outgoing packets. At each hop this TTL is decremented by the amount of time spent by the packet in the queue before transmission. This TTL is used directly as the packet priority for DWOP, while it is mapped to 6 discrete priority levels for Siren and PMAC.

We report the distribution of the average per-flow delay for 15 runs with 10 flows each in Figure 4 (A). Figure 4 (B) shows the fraction of the runs for each MAC in which the average per-flow delay is less than the EDF deadline of 500ms. Siren is able to maintain a delay less than the desired TTL for 80% of the runs, while in the case of DWOP and PMAC, only 30% and 15% of the runs fulfil the EDF deadline. The poor performance of DWOP is due to inconsistent cache information causing



(A) (B) (C)

Fig. 4. (A) shows the distribution of delays in the EDF scheduling experiement. (B) shows the fraction of runs which have average per-flow delay to be less than the EDF deadline of 500ms. (C) shows the aggregate throughput of all such runs for each MAC.

frequent deadlocks. This problem comes up quite frequently on our testbed due to long, lossy links common in wireless networks. One such problem scenario is illustrated in Figure 5. Such a scenario causes large delays for the affected flows.

Siren does not suffer from the same problem in the presence of lossy channel. This is because HOL priorities are advertised through beacons which are short packets, and hence do not incur high loss rate. In addition, it is not necessary for a node to receive the beacon fully and correctly to determine that a neighbor has a high priority packet pending. As long as it detects a busy channel, it will defer access.

We also run the same experiment with Siren with beacon transmission power set the same as data transmission power, denoted by *Siren - Normal Power Beacons*. Although this seems to slightly reduce the average per-flow delay compared to Siren with high power beacons, the aggregate throughput per run is also reduced. This is because the normal power beacons cannot preserve priority across two hops, causing hidden terminal collisions, reducing the throughput at the destination.

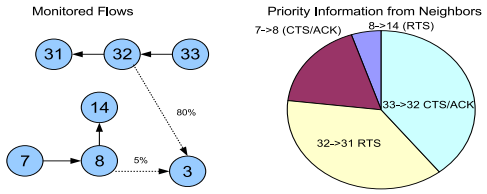


Fig. 5. Above figure illustrates the DWOP problem in lossy networks. Flows $33 \rightarrow 32 \rightarrow 31$ and $7 \rightarrow 8 \rightarrow 14$ are real-time flows from one of the runs in the EDF scheduling experiment. Node 3 is the source of a flow which was observed to be starved. We monitored node 3 and found that it was continuously deferring its own transmission due to perceived high priority packets in its neighborhood. The figure on the right represents the normalized fraction of time spent in deferring attributed to the constituent flow segments. About 75% of the time 3 defers due to the RTS heard from node 32 ($32 \rightarrow 31$) or the CTS heard from 32 ($33 \rightarrow 32$). This is not because the flow $33 \rightarrow 31$ is a high rate flow (the flow rate is set at 3Kbps), but because the RTS/CTS packets from 32 experience an 80% loss at node 3. The consequent data packet signaling the end of the transmission is always lost due to data packet size being considerably larger than the RTS/CTS size, leaving 3 deadlocked. The priority information from neighbor node 8 reaches 3 without much problem since the loss rate from 8 to 3 is 5%.

C. Static Priority Scheduling

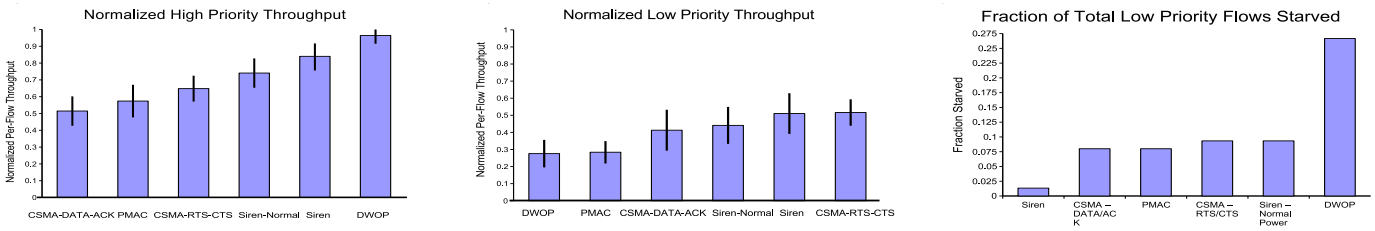
We consider an environment where two classes of applications coexist – high priority (HP) and low priority (LP). Our fairness policy is to allow the HP flows access to the channel as much as possible, deferring access to LP flows. This is achieved by setting the priority of all HP flows to be higher (set to 2) than that of all the LP flows (set to 3). Our experiment consists of 5 HP flows which transmit at a constant bit rate of 3 Kbps, while 5 LP flows transmit at a constant bit rate of 5 Kbps. The sources and destinations of these 10 flows are selected randomly. This experiment is repeated 15 times with different (source, destination) pairs. We report the throughput of each class normalized with the source sending rate in Figures 6 (A) and 6 (B).

Figure 6 (A) shows that both DWOP and Siren are able to provide full access to HP flows with DWOP performing somewhat better. However, DWOP does this at the expense of throughput for LP flows. From Figure 6 (B) we can see that LP flows in DWOP get about 50% of the throughput achieved in Siren. This is again due to the inconsistent cache information caused by lossy links in our testbed. Note that for DWOP, the HP flows do not affect each other, since they are all set to the same priority level (2), hence HP flows do not get into deadlock due to each other. However, LP get deadlocked if there is a high priority flow within two-hops. Figure 6 (C) shows the fraction of LP flows in all the runs that get starved (0 throughput).

D. Proportional Fairness in Multi-hop Networks

We now look at proportional fairness in multi-hop networks using the cross-layer optimization framework described in Section IV-C. The optimization framework consisted of three components, source rate control, per-flow queuing and MAC scheduling. Instead of directly comparing the performance of the framework under each of DWOP, Siren and PMAC, we take the approach of incremental performance evaluation of each component.

We start with the baseline, which is a framework without any congestion control or scheduling. We select 8 sources randomly which transmit packets as fast as possible toward 8 randomly selected destinations. We refer to this as *CSMA-DATA/ACK-Full-Rate*. We then add the source rate control



(A) (B) (C)

Fig. 6. (A) and (B) show the normalized per-flow throughput of high and low priority flows respectively, in increasing order. Throughput is normalized w.r.t source sending rate (3Kbps for high priority and 5Kbps for low priority). (C) shows the fraction of low priority flows getting starved (0 throughput).

and per-flow queuing components to the framework, but leave out the MAC scheduling component. We refer to this result as *CLO-CSMA-XX* – the suffix *XX* denoting whether the underlying MAC uses RTS/CTS or just DATA/ACK. Finally, we test the full framework under Siren, DWOP and PMAC, denoted respectively as *CLO-Siren*, *CLO-DWOP*, and *CLO-PMAC*. The performance metric we report is the aggregate utility ($\sum log(x)$), where x is the throughput obtained by a flow. We report the distribution of the aggregate utility in Figure 7.

Note that a low value of the aggregate utility indicates severe starvation for some flows, although the aggregate throughput may be high as seen in Figure 8. The main causes of flow starvation in conventional wireless networks are the inherent MAC-layer unfairness observed in CSMA-based MAC protocols [24] and the lack of effective congestion control which causes queue overflows at intermediate nodes [16], [29]. Both these factors cause the low aggregate utility in the case of (*CSMA-DATA-ACK-Full Rate*). By adding source rate control and per-flow queuing the starvation due to queue overflows is mitigated to some extent, as can be seen in the case of *CLO-CSMA-DATA/ACK* and *CLO-CSMA-RTS/CTS*. The large improvement seen in *CLO-CSMA-RTS/CTS* indicates that a significant factor of the flow starvation is due to hidden terminal scenarios on our testbed. The disappointingly low utility in *CLO-PMAC* compared to *CLO-CSMA-RTS/CTS* is due to the vulnerability of PMAC to hidden terminals.

Finally *CLO-Siren* and *CLO-DWOP* both show the maximum value of the aggregate utility among all the tested cases. This shows that even after congestion control and per-flow queuing, there is still exists some scope of improvement by tuning the MAC scheduling component. It is interesting that both Siren and DWOP provide more or less equivalent performance for the proportional fairness framework.

E. Proportional Rate Allocation

We test the efficiency of Siren in approximating the rate allocation by running the following test. We pick 4 and 6 nodes of our testbed in sequence in such a way that they are all within radio range of each other. These nodes then transmit packets to a common sink node by the proportional rate allocation algorithm described in IV-A. The sink node records the ratio of the throughputs obtained from each node

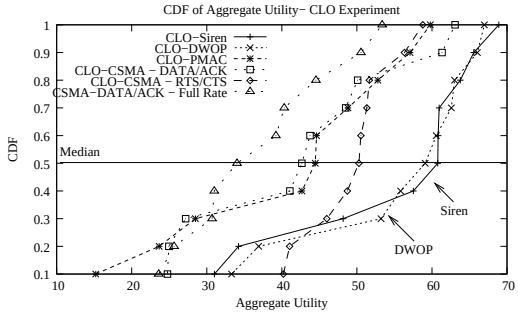


Fig. 7. Distribution of aggregate utility $\sum log(x)$ for CLO experiment.

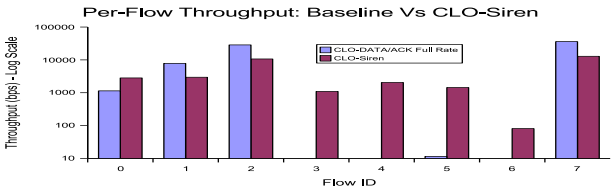


Fig. 8. Comparison of per-flow throughput of baseline CSMA with CLO-Siren for one specific run. Note that CSMA's aggregate throughput is higher, but CLO-Siren gets better utility since no flow is starved.

which we report in Figure 9. We use Siren with 6 priority levels for this experiment.

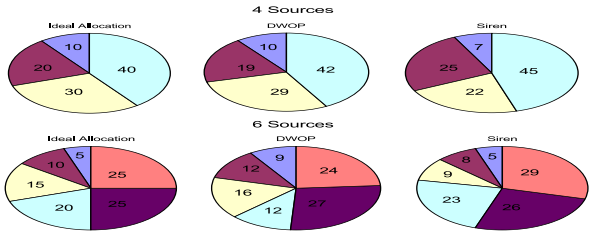


Fig. 9. Proportional rate allocations for DWOP and Siren compared with the ideal rate allocations for 4 and 6 sources.

DWOP represents the faithful implementation of our rate allocation algorithm, since it is not constrained by number of priority levels. As a result, its rate allocation is closest to the desired ratios. Siren with just 6 priority levels shows performance close to that of DWOP.

VII. Related Work

Substantial research exists on the individual components of a fairness control framework i.e. fairness policy, PR, and medium access. However we will focus only on the PR scheme in this section. Priority resolution has been achieved in the MAC layer by various means. Broadly, they can be classified into two approaches – beacon based [26], [5], [30] or backoff based [6], [18]. This bifurcation can be traced back to the original two competing standards for the wireless MAC – 802.11/802.11e [6], which uses a differentiated backoff stratergy, and HIPERLAN/1 [5] which uses beacons.

Vaidya et al. propose a PR scheme, BTPS [30] (Busy Tone Priority Scheduling) supporting two priority levels. In this scheme, nodes are required to listen on three channels during idle periods – a data channel and two narrow-band busy-tone channels. A node with a backlogged high priority packet transmits a busy tone signal (BT1) every M slots. Neighbors that hear BT1 will forward this signal to the node's two-hop neighbors on a different band (BT2). The node's two-hop neighbors with backlogged low priority data that hear BT2 will defer their transmissions. This ensures the transmission of high priority packets. BTPS solves the hidden terminal problem effectively, but requires nodes to listen on multiple channels during idle periods. It is also constrained to two priority levels.

Kanodia et al. propose a dynamic PR algorithm for multihop wireless networks [18]. In this scheme, nodes piggyback the priority of their head of line (HOL) packet in RTS/CTS packets. By overhearing such RTS/CTS packets, nodes become aware of the HOL priorities of the nodes in their neighborhood and use this information to calculate the 802.11 backoffs with which they should access the channel. DWOP [19] is similar to the above scheme, except that it is stricter in its enforcement of packet priorities – nodes are not allowed to send an RTS unless the maximum priority level in their cache corresponds to their HOL packet. Since both schemes rely on overheard information, their performance degrades in lossy channels.

VIII. Conclusion

We take a fresh look at the existing wireless MAC architecture for QoS support and propose new design guidelines to ensure greater flexibility and efficiency. We then propose Siren, a new MAC protocol which fulfils these guidelines and then implement it on a conventional CSMA-based wireless radio. We also propose and implement several different fairness policies on top of Siren demonstrating its effectiveness and flexibility.

References

[1] CC1000 Low Power FSK tranciever, Chipcon Corporation.

[2] CC2420 2.4 GHz IEEE 802.15.4/ZigBee-ready RF Transceiver.

[3] XBOW MICAz Mote Specifications, http://www.xbow.com/.

[4] ZigBee Alliance, IEEE 802.15.4, ZigBee standard, http://www.zigbee.org/.

[5] Radio Equipment and Systems (RES); High Performance Radio Local Area Network (HIPERLAN); Functional Specifications, ETSI, 1995.

[6] IEEE WG, Draft Supplement to Standard for Telecommunications and Information Exchange between Systems-LAN/MAN Specific Requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC), Enhancements for Quality of Service (QoS), 802.11e Draft 4.1, February, 2003.

[7] I. Aad and C. Castelluccia. Differentiation mechanisms for IEEE 802.11. In *INFOCOM '01*.

[8] S. Biswas and R. Morris. Exor: opportunistic multi-hop routing for wireless networks. In *SIGCOMM '05*.

[9] L. Chen, S. Low, M. Chiang, and J. Doyle. Optimal cross-layer congestion control, routing and scheduling design in ad hoc wireless networks. In *INFOCOM '06*.

[10] S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *SenSys '03*.

[11] M. Garetto, J. Shi, and E. W. Knightly. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *MobiCom '05*.

[12] Y. Ge, J. Hou, and S. Choi. An analytic study of tuning system parameters in ieee 802.11e enhanced distributed channel access. *Computer Networks*, 2005.

[13] D. Hadaller, S. Keshav, and T. Brecht. Mv-max: improving wireless infrastructure access for multi-vehicular communication. In *CHANTS '06*.

[14] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda. Performance anomaly of 802.11b. In *INFOCOM '03*.

[15] M. Heusse, F. Rousseau, R. Guillier, and A. Duda. Idle sense: an optimal access method for high throughput and fairness in rate diverse wireless lans. In *SIGCOMM '05*.

[16] B. Hull, K. Jamieson, and H. Balakrishnan. Mitigating congestion in wireless sensor networks. In *SenSys '04*.

[17] K. Jamieson, H. Balakrishnan, and Y. Tay. Sift: a MAC Protocol for Event-Driven Wireless Sensor Networks. In *EWSN '06*.

[18] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly. Distributed multi-hop scheduling and medium access with delay and throughput constraints. In *MobiCom '01*.

[19] V. Kanodia, A. Sabharwal, B. Sadeghi, and E. Knightly. Ordered packet scheduling in wireless ad hoc networks: mechanisms and performance analysis. In *MobiHoc '02*.

[20] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft. Xors in the air: practical wireless network coding. *SIGCOMM Comput. Commun. Rev.*, 36(4):243–254, 2006.

[21] S. Lin, J. Zhang, G. Zhou, L. Gu, J. A. Stankovic, and T. He. Atpc: adaptive transmission power control for wireless sensor networks. In *SenSys '06*.

[22] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. The flooding time synchronization protocol. In *SenSys '04*.

[23] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *SenSys '04*.

[24] J. Shi, T. Salonidis, and E. W. Knightly. Starvation mitigation through multi-channel coordination in csma multi-hop wireless networks. In *MobiHoc '06*.

[25] V. A. Siris and G. Stamatakis. Optimal cwmin selection for achieving proportional fairness in multi-rate 802.11e wlans: test-bed implementation and evaluation. In *WiNTECH '06*.

[26] J. L. Sobrinho and A. S. Krishnakumar. Quality-of-service in ad hoc carrier sense multiple access wireless networks. In *IEEE JSAC Vol.17, No.8, pp. 1353-1368, 1999*.

[27] J. A. Stine and G. de Veciana. A paradigm for quality-of-service in wireless ad hoc networks using synchronous signaling and node states. *IEEE Journal on Selected Areas of Communication*, 22(7), 2004.

[28] I. Tinnirello and S. Choi. Temporal fairness provisioning in multi-rate contention-based 802.11e wlans. In *WOWMOM '05*.

[29] K. Xu, M. Gerla, L. Qi, and Y. Shu. Enhancing tcp fairness in ad hoc wireless networks using neighborhood red. In *MobiCom '03*.

[30] X. Yang and N. H. Vaidya. Priority scheduling in wireless ad hoc networks. In *MobiHoc '02*.

[31] S. Yoo, J. Choi, J. Hwang, and C. Yoo. Eliminating the performance anomaly of 802.11b. volume 3421, pages 1055–1062, 2005.

[32] G. Zhou, T. He, J. Stankovic, and T. Abdelzaher. Rid: radio interference detection in wireless sensor networks. In *INFOCOM '05*.