MMSN: Multi-Frequency Media Access Control for Wireless Sensor Networks

Gang Zhou, Chengdu Huang[†], Ting Yan, Tian He[‡], John A. Stankovic and Tarek F. Abdelzaher[†]
Department of Computer Science, University of Virginia, Charlottesville 22903
{gzhou,ch4pp,ty4k,tianhe,stankovic,zaher}@cs.virginia.edu

Abstract— Multi-frequency media access control has been well understood in general wireless ad hoc networks, while in wireless sensor networks, researchers still focus on single frequency solutions. In wireless sensor networks, each device is typically equipped with a single radio transceiver and applications adopt much smaller packet sizes compared to those in general wireless ad hoc networks. Hence, the multi-frequency MAC protocols proposed for general wireless ad hoc networks are not suitable for wireless sensor network applications, which we further demonstrate through our simulation experiments. In this paper, we propose MMSN, which takes advantage of multi-frequency availability while, at the same time, takes into account the restrictions in wireless sensor networks. In MMSN, four frequency assignment options are provided to meet different application requirements. A scalable media access is designed with efficient broadcast support. Also, an optimal non-uniform backoff algorithm is derived and its lightweight approximation is implemented in MMSN, which significantly reduces congestion in the time synchronized media access design. Through extensive experiments, MMSN exhibits prominent ability to utilize parallel transmission among neighboring nodes. It also achieves increased energy efficiency when multiple physical frequencies are available.

I. INTRODUCTION

As an emerging technology, wireless sensor networks (WSNs) have a wide range of potential applications [1] [2] [3], including environment monitoring, smart buildings, medical care, industry and military applications. Being an essential part of the communication stack, media access control (MAC) has received intense research attention, and a number of solutions have been proposed [4] [5] [6] [7] [8] [9]. While these solutions work well when one physical frequency is used, parallel data transmission when multiple frequencies are available is not considered. On one hand, the radio bandwidth in WSNs is very limited, 19.2Kbps in MICA2 [10] and 250Kbps in MICAz [11] and Telos [12]. On the other hand, the current WSN hardware, such as MICAz and Telos that use CC2420 radio [13], already provide multiple frequencies [10]. So it is imperative to design multi-frequency MAC protocols in wireless sensor networks to take full advantage of parallel transmission to improve network throughput.

In the state-of-the-art research, a significant number of multi-frequency MAC protocols have been proposed, for wireless networks in general. However, these protocols are not

suitable for typical WSN applications. First, to save energy and reduce product cost, each sensor device is usually equipped with a single radio transceiver. This single transceiver can not transmit and receive at the same time, nor can it function on different frequencies simultaneously. This restricted hardware is quite different from more powerful hardware assumed in other wireless systems. For example, protocols [14] [15] are designed for frequency hopping spread spectrum (FHSS) wireless cards, and protocol [16] assumes the busy-tone functionality on the hardware. In protocols [17] [18] [19] [20], the hardware is assumed to have the ability to listen to multiple frequencies at the same time. Second, the network bandwidth in WSNs is very limited and the MAC layer packet size is very small, 30∼50 bytes, compared to 512+ bytes used in general wireless ad hoc networks. Due to the small data packet size, the RTS/CTS control packets in IEEE 802.11 [21] no longer constitute a small overhead that can be ignored. So protocols [22] [23] [24] [25] [26] that are based on IEEE 802.11, and protocols [27] [28] [29] [14] that use RTS/CTS for frequency negotiation are not suitable for WSN applications, even though they exhibit good performance in general wireless ad hoc networks.

In this paper, we propose MMSN, abbreviation for Multi-frequency Media access control for wireless Sensor Networks. MMSN takes full advantage of multiple frequencies and is especially designed to meet WSN requirements. The detailed MMSN design is presented from two aspects: *frequency assignment* and *media access*, and its performance is evaluated through extensive simulation. The main contributions of this work can be summarized as follows:

- To the best of our knowledge, the MMSN protocol is the first multi-frequency MAC protocol especially designed for WSNs, in which each device is equipped with a single transceiver and the MAC layer packet size is very small.
- Instead of using pair-wise RTS/CTS frequency negotiation [27] [28] [29] [14], we propose lightweight frequency assignment, which takes advantage of the static property of many deployed wireless sensor networks [30] [31] [32] [33]. Even though pair-wise frequency negotiation is efficient when devices are highly mobile, it involves unnecessary overhead and is too costly when applied to static WSN applications.

This paper gives a complete study of tradeoffs among physical frequency requirements, potential conflict re-

duction and communication overhead, during frequency assignment. Four optional frequency assignment schemes are proposed for MMSN, which exhibit distinguished advantages in different scenarios.

- We develop new toggle transmission and toggle snooping techniques to enable the single transceiver sensor device to achieve scalable performance, avoiding the non-scalable "one control channel + multiple data channels" design [34]. Also, MMSN has efficient broadcast support, which either is not addressed in [27] or is implemented by repeated link-layer retransmission of broadcast packets enqueued by higher layers in [22].

  Moreover, through strict theoretical analysis, an optimal non-uniform backoff algorithm is derived and its lightweight approximation is implemented in MMSN. Compared with a uniform backoff algorithm, this non-uniform scheme significantly reduces potential conflicts among neighboring nodes.

The rest of this paper is organized as follows: In Section II, we present the motivation of this work. In Section III, the design details of MMSN are explained. In Section IV, extensive experiments are conducted to evaluate MMSN's performance. Finally, in Section V, we give the conclusions and point out future work.

II. MOTIVATION

To obtain a better understanding of the cost that RTS/CTS control packets incur in general wireless ad hoc networks versus WSNs, we choose a typical multi-frequency MAC protocol, the MMAC [27] protocol, proposed for general wireless ad hoc networks, as a case study. In MMAC, periodically transmitted beacons divide time into fixed-length beacon intervals. At the beginning of each beacon interval, there is a small window called the ATIM window, in which the nodes that have packets to send negotiate frequencies with destination nodes. After the ATIM window, nodes that have successfully negotiated frequencies with their destinations can send out data packets using the IEEE 802.11 protocol, i.e. exchanging RTS/CTS before sending out DATA packets. We implement MMAC in GloMoSim [35], a scalable discrete-event simulator developed by UCLA, and observe the performance. We adopt the same experiment set up as in [27]: 100 nodes are randomly placed in a 500m×500m terrain. The transmission range of each node is 250m. Each node has 3 physical frequencies. Forty nodes are randomly chosen to be sources, and 40 nodes are randomly chosen to be destinations. Source nodes generate CBR traffic to destinations with a rate of 10 packets per second. Figure 1 plots the aggregate MAC throughput of the network with different packet sizes.

As can be observed in Figure 1, when the packet size is large, the MMAC protocol with 3 frequencies and a beacon interval of 100ms (the default configuration suggested in [27]) impressively enhances the aggregate MAC throughput by a factor of nearly 2 over IEEE 802.11. This result is consistent with that presented in [27]. However, the performance of both MMAC and IEEE 802.11 degrades when the packet

size reduces. This is because the overhead of using RTS/CTS packets becomes more prominent when the data packet size is small. Moreover, the performance improvement of MMAC over IEEE 802.11 diminishes when the packet size becomes smaller. When the packet size is as small as 32 bytes, IEEE 802.11 has even a slightly higher throughput than MMAC. The reason is when the packet size reduces, more packets *could* be sent in a beacon interval. However, since nodes generally can not switch frequency during a beacon interval, the bandwidth wasted is more severe compared to the case when the packet size is large. Changing the length of the beacon interval could be beneficial, but the effect is two-sided. While lengthening the beacon interval can mitigate the overhead of having a fixed period of frequency negotiation, it deteriorates the bandwidth caused by the requirement that nodes have to stick to the frequency they have negotiated with some of their neighbors. In Figure 1, we also plot the cases with different beacon intervals. We can see that while using a shorter beacon interval (50ms) helps to some extent, MMAC with 3 frequencies still can not even outperform IEEE 802.11 with a single frequency, when the packet size is as small as 64 or 32 bytes. The main observation we make here is that while MMAC is a good multi-frequency MAC protocol for general wireless ad hoc networks where packets usually have large sizes, it is not suitable for WSNs where packets are much smaller.
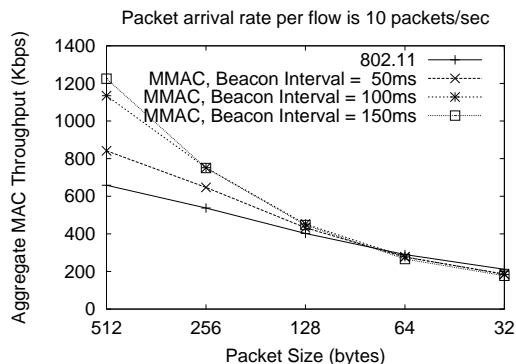

Fig. 1.   Effect of Packet Size on MMAC

III. MMSN PROTOCOL

This section presents the MMSN multi-frequency MAC protocol. MMSN is especially designed for WSNs, which is composed of hundreds of simple devices geographically dispersed in an ad hoc network over a large geographic area. Each device is equipped with a single transceiver and the packet size is very small, 30∼50 bytes. The MMSN protocol consists of two aspects: frequency assignment and media access. The frequency assignment is used to assign different frequencies if enough frequencies exist, or evenly allocate available frequencies if there are more neighbors than available frequencies, to nodes that have potential communication conflicts. MMSN allows users to choose 1 of 4 available frequency assignment strategies. In media access design, nodes that have

potential conflicts coordinate to access the shared physical frequencies, in a distributed way.

A. Frequency Assignment

In frequency assignment, each node is assigned a physical frequency for data reception. The assigned frequency is broadcast to its neighbors, so that each node knows what frequency to use to transmit unicast packets to each of its neighbors. We do not adopt RTS/CTS frequency negotiation, because it involves unnecessary overhead for many deployed wireless sensor networks [30] [31] [32] [33] where devices are generally not mobile. In WSNs, frequency assignment can either be done once at the beginning of the system deployment, or it can be done very infrequently just for adaptation to system aging. In order to reduce communication interference and hence reduce hidden terminal problems [21], nodes within two communication hops[1] are evenly assigned available physical frequencies.

In this section, four optional frequency assignment schemes are put forth: exclusive frequency assignment, even selection, eavesdropping and implicit-consensus. Among these four, exclusive frequency assignment guarantees that nodes within two hops are assigned different frequencies, when the number of frequencies is equal to or greater than the node number within two hops. Implicit-consensus also provides this guarantee, with less communication overhead, but requires more physical frequencies. Even selection and eavesdropping do not provide this guarantee and are designed for use when the number of available frequencies is smaller than the node number within two hops. Among these two, even selection leads to fewer potential conflicts while eavesdropping is more energy efficient. Users of MMSN can choose any one of the four options depending on their WSN attributes. Details of these four schemes are presented in the following subsections.

1) Exclusive Frequency Assignment: In exclusive frequency assignment, nodes first exchange their IDs among two communication hops, so that each node knows its two-hop neighbors' IDs. A simple way to implement this is for each node to broadcast twice. In the first broadcast, each node beacons its node ID, so that each node knows its neighbors' IDs within one communication hop. In the second broadcast, each node beacons all neighbors' IDs it has collected during the first broadcast period. Hence, after the second beacon period, each node gets its neighbors' IDs within two communication hops. Currently, we do not consider radio irregularity and link asymmetry [37] [38] [39] [40]. Readers can refer to [41] [42] [43] for more information about reliability issues in broadcast.

After nodes collect ID information of all neighbors within two hops, they make frequency decisions in the increasing order of their ID values. If a node has the smallest ID among its two communication hops, it chooses the smallest frequency

among available ones, and then beacons the frequency choice within two hops. If a node's ID is not the smallest one among two hops, it waits for frequency decisions from other nodes within two hops that have smaller IDs. After decisions from all those nodes are received, the node chooses the smallest available (not chosen by any of its two-hop neighbors) frequency and broadcasts this choice among two hops.

This scheme guarantees to assign different frequencies to different nodes within any two-hop neighborhood, when the number of frequencies is at least as large as the two-hop node number.

2) Even-Selection: In exclusive frequency assignment, when there are not enough frequencies, it is possible that when a node makes its frequency decision, all physical frequencies have already been chosen by at least one node within two hops. In this case, the exclusive frequency assignment is extended by randomly choosing one of the least chosen frequencies. For convenience, we call this extension *even selection*, which makes an even allocation of available frequencies to all nodes within any two communication hops.

3) Eavesdropping: Even though the even selection scheme leads to even sharing of available frequencies among any two-hop neighborhood, it involves a number of two-hop broadcasts. To reduce the communication cost, we propose a lightweight eavesdropping scheme. In eavesdropping, each node takes a random backoff before it broadcasts its physical frequency decision. During the backoff period, each node records any physical frequency decision overheard. When a node's backoff timer fires, it randomly chooses one of the least chosen frequencies for data reception. Compared with even selection, eavesdropping has less communication overhead, but it also results in more potential conflicts, because it only collects information within one hop for frequency decisions.

4) Implicit-Consensus: When physical frequencies are abundant, the communication overhead in exclusive frequency assignment can be further reduced, while all nodes within any two-hop neighborhood can still be guaranteed to get assigned different frequencies. To achieve this performance, we propose the implicit-consensus scheme, which is inspired by the pseudo random number generator algorithms proposed in the NAMA [44] paper. In NAMA, the pseudo random number generators are used to design distributed time scheduling in TDMA. In this paper, we extend this basic pseudo random number generator idea, proposing a distributed frequency assignment algorithm for multi-frequency MAC designs.

In implicit-consensus, nodes' IDs need to be collected within two hops, in the same way as what is done in exclusive frequency assignment. Then, each node calculates its frequency number with a local computation. In the system, all nodes share the same pseudo random number generator, which is able to generate a unique random number sequence for each specified seed, the node ID here. Algorithm 1 presents the scheme for each node to calculate its frequency number. To assist explanation, node α is taken as an example.

As algorithm 1 states, for each frequency number, each node calculates a random number (Rnd_α) for itself and a

---

[1] In [36], it is pointed out that interference hops (connectivity based on interference relations), rather than communication hops, should be used for this purpose. For simplicity, we use two communication hops in this work. All algorithms proposed here can be easily extended by replacing the two communication hops with two interference hops.
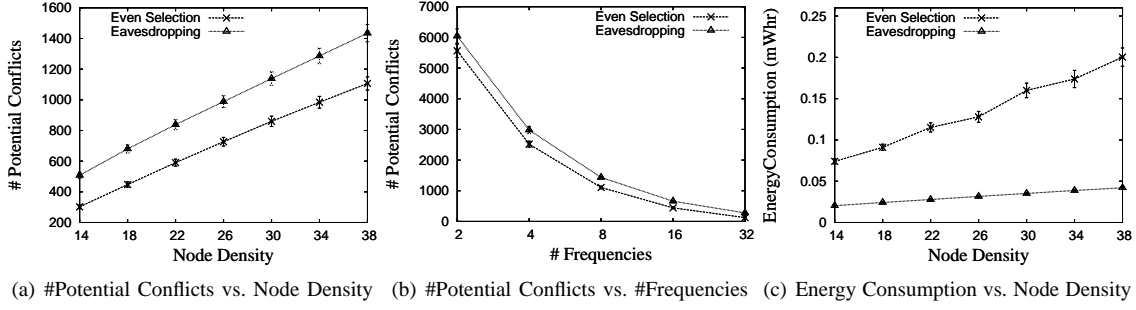
(a) #Potential Conflicts vs. Node Density   (b) #Potential Conflicts vs. #Frequencies   (c) Energy Consumption vs. Node Density

Fig. 2.   Performance Evaluation of Frequency Assignment

Algorithm 1 Frequency Number Computation
_____
Input: Node α's ID (ID_α), and node α's neighbors' IDs within two communication hops.
Output: The frequency number ($FreNum_\alpha$) node α gets assigned.

   $index = 0$; $FreNum_\alpha = -1$;
   repeat
     $Rnd_\alpha = \text{Random}(ID_\alpha, index)$;
     $Found = TRUE$;
     for each node β in α's two communication hops **do**
       $Rnd_\beta = \text{Random}(ID_\beta, index)$;
       if ($Rnd_\alpha < Rnd_\beta$) or ($Rnd_\alpha == Rnd_\beta$ and $ID_\alpha < ID_\beta$) **then**
         $Found = FALSE$; break;
       end if
     end for
     if $Found$ **then**
       $FreNum_\alpha = index$;
     else
       $index$ ++;
     end if
   until $FreNum_\alpha > -1$
_____

random number (Rnd_β) for each of its two-hop neighbors, with the same pseudo random number generator. A node wins the current frequency number if and only if its current random number is the highest among all current random numbers generated by all nodes within two hops. When two random numbers tie, the one with the larger node ID wins. In this way, each node explores all frequency numbers from zero to positive infinity until it finds the frequency for which it has the highest priority. By using the same pseudo random number generator, it is guaranteed that when a node decides that it wins frequency number $FreNum_i$, all nodes within two hops automatically agree with that decision and consensus is implicitly achieved, without any communication.

Here, a question may arise, since each node has a global ID. Why don't we just map nodes' IDs within two hops into a group of frequency numbers and assign those numbers to all nodes within two hops? Unfortunately, this scheme does not work, because a node's ID may get mapped to different frequency numbers in different two-hop neighborhoods. Also, it is not scalable to build a one-to-one mapping between nodes' IDs and all available frequencies, because this makes the frequency requirement depend on the network size, rather than the node density.

In implicit-consensus, when a node (node A) does not win the current frequency number ($FreNum_c$), because its random number is smaller than that of one of its two-hop neighbors (node B), it may happen that this neighbor (node B) has already won a previous frequency number. In this case, node B does not need the current frequency number. Node B should have already terminated its frequency number computation before it takes $FreNum_c$ into consideration, according to the repeat-until loop termination condition in algorithm 1. So, node A keeps trying larger frequency numbers until it finally finds one, while at the same time frequency number $FreNum_c$ is not chosen by any node within this two-hop neighborhood. Accordingly, the finally assigned frequency numbers among two communication hops may not be continuous. There may be holes, and some frequency numbers may not be assigned to any node, which is why the implicit-consensus scheme assumes that the available frequencies are abundant.

With the assigned frequency numbers, each node can easily calculate its physical frequency, with a local mapping. Let's put the available frequencies in a sorted list, $FreList = \{f_0, f_1, \ldots, f_N\}$, in increasing order. If the assigned frequency number is $FreNum_i$, the corresponding physical frequency is mapped to f_{FreNum_i}. After each node gets its physical frequency, it broadcasts this information to its one hop neighbors, so that each node knows what frequency to use to transmit packets to its neighbors.

B. Evaluation of Frequency Assignment

In this section, we compare the performance of even selection and eavesdropping, when available frequencies are not enough and potential conflicts exist. Performance comparison of exclusive frequency assignment and implicit-consensus are not presented, because both of them guarantee that there are no potential conflicts within any two-hop neighborhood.

In the experiments, performance is compared from three aspects. First, we compare the performance when the node

density[2] increases while the number of available frequencies is fixed at 5. We use the number of potential conflicts as the performance metric, which is defined as the total number of node pairs in the system that satisfies the condition: the node pair is within two communication hops and both nodes share the same frequency. Since the two nodes are within two hops, two of their common neighbors may simultaneously transmit packets to them respectively. When they are assigned the same frequency, these two data transmissions interfere with each other, and packet loss may happen. So the number of potential conflicts measures the system's ability of full multi-frequency utilization. Second, besides node density, we also vary the number of available frequencies, to test the performance stability of even selection and eavesdropping. Third, we measure the communication energy consumption of all nodes within the system to compare the cost each scheme pays for its performance. We also explore the cost variation when different node densities are used.

The performance comparison is conducted in GloMoSim [35], in which 289 (17×17) nodes are uniformly deployed in a terrain of 200m×200m square. The radio type is set to RADIO-ACCNOISE [35] and the radio bandwidth is set to 250Kbps. The performance results are illustrated in Figure 2. For each data value we present, its 90% confidence interval is given as well.

As shown in Figure 2 (a), for all the node densities we set from 14 to 38, even selection always performs better than eavesdropping. For instance, when node density is 14, even selection has 302 potential conflicts, which is 40% less than the 507 potential conflicts eavesdropping has. When the node density increases to 38, even selection has 1106 potential conflicts and that is 23% less than the 1434 potential conflicts eavesdropping has. Even selection achieves this good performance because when a frequency decision is made, it is always the case that one of the least loaded frequencies is preferred within two hops. In this way, load is well distributed among all available frequencies within any two-hop neighborhood. However, in eavesdropping, nodes make frequency decisions based on overheard information within only one hop, which leads to a lower performance than even selection. From Figure 2 (a), it is also observed that the number of potential conflicts increases for both even selection and eavesdropping, when the node density increases. This is because the number of frequencies is fixed at 5, so the increased node density results in the increased number of nodes that share the same frequency within two hops.

Besides node density, we also vary the number of available frequencies to compare the performance of even selection and eavesdropping. In Figure 2 (b), the similar phenomenon is observed: even selection performs consistently better than eavesdropping, for all the numbers of frequencies we choose from 2 to 32.

With respect to energy consumption, Figure 2 (c) shows

that even selection consumes more energy than eavesdropping. This is because even selection has two-hop neighbor discovery as well as two-hop broadcasts of frequency decisions, while eavesdropping only has one hop broadcasts.

However, this energy consumption is amortized during data transmission, because in many running sensor network applications [30] [31] [32] [33], sensor devices are generally static, so frequency assignment can either be done once at the beginning of the system deployment, or it can be done very infrequently just for adaptation to system aging. Accordingly, if the specific sensor network system is mostly static and the network congestion is a big issue, even selection is a better choice. On the other hand, if the system topology varies a lot with time and the network is lightly loaded, eavesdropping can be used to save more energy.

C. Media Access Design

After frequency assignment, each node gets a physical frequency for data reception. With the assigned frequencies, nodes cooperate to maximize parallel transmission among neighboring space in media access. To provide efficient broadcast support, nodes are time synchronized [45] during media access. A time slot consists of a broadcast contention period (T_{bc}) and a transmission period (T_{tran}). During the T_{bc} period, nodes compete for the same broadcast frequency and during the T_{tran} period, nodes compete for shared unicast frequencies. The T_{tran} period also provides enough time to actually transmit or receive a broadcast or unicast data packet. The time slot size depends on the number of nodes that compete for the same frequency and the data packet size. The regular time slot size is 3~5ms.

Within one time slot, a node is able to either transmit or receive one packet. Each node first checks the broadcast frequency f_0[3] for receiving or transmitting a broadcast packet. If there is no broadcast packet to transmit or receive, unicast packet transmission and reception are considered. Each node's behavior differs depending on whether it has one packet to transmit or not, as well as whether it has a unicast packet or a broadcast packet to transmit. What follows explains the details.
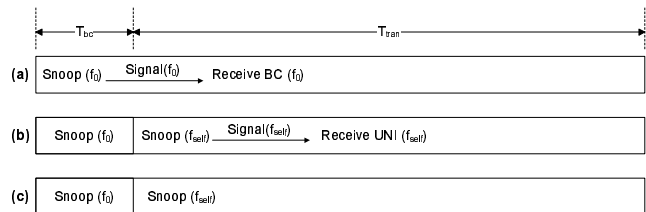

Fig. 3. When a Node Has no Packet to Transmit

1) Has No Packet to Transmit: If a node does not have any packet to transmit within a time slot, it behaves as Figure 3 presents. It first snoops on frequency f_0 during the time period

---

[2]The node density is defined as the number of nodes within one communication hop, and different node densities are configured by setting different radio ranges.

[3]One specific physical frequency is used for broadcast during the T_{bc} period, and this frequency can be reused during the T_{tran} phase for unicast transmission. So all frequencies are fully utilized.

T_{bc}. If the channel is busy, it becomes aware that another node is broadcasting a packet. So it receives the broadcast packet during the rest of the time slot, which is illustrated in case (a). On the other hand, if no signal is sensed during the time period T_{bc}, the node switches to snoop on frequency f_{self}, which is the frequency assigned to it for unicast packet reception. If a signal is sensed in frequency f_{self}, it receives the packet during the rest of the time slot, as shown in case (b). Here, we define $T_{Packet\ Transmission}$ as the time to deliver a packet after it gets the channel, which depends on the packet size and radio bandwidth. A nodes needs to keep on sensing the channel for a possibly incoming unicast packet, until the time left for the current time slot is shorter than $T_{Packet\ Transmission}$, as shown in case (c). When the time left for the current time slot is less than $T_{Packet\ Transmission}$, no neighboring nodes will send a packet to this node, so it turns off carry sensing until the start of the next time slot to save energy.
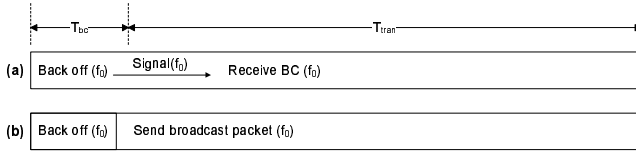
Fig. 4.   When a Node Has a Broadcast Packet to Transmit

2) Has a Broadcast Packet to Transmit: If a node has a broadcast packet for transmission, it may have two different behaviors as illustrated in Figure 4. At the beginning of the time slot, the node uses frequency f_0, which is specified for transmitting and receiving broadcast packets. It first sets a random backoff within the time period T_{bc}. If it senses any signal during the backoff period, it becomes aware that another node is broadcasting a packet. In this case (case (a)), the node spends the rest of the time slot receiving the broadcast packet. There is another case, case (b), in which the node does not sense any signal in frequency f_0, during its backoff time period. In this scenario, a broadcast packet is sent out from this node, after the backoff timer fires.
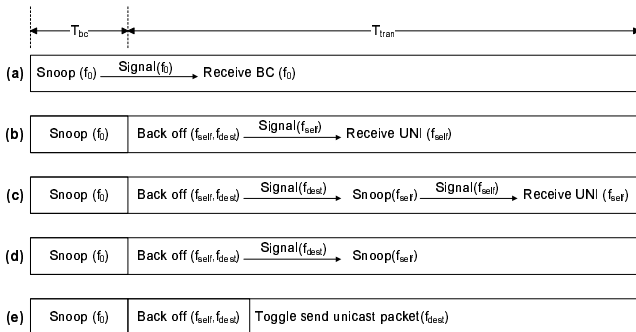

Fig. 5.   When a Node Has a Unicast Packet to Transmit

3) Has a Unicast Packet to Transmit: Figure 5 illustrates the different behaviors a node may take, if it has a unicast packet for transmission. The node first listens to the broadcast frequency f_0 during time period T_{bc}. If it senses any signal

during T_{bc}, which must be a broadcast packet, the node spends the rest of the time slot receiving the broadcast packet, as shown in case (a).

Cases (b)(c)(d)(e) illustrate the other four scenarios in which the node does not sense any broadcast signal during the time period T_{bc}. In these cases, the node takes a random backoff within the time period $T_{tran} - T_{Pakcet\ Transmission}$. During the backoff time period, the node snoops on two frequencies. On one hand, it snoops on frequency f_{self}, which is assigned to it for data reception, to get prepared for a possibly incoming unicast packet. On the other hand, it also snoops on frequency f_{dest}, which is assigned to the destination node of its unicast packet for data reception. If frequency f_{dest} is sensed busy, it can be aware that another node is transmitting a unicast packet to the same destination node, and it can choose not to transmit the unicast packet in the current time slot to avoid collisions. The node snoops on these two frequencies alternately, and we call this scheme *toggle snooping*, which is discussed in detail in subsection III-C.4.

During toggle snooping, if the node senses any signal on frequency f_{self}, it gets to know that it itself is the destination of an incoming unicast packet. So it stops toggle snooping to receive the data packet, which is illustrated in case (b). During the toggle snooping, the node may also sense a signal on frequency f_{dest}. When frequency f_{dest} is sensed busy, the node gets to know that another node is competing for the shared frequency, by sending a unicast packet to the same destination node. In this case, the node stops toggle snooping and switches to snoop on frequency f_{self} only. It gives up transmitting a unicast packet in this time slot and prepares to receive possible data packet transmitted to it. So if any signal is sensed in frequency f_{self}, as shown in case (c), it receives the unicast packet during the rest of the time slot. Before the node senses any signal in frequency f_{self}, it keeps sensing the frequency until the time left for the current time slot is $T_{Packet\ Transmission}$, as shown in case (d). When the left time for the current time slot is shorter than $T_{Packet\ Transmission}$, it turns off carry sensing to save energy.

If the node does not sense any signal in both frequency f_{self} and f_{dest} during the backoff time period, as shown in case (e), it sends out a unicast packet with the toggle transmission technique, which is illustrated in Figure 6.
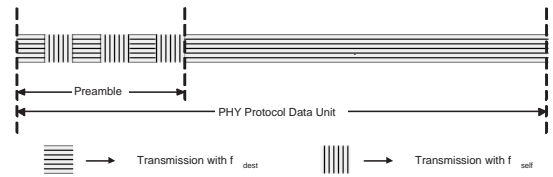

Fig. 6.   Toggle Transmission

As Figure 6 illustrates, the preamble bytes of the physical layer protocol data unit (PPDU) is transmitted with two frequencies, f_{self} and f_{dest}, in an alternating way. The rest of the PPDU is transmitted to the destination node in frequency f_{dest}. The toggle transmission scheme is useful to reduce collisions.

As shown in Figure 7 (a), when node B is transmitting a unicast packet to node C with the toggle transmission technique, the preamble transmitted in frequency f_{self} informs other nodes that this channel is busy, so that any node that wants to send a packet to node B can back off. On the other hand, the preamble transmitted in frequency f_{dest} informs any node that wants to send a data packet to node C to back off and avoid possible collisions. The relation of toggle transmission and toggle snooping is analyzed in the following subsection.

4) Toggle Snooping and Toggle Transmission: When a node has a unicast packet for transmission, toggle snooping is used during the T_{tran} period and the node snoops on two frequencies alternatingly: the frequency it uses for data reception (f_{self}), and the frequency the destination node of its unicast packet uses for data reception (f_{dest}). The time a node takes to snoop on both of the two frequencies for one round is called the toggle snooping period, represented by parameter T_{TS}. In toggle transmission, a node transmits the preamble bytes of the PPDU with two frequencies, the frequency the node itself uses for data reception (f_{self}) and the frequency the destination node of the unicast packet uses for data reception (f_{dest}). The transmitter switches between these two frequencies alternatingly and the time the node sweeps the two frequencies for one round is called the toggle transmission period, represented by parameter T_{TT}.
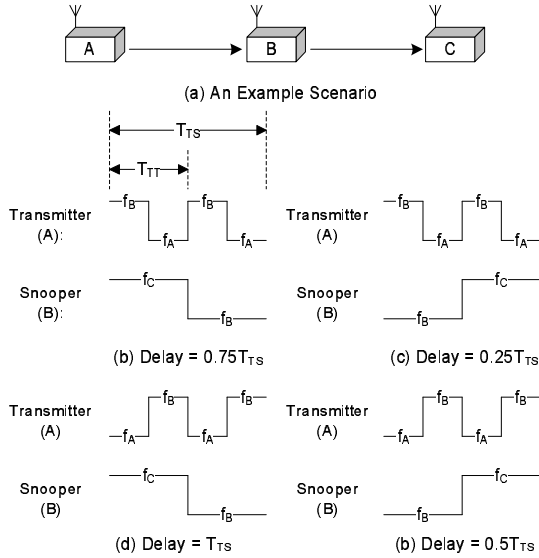

Fig. 7. Toggle Snooping

In the MMSN protocol, we let $T_{TS} = 2 \times T_{TT}$, so that when one node sends out a packet using the toggle transmission scheme, any other node that is snooping using the toggle snooping scheme is able to detect this transmission within a maximal delay of T_{TT}, if toggle transmission and toggle snooping have any shared frequency. With the help of Figure 7, we make this point more clear. In Figure 7 (a), node A uses frequency f_A for packet reception and it has a unicast packet to send to node B. Node B uses frequency f_B for packet reception and it has a unicast packet to send to node C, which

uses frequency f_C for packet reception. During the T_{tran} time period, both A and B set up backoff timers and snoop on two frequencies. Node A snoops on frequency f_A and f_B and node B snoops on frequency f_B and f_C. Let's suppose that node A's timer fires first. So node A switches from toggle snooping to toggle transmission, while node B is still in the toggle snooping state. In different application scenarios (not only limited to the case in Figure 7 (a)), node B may take different time delays to become aware that node A is transmitting, as shown in Figure 7 (b)(c)(d)(e). In the scenario presented in case (b), node B is able to detect node A's transmission in frequency f_B after the time delay of $0.75\,T_{TS}$. In case (c), the delay to detect node A's transmission is $0.25\,T_{TS}$. In case (d) and (e), the delays are T_{TS} and $0.5\,T_{TS}$, respectively.

According to the above analysis, it is guaranteed that when one node transmits a packet using the toggle transmission scheme, the maximum time delay for another node, which uses the toggle snooping scheme, to detect the transmission is T_{TS}. Accordingly, if the backoff timer used in the slotted time period in Figure 5 is only allowed to fire at the end of a toggle snooping period, a node whose backoff timer fires after the previous one can have enough time to detect the previous node's transmission, and hence abandon its transmission in the current time slot to reduce congestion.

D. IMPLICATION OF BACKOFF ALGORITHMS

In media access, neighboring nodes may compete for the same physical frequency in both the broadcast contention period (T_{bc}) and the transmission period (T_{tran}), as explained in Section III-C. To reduce congestion, random backoff is needed for both broadcast and unicast transmission. Taking unicast backoff as an example, we give theoretical analysis to prove that a uniform backoff algorithm is not a good choice for the time synchronized media access in MMSN, and a non-uniform backoff algorithm achieves better performance. We derive an optimal non-uniform backoff algorithm, and choose its lightweight approximation for implementation in MMSN. All results derived here also apply for the broadcast transmission in MMSN.

During the backoff in the T_{tran} period in Figure 5, the time slot is further divided into small time slices. As explained in the previous section, each time slice has the length of T_{TS} and each backoff timer is only allowed to fire at the end of a time slice. If any two nodes choose the same backoff time slice, there is a collision. In order to minimize the probability of collision, we derive an optimal bound and a simple suboptimal distribution of the backoff time slices.

First we derive the optimal probability distribution $P(t)$ of backoff time slice t to minimize the probability of collisions when two nodes attempt to grab the same time slice after backoff. $P(t)$, $t = 0, 1, ..., T$, denotes the probability that a node attempts to grab time slice t and T is the maximum backoff time slice. Obviously $0 \le P(t) \le 1$ and $\sum_{t=0}^{T} P(t) = 1$. We assume that each node independently selects the backoff time slice conforming to the same distribution.

According to the analysis in Section III-C, in a time slot, the node that selects the earliest backoff time slice gets the physical channel, and all nodes whose backoff timers fire later should abandon their transmission. Hence, a node successfully grabs time slice t if all other nodes attempt to grab time slices after t. If at least two nodes in the same neighborhood attempt to grab the same earliest time slice, there is a collision. We need to find the probability distribution $P(t)$ to maximize P_{nc}, the probability that there is only one node that grabs the earliest time slice, to avoid collisions as much as possible. Assuming the earliest time slice is i, $0 \leq i \leq T-1$, and there are N nodes in the neighborhood, the probability that one and only one node attempts to grab this time slice and all other nodes attempts to grab later time slices is $N \cdot P(i) \cdot S_{i+1}^{N-1}$, where $S_{i+1} = \sum_{t=i+1}^{T} P(t)$. Considering all possible earliest time slices, we have

$$P_{nc} = \sum_{i=0}^{T-1} N \cdot P(i) \cdot S_{i+1}^{N-1}.$$

Now we apply a recursive approach to decide the optimal probability distribution $P(t)$. First we assume that the values for $P(t)$, $t = 0, ..., T-2$, are already known. From the constraints that the sum of all $P(t)$'s is 1, $S_{T-1} = P(T-1) + P(T)$ is also known. The question is how to divide S_{T-1} between $P(T-1)$ and $P(T)$ to maximize P_{nc}. This division only affects the term $N \cdot P(T-1) \cdot P(T)^{N-1}$ in the calculation of P_{nc}. The other terms are not affected by the way S_{T-1} is divided. For simplicity we denote $P(T)$ as a and $P(T-1)$ as b. The first order condition for maximizing is

$$\frac{d}{da}(Nba^{N-1}) = N(N-1)S_{T-1}a^{N-2} - N^2 a^{N-1} = 0,$$

and we have

$$S_T = P(T) = a = k_T S_{T-1},$$

where $k_T = \frac{N-1}{N}$.

We omit the validation of the second order condition for brevity, but for $N \geq 2$, the above equation does give a maximized result

$$N \cdot P(T-1) \cdot P(T)^{N-1} = k_T^{N-1} S_{T-1}^N.$$

Then we consider the division of probability S_{T-2} between $P(T-2)$ and S_{T-1} assuming that the values for $P(t)$, $t = 0, ..., T-3$ are known. For simplicity we denote $P(T-2)$ as c. The terms affected by this division are only

$$NcS_{T-1}^{N-1} + k_T^{N-1}S_{T-1}^N.$$

The first order condition is

$$N(N-1)S_{T-2}S_{T-1}^{N-2} + (k_T^{N-1} - N)NS_{T-1}^{N-1} = 0,$$

and therefore

$$S_{T-1} = k_{T-1}S_{T-2},$$

where $k_{T-1} = \frac{N-1}{N-k_T^{N-1}}$. Then we obtain the optimal value of the sum of the two terms:

$$NcS_{T-1}^{N-1} + k_T^{N-1}S_{T-1}^N = k_{T-1}^{N-1}s_{T-2}^N.$$

With the similar approach we get the recursive formulas for S_t and k_t as follows.

$$S_{t+1} = k_{t+1}S_t,$$

where $t = 0, ..., T-1$, and $S_0 = 1$.

$$k_{T-t-1} = \frac{N-1}{N - k_{T-t}^{N-1}},$$

where $t = 0, ..., T-2$, and $k_T = \frac{N-1}{N}$.

Therefore, the optimal distribution $P(t)$ is

$$P(t) = S_t - S_{t+1},$$

where $t = 0, ..., T-1$, and $P(T) = S_T$.

The optimal distribution gives an optimal bound of the non-collision probability. However, the distribution depends on the number of competing nodes, which may vary from neighborhood to neighborhood in deployed systems. Also, the process of computing the distribution is complicated and hence too costly for power-limited sensor devices. Accordingly, if a simple solution can provide a non-collision probability close to the optimal bound, it is more favorable. We propose a suboptimal distribution to be used by each node, which is easy to compute and does not depend on the number of competing nodes. A natural candidate is an increasing geometric sequence, in which

$$P(t) = \frac{b^{\frac{t+1}{T+1}} - b^{\frac{t}{T+1}}}{b - 1}, \tag{1}$$

where $t = 0, ..., T$, and b is a number greater than 1.

The problem is which value of b should be chosen. We choose various b values to calculate the corresponding non-collision probabilities and compare them with that from the optimal $P(t)$. To be consistent with the evaluation section, we choose the same number of time slices and node densities. The results are shown in Figure 8. From the figure we can see that if we choose $b = 1000$, the difference between the simple solution's non-collision probability and that of the optimal $P(t)$ is smaller than 6% for the node densities we choose and $T = 33$, which is the number we use in the simulation.
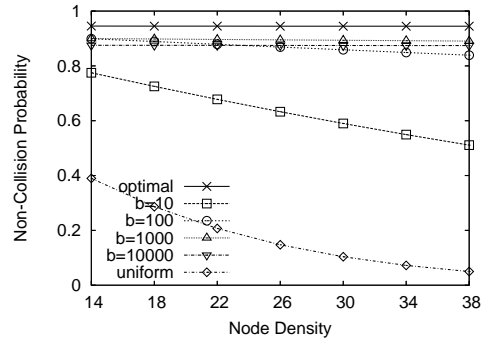

Fig. 8. Non-Collision Probability with Various b Values

A similar approach is also proposed in [46], which minimizes collisions in slotted CSMA. We deem that the optimal

solution is more relevant for our MAC than for slotted CSMA. The slice time for slotted CSMA can be chosen as small as the sum of propagation delay, detection time and other processing delays, which are in microseconds typically. Compared with the maximum backoff time, the slice time is orders of magnitudes smaller, so the number of slices is large. When the slice number approaches infinity, the non-collision probability for a uniform distribution is

$$\lim_{T \to \infty} P_{nc} = \lim_{T \to \infty} (N \sum_{i=0}^{T-1} \frac{1}{T+1} (\frac{T-i}{T+1})^{N-1}).$$

Let $a = \frac{T-i}{T+1}$, from the definition of the Riemann integral, we have

$$\lim_{T \to \infty} P_{nc} = N \int_0^1 a^{N-1} da = N \cdot \frac{1}{N} = 1.$$

Therefore, when the slice number $T+1$ approaches positive infinity, the non-collision probability approaches 1, which means even the uniform distribution gives a very small chance of collision. Calculation shows that if we have 1000 time slices, even when 200 nodes compete, the non-collision probability for a uniform distribution is still above 90%. Since the slice number we use in MMSN is much smaller ($T+1 = 34$ in our simulation), to reduce protocol overhead, the suboptimal approach shows a significant performance improvement over the uniform distribution, as shown in Figure 8.

In our algorithm, we use the suboptimal approach for simplicity and generality. We need to make the distribution of the selected backoff time slice at each node conform to that shown in Equation (1). It is implemented as follows: first a random variable α with a uniform distribution within the interval $[0, 1)$ is generated on each node; then time slice i is selected according to the following equation:

$$i = \lfloor (T+1) \log_b [\alpha(b-1) + 1] \rfloor.$$

It can be easily proven that the distribution of i conforms to Equation (1).

IV. PERFORMANCE EVALUATION

We implement MMSN in GloMoSim [35] and conduct extensive experiments to evaluate its performance and compare it with CSMA as well. In this evaluation, MMSN uses even selection for frequency assignment, since it results in fewer potential conflicts. For this performance evaluation, three groups of experiments are designed. In the first group, different traffic patterns are used. In the second group, different system loads are considered, and in the third group of experiments, the node density is varied.

For all the three groups of experiments, four performance metrics are adopted: aggregate MAC throughput, packet delivery ratio, channel access delay, and energy consumption. The aggregate MAC throughput measures the performance gain and is calculated as the total amount of useful data successfully delivered through the MAC layer in the system per unit time. The packet delivery ratio is calculated as the

| TERRAIN	(200m×200m) Square
Node Number	289
Node Placement	Uniform
Application	Many-to-Many/Gossip CBR Streams
Payload Size	32 Bytes
Routing Layer	GF
MAC Layer	CSMA/MMSN
Radio Layer	RADIO-ACCNOISE
Radio Bandwidth	250 Kbps
Radio Range	20m~45m

ratio of the total number of data packets successfully delivered by the MAC layer over the total number of data packets the network layer requests the MAC to transmit. The channel access delay measures the time delay a data packet from the network layer waits for the channel before it gets sent out. The energy consumption reflects the cost each protocol pays to achieve its performance, which is calculated as the energy consumed to successfully deliver a useful data byte. Since we have measured the cost for each frequency assignment scheme in Section III-B and this energy consumption is amortized during data transmission, it is no longer counted here.

During all the experiments, the Geographic Forwarding (GF) [47] routing protocol is used, and simulation is configured according to the settings in Table I. For each data value we present in the results, we also give its 90% confidence interval.
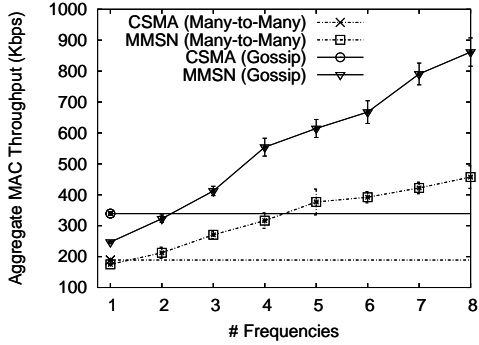
A. Performance Evaluation with Different Traffic Patterns

In the first group of experiments, two different traffic patterns are used, many-to-many and gossip traffic patterns. The many-to-many traffic pattern is used to simulate the typical sensor network application scenario: multiple sensor nodes report their readings to multiple base stations over multiple hops. Since the routing design affects the contention level at the MAC layer (e.g., hot spots), the MAC performance is more statistically valid when a simulation can isolate the effect from the routing layer. Therefore, we also evaluate the MAC performance with the gossip traffic pattern, in which each node only communicates with its neighbors. For both of these two traffic patterns, we increase the number of available frequencies, to observe the performance variation. In this group of experiments, 50 CBR streams are used and the node density is set to 38, by configuring the radio range to 40m. To achieve meaningful results, we evaluate the performance when the packet delivery ratio in the MAC layer is reasonably high, higher than 93%. The small amount of packet loss is due to hidden terminal problems [21].

The performance results illustrated in Figure 9 confirm MMSN's scalability. When the number of frequencies increases from 1 to 8 and the gossip traffic is used, (a) illustrates that the packet delivery ratio increases from 95.4% to 98.1%, (b) shows that the aggregate MAC throughput increases from 246.9 Kbps to 861.8 Kbps, (c) informs that the average channel access delay decreases from 0.069s to
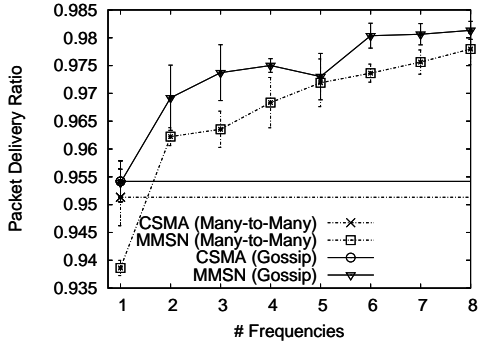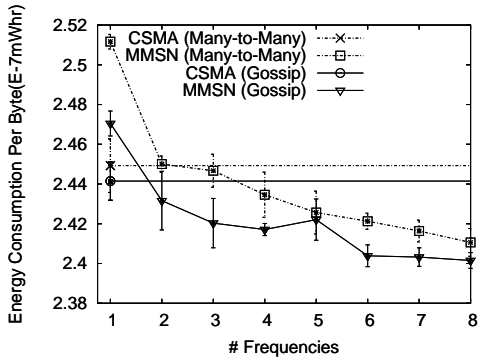
(a) Packet Delivery Ratio in MAC

(b) Aggregate Throughput in MAC
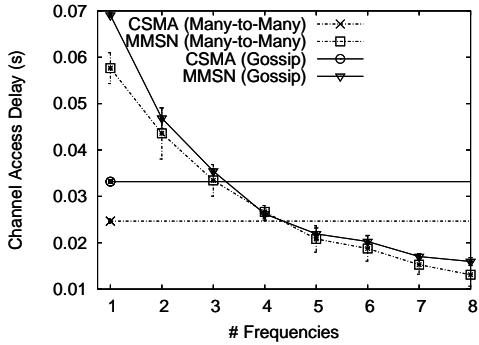
(c) Average Channel Access Delay

(d) Energy Consumption Per Delivered Data Byte

Fig. 9. Performance Evaluation with Different #Physical Frequencies

0.016s, and (d) states that MMSN becomes more energy efficient: the energy consumption per byte of successfully de-

livered data decreases from 2.47×10^{-7} mWhr to 2.40×10^{-7} mWhr. Similar performance increase is also exhibited when many-to-many traffic pattern is used. MMSN's performance increases, because available physical frequencies are evenly shared within two-hop neighboring nodes, and the increase of available frequencies leads to a higher degree of parallel data transmission within each neighborhood. When more physical frequencies are used, more nodes are able to conduct simultaneous transmission in the deployed system without collisions, so the aggregate MAC throughput increases. Plus, fewer nodes are assigned to use the same frequency within two hops. So communication interference decreases, which leads to less backoff and decreased channel access delay. Also, the decreased communication interference leads to less packet loss, and more useful data bytes are successfully delivered with the same amount of energy. On the other hand, MMSN does not achieve 8 times performance improvement when 8 frequencies are used compare to the case when one frequency is used. This is due to the fundamental hardware limitation of using a single transceiver in each sensor device.

Compared with CSMA, MMSN has similar or a little lower performance when the number of frequencies is small. This is because MMSN has a fixed backoff time period allocated within each time slot, while CSMA can fire the backoff timers at any time within the backoff window. However, when the number of frequencies increases, more parallel transmission within each neighborhood occurs and it results in more gains than the cost paid due to the fixed backoff period, and MMSN outperforms CSMA.

We are also aware that MMSN has constantly increasing aggregate MAC throughput when the gossip traffic pattern is used, while the speed of throughput increase slows down when the many-to-many traffic pattern is used. This is because the many-to-many traffic consists of a number of many-to-one traffic, in which multiple nodes transmit data packets to the same destination node. In this case, all these transmitters use the same physical frequency that the destination node gets assigned, and hence there is no potential parallel transmission that can be utilized. This is also one major difference between the single-transceiver[4] multi-frequency MMSN protocol and the multi-transceiver multi-frequency protocols proposed in [17] [18] [19].

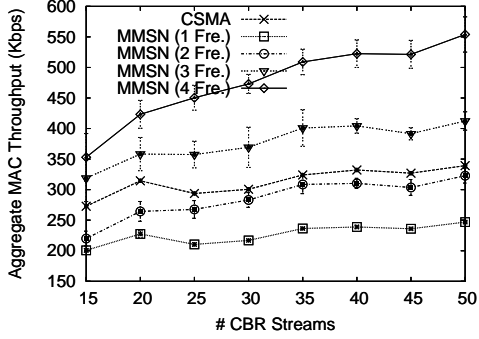### B. Performance Evaluation with Different System Loads

In the second group of experiments, we explore MMSN's performance when different system loads are used, which are generated by different numbers of CBR streams. To analyze performance scalability, we conduct all experiments with different numbers of frequencies as well. In the experiments, the node density is set to 38, and the gossip traffic pattern is used.

As Figure 10 shows, for all the system loads we configure from 15 CBR streams to 50 CBR streams, it is observed that MMSN always exhibits better performance when more
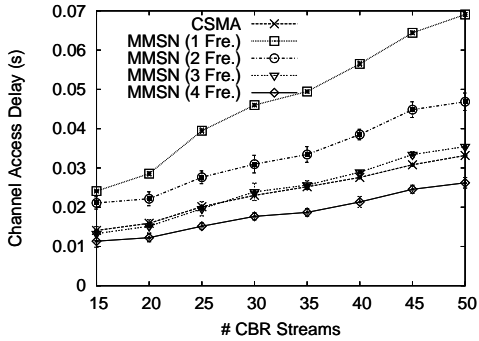
---

[4]One solution is for each base state to have multiple transceivers. The multiple transceivers snoop on different frequencies, so that the base station can receive simultaneous data reporting from multiple nodes.
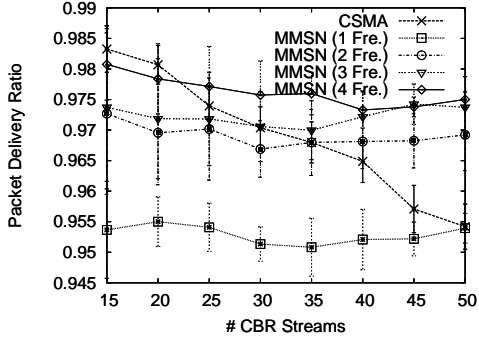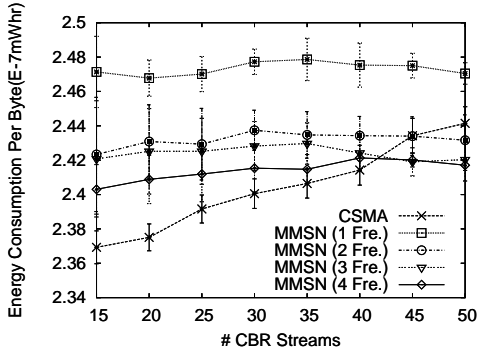
(a) Packet Delivery Ratio in MAC

(b) Aggregate Throughput in MAC

(c) Average Channel Access Delay

(d) Energy Consumption Per Delivered Data Byte

Fig. 10.   Performance Evaluation with Different System Loads

frequencies are used, which is consistent with the result presented in the previous group of experiments. For example, as shown in Figure 10, when the number of frequencies

increases from 1 to 4 and 40 CBR streams are used, MMSN's packet delivery ratio increases from 95.2% to 97.3% in (a). At the same time, MMSN's aggregate MAC throughput increases by 119% from 239Kbps to 523Kbps as shown in (b), and the channel access delay decreases to 0.021s, which is 37.5% of the delay when only 1 frequency is available, as shown in (c). In such a case, (d) also informs that MMSN's energy consumption for each successfully delivered data byte decreases from 2.48×10^{-7}mWhr to 2.42×10^{-7}mWhr. MMSN achieves improved performance when the number of frequencies increases, because the increased frequencies lead to increased parallel transmission within the same neighboring space and to decreased congestion for the same physical frequency.

Figure 10 (a) also shows that CSMA has a decreased packet delivery ratio from 98.3% to 95.4%, while MMSN does not have such an obvious packet loss. This is because the non-uniform backoff algorithm design is more tolerant to the system load variation than the uniform backoff algorithm. The sharply increased system load, from 15 CBR streams to 50 CBR streams, leads to more congestion and more packet loss in CSMA while the slotted backoff is not impacted as much. In (b), the aggregate MAC throughput increases with the increase of system load, because more nodes get involved in communication and more parallel data transmission occurs. In addition, the increased nodes becoming involved in communication result in increased congestion and hence increased channel access delay increases in (c). Since CSMA is more sensitive to system load and has lower packet delivery ratio, it is less energy efficient when the system load increases, while MMSN' packet delivery ratio is more tolerant to system load and hence does not exhibit apparent decrease of energy efficiency.

For similar reasons as explained in the previous experiments, MMSN is observed to have a lower performance than CSMA when there is only one, or two in some cases, physical frequencies available as shown in Figure 10. However, MMSN outperforms CSMA when three or more frequencies are used, which is also exhibited in Figure 10.

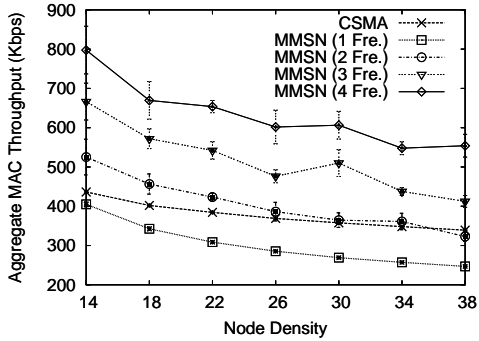### C. Performance Evaluation with Different Node Densities

In many deployed sensor network systems [30] [31] [32] [33], providing node redundancy is an efficient and effective method to increase the system lifetime. So, in the third group of experiments, we evaluate MMSN's performance when different node densities are utilized. The node density is increased from 14 to 38, by configuring different radio ranges, and a gossip traffic pattern is used that consists of 50 CBR streams. We also measure the performance difference when different numbers of frequencies are used as well.

Once again, the experimental results confirm that MMSN always achieves a higher performance when more frequencies are available, which can be observed in Figure 11 (a)~(d). The corresponding reasons can be found in the first two groups of experiments and are not repeated here.
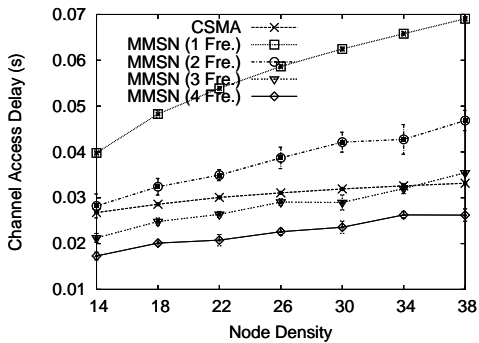
From Figure 11 (b), it is observed that the aggregate MAC throughputs in both CSMA and MMSN decrease with the
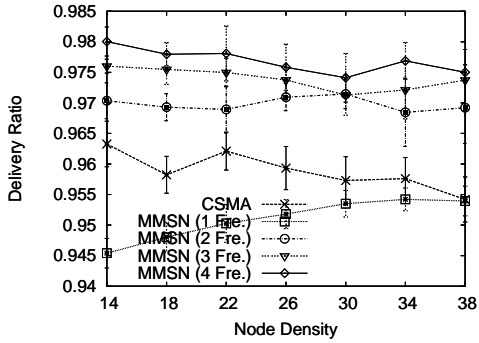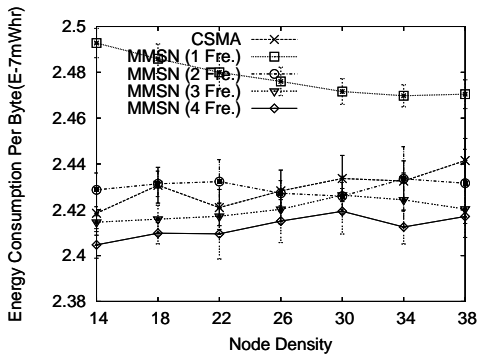
(a) Packet Delivery Ratio in MAC

(b) Aggregate Throughput in MAC

(c) Average Channel Access Delay

(d) Energy Consumption Per Delivered Data Byte

Fig. 11.   Performance Evaluation with Different Node Density

increase of node density. This is because when node density increases, the same number of frequencies are shared by

more nodes within two hops. When the same percentage of nodes participate in communication, congestion is increased and hence backoff and channel access delay are increased, as shown in Figure 11 (c). We do not observe consistent trends for packet delivery ratio variation in (a) and energy consumption variation in (d), when the number of frequencies is greater than one and the node density is increased from 14 to 38. But we do notice that when there is only one frequency, the packet delivery ratio of MMSN increases in (a), with the increase of node density. We think this is because of decreased hidden terminal problems, when the radio range gets increased to increase node density, while at the same time the system topology is fixed to be 200m×200m. When the number of frequencies increases, this effect becomes very small and no similar trend is observed. Also, because of the increased packet delivery ratio, the energy consumption becomes more efficient as shown in (d), when MMSN uses one frequency and the node density increases from 14 to 38.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we propose the first effort to design a multi-frequency MAC protocol for wireless sensor network applications, in which each device adopts a single radio transceiver. The different MAC design requirements for wireless sensor networks and general wireless ad hoc networks are compared, and a complete WSN multi-frequency MAC design (MMSN) is put forth. During the MMSN design, we analyze and evaluate different choices for frequency assignment, and also discuss the non-uniform backoff algorithms for the slotted media access design. Finally, we evaluate MMSN's performance through extensive experiments, and the performance results show that MMSN exhibits prominent ability to utilize parallel transmission among neighboring nodes. MMSN also achieves increased energy efficiency when multiple physical frequencies are available.

In the future, we plan to implement MMSN in a large scale running sensor network system and evaluate its performance with different sensor devices. In addition, the current work assumes that we have access to multiple well separated channels. In future work, we also plan to extend MMSN to the case when the multiple channels have partially overlapping [48] frequency bandwidths.

ACKNOWLEDGEMENT

REFERENCES

[1] D. Culler, D. Estrin, and M. Srivastava, "Overview of Sensor Networks," in *IEEE Computer, Special Issue on Sensor Networks*, August 2004.
[2] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," in *ACM MobiCom 1999*, August 1999.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: a Survey," *Computer Networks*, vol. 38, pp. 393–422, 2002.

[4] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Median Access for Wireless Sensor Networks," in *ACM SenSys 2004*, November 2004.

[5] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *IEEE INFOCOM 2002*, June 2002, pp. 1567–1576.

[6] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves, "Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks," in *ACM SenSys 2003*, November 2003.

[7] T. Dam and k. Langendoen, "An Adaptive Evergy-Sufficient MAC Protocol for Wireless Sensor Networks," in *ACM SenSys 2003*, November 2003.

[8] A. Woo and D. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks," in *ACM MobiCom 2001*, July 2001.

[9] A. El-Hoiyi, J.-D. Decotignie, and J. Hernandez, "Low Power MAC Protocols for Infrastructure Wireless Sensor Networks," in *The Fifth European Wireless Conference*, February 2004.

[10] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System Architecture Directions for Networked Sensors," in *The Nineth International Conference on Architectural Support for Programming Languages and Operating Systems*, November 2000, pp. 93–104.

[11] "XBOW MICA2 Mote Specifications," http://www.xbow.com.

[12] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling Ultra-Low Power Wireless Research," in *ACM/IEEE IPSN/SPOTS 2005)*, April 2005.

[13] "CC2420 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver," http://www.chipcon.com.

[14] Z. Tang and J.J. Garcia-Luna-Aceves, "Hop-Reservation Multiple Access (HRMA) for Ad-Hoc Networks," in *IEEE INFOCOM 1999*, March 1999.

[15] A. Tyamaloukas and J. J. Garcia-Luna-Aceves, "Channel-Hopping Multiple Access," in *IEEE ICC 2000*, 2000.

[16] J. Deng and Z. Haas, "Dual Busy Tone Multiple Access (DBTMA): A New Medium Access Control for Packet Radio Networks," in *IEEE ICUPC*, October 1998.

[17] A. Nasipuri, J. Zhuang, and S. R. Das, "A Multichannel CSMA MAC Protocol for Multihop Wireless Networks," in *IEEE WCNC*, September 1999.

[18] S.-L. Wu, C.-Y. Liu, Y.-C. Tseng, and J.-P. Shen, "A New Multi-Channel MAC Protocol with On-Demand Channel Assignment for Multi-Hop Mobile Ad Hoc Networks," in *I-SPAN*, 2000.

[19] A. Nasipuri and S. R. Das, "Multichannel CSMA with Signal Power-based Channel Selection for Multihop Wireless Networks," in *IEEE Vehicular Technology Conference*, September 2000.

[20] M. Caccaco, L. Y. Zhang, L. Sha, and G. Buttazzo, "An Implicit Prioritized Access Protocol for Wireless Sensor Networks," in *IEEE RTSS 2002*, December 2002.

[21] "IEEE 802.11, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification," 1999, ANSI/IEEE Std. 802.11, 1999.

[22] P. Bahl, R. Chancre, and J. Dungeon, "SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad-Hoc Wireless Networks," in *ACM MobiCom 2004*, September 2004.

[23] A. Raniwala and T. Chiueh, "Architecture and Algorithm for an IEEE 802.11-Based Multi-Channel Wireless Mesh Network," in *IEEE INFOCOM 2005*, March 2005.

[24] A. Adya, P. Bahl, J. Padhye, A.Wolman, and L. Zhou, "A Multi-Radio Unification Protocol for IEEE 802.11 Wireless Networks," in *IEEE Broadnets 2004*, 2004.

[25] F. Fitzek, D. Angelini, G. Mazzini, , and M. Zorzi, "Design and Performance of an Enhanced IEEE 802.11 MAC Protocol for Multihop Coverage Extension," in *IEEE Wireless Communications*, 2003.

[26] J. Li, Z. J. Haas, M. Sheng, , and Y. Chen, "Performance Evaluation of Modified IEEE 802.11 MAC for Multi-Channel Multi-Hop Ad Hoc Network," in *IEEE AINA 2003*, 2003.

[27] J. So and N. Vaidya, "Multi-Channel MAC for Ad-Hoc Networks: Handling Multi-Channel Hidden Terminal Using A Single Transceiver," in *ACM MobiHoc 2004*, May 2004.

[28] N. Jain and S. R. Das, "A Multichannel CSMA MAC Protocol with Receiver-Based Channel Selection for Multihop Wireless Networks," in *IEEE IC3N*, October 2001.

[29] A. Tzamaloukas and J.J. Garcia-Luna-Aceves, "A Receiver-Initiated Collision-Avoidance Protocol for Multi-Channel Networks," in *IEEE INFOCOM 2001*, April 2001.

[30] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson., "Wireless Sensor Networks for Habitat Monitoring," in *ACM WSNA 2002*, September 2002.

[31] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A Wireless Sensor Network For Structural Monitoring," in *ACM SenSys 2004*, November 2004.

[32] T. He, S. Krishnamurthy, J. A. Stankovic, T. F. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh, "Energy-Efficient Surveillance System Using Wireless Sensor Networks," in *ACM MobiSys 2004*, June 2004, pp. 270–283.

[33] G. Simon, M. Maróti, Á. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton, "Sensor Network-Based Countersniper System," in *ACM SenSys 2004*, November 2004.

[34] Y. Li, H. Wu, D. Perkins, N.-F. Tzeng, and M. Bayoumi, "MAC-SCC: Medium Access Control with a Separate Control Channel for Multihop Wireless Networks," in *IEEE ICDCSW 2003*, 2003.

[35] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks," in *The 12th Workshop on Parallel and Distributed Simulations*, May 1998.

[36] G. Zhou, T. He, J. A. Stankovic, and T. F. Abdelzaher, "Radio Interference Detection in Wireless Sensor Networks," in *IEEE INFOCOM 2005*, March 2005.

[37] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," in *ACM SenSys 2003*, November 2003.

[38] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Impact of Radio Irregularity on Wireless Sensor Networks," in *ACM MobiSys 2004*, June 2004, pp. 125–138.

[39] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," in *ACM SenSys 2003*, November 2003.

[40] A. Cerpa, J. L. Wong, L. Kuang, M. Potkonjak, and D. Estrin, "Statistical Model of Lossy Links in Wireless Sensor Networks," in *ACM/IEEE IPSN'05*, April 2005.

[41] J. Chang and N. F. Maxemchuk, "Reliable Broadcast Protocols," in *ACM Transactions on Computer Systems (TOCS)*, 1984.

[42] J. Tourrilhes, "Robust Broadcast: Improving the Reliability of Broadcast Transmissions on CSMA/CA," in *IEEE PIMRC 1998*.

[43] E. Vollset and P. Ezhilchelvan, "A Survey of Reliable Broadcast Protocols for Mobile Ad-Hoc Networks," in *Technical Report CS-TR-792, University of Newcastle upon Tyne*, 2003.

[44] L. Bao and J. J. Garcia-Luna-aceves, "A New Approach to Channel Access Scheduling for Ad Hoc Networks," in *ACM MobiCom 2001*, July 2001, pp. 210–221.

[45] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The Flooding Time Synchronization Protocol," in *ACM SenSys 2004*, November 2004.

[46] Y. C. Tay, K. Jamieson, and H. Balakrishnan, "Collision-Minimizing CSMA and Its Applications to Wireless Sensor Networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, pp. 1048–1057, 2004.

[47] B. Karp, *Geographic Routing for Wireless Networks*, Ph.D. thesis, Harvard University, Cambridge, MA, 2000.

[48] A. Mishra, E. Rozner, S. Banerjee, and W. Arbaugh, "Exploiting Partially Overlapping Channels in Wireless Networks: Turning a Peril into an Advantage," in *ACM/USENIX IMC 2005*, October 2005.