# Continuous Asynchronous Discovery with Efficient Synchronous Communication for Mobile Networks

Hui Cao
The Ohio State University
caohu@cse.ohio-state.edu

Anish Arora
The Ohio State University
anish@cse.ohio-state.edu

Kenneth W. Parker
The *Samraksh* Company
kenneth.parker@skynetaccess.net

Ten H. Lai
The Ohio State University
lai@cse.ohio-state.edu

***Abstract*—A low duty cycle wireless sensor network (WSN) implies that node radios must be off most of the time. Many researchers have observed that synchronous MACs achieve significantly higher energy efficiencies than the theoretical limits for asynchronous MACs. Yet the common practice is to use asynchronous MACs. This is in large part due to the bootstrapping problem created by synchronous communication: discovery is inherently asynchronous. Systems that employ a lower efficiency asynchronous discovery phase followed by a higher efficiency synchronous communication phase are often brittle, especially for mobile networks where nodes need to be added and removed continuously.**

**The solution presented here is to create virtual MAC services that share the same link. Specifically, we design an energy efficient protocol that provides robust asynchronous discovery and synchronous (unicast and broadcast) communication in parallel. Discovery, unicasts, and broadcasts each have different wakeup time intervals that either do not overlap with the others or that overlap with low probability; this is realized via pseudo-random slot selection.**

**To minimize the energy consumption of asynchronous discovery, we design a wakeup schedule that achieves the optimal bound for $3$ state (listen, beacon, and sleep) radios. To minimize the energy consumption of synchronous communication, we adapt the duty cycle to correspond to actual communication traffic levels. We have implemented our protocol for the TelosB mote platform. Our experimental results show asynchronous discovery is achieved within a discovery frame, and that our duty cycle adaptation works well in the presence of varying traffic in different environments.**

## I. Introduction

To achieve a low duty cycle for battery powered wireless sensor networks, node radios must be scheduled to switch off most of the time. Current practice in wireless sensor networks has exploited several wakeup-sleep schedules, which are broadly classified as synchronous or asynchronous. Synchronous approaches have been shown to achieve higher energy efficiency than asynchronous ones, analytically [1], [2] and by simulation [3].

### *A. Applying Synchronous Protocols in a Mobile Network*

Bootstrapping a network requires initial discovery of nodes and initial node/time synchronization, and so cannot rely on a synchronous protocol. Bootstrapping has typically been handled, if at all, by running a less energy efficient asynchronous MAC in order to setup the network and then switching to a high efficiency protocol for routine operation. This two phase approach is unsuitable for mobile networks, which need to continuously add and subtract nodes and also to partition and recombine.

For an always-on sensor network, neighborhood discovery is not an issue, since packet transmission can be overheard by neighboring nodes. However, in a low duty cycle mobile network, two neighboring nodes may never communicate because their wakeup schedules never overlap. Even worse, for any wakeup schedule that is based on time synchronization, unsynchronized nodes may be lost forever because synchronized nodes and unsynchronized nodes are mutually unaware of each other.

In addition, link quality dynamics and clock variation also require continuous neighbor discovery. In [4], multiple schedules (time zones) have been shown to exist consistently on a $50$ Mica2Dot motes network running S-MAC. Our simulations also confirm that network partitions can break down time synchronization in low duty cycled network. We conclude that a synchronous protocol must coexist with energy efficient neighbor discovery in mobile, dynamic low duty cycle networks.

Both [5] and [6] have developed schedules that have the property that, for all possible time shifts for a schedule, there is a time when the active slots of nodes overlap. However, the schedules are derived for $802.11$ radios, where a node can both beacon and listen in one slot during its wakeup. For typical radios used in WSNs, such as $802.15.4$, a node can not both beacon and listen in the same slot. For the three-state (beacon, listen, and sleep) WSN radios, as far as we know, no theoretical results about optimal bounds and optimal schedules have been presented.

Other than bootstrap, duty cycle adaptation is a key consideration for mobile network MACs. Because the point of designing a low duty cycle system is typically for energy efficiency, adapting the duty cycle according to the network traffic is key. Analytical results shown in [1] indicate that different traffics require different duty cycles to achieve optimal energy efficiency. In other words, nodes must adapt their duty cycle according to traffic changes. If the duty cycle is lower than required, higher collision or sender buffer overflow can happen; if the duty cycle is higher than required, energy is wasted on idle listening.

Extant work in duty cycle adaptation [7] first estimates incoming traffic and then calculates an optimal duty cycle. This approach has two issues: One, estimation of dynamic traffic can be error prone and yield an incorrect duty cycle. And two, collisions are ignored in measuring the incoming traffic. However, collisions resulting from a lower-than-desired duty cycle are an important indicator. Our duty cycle adaptation mechanism tries to avoid those shortcomings.

### B. *Summary of the Results*

In this paper, we design an energy efficient protocol for mobile WSNs that provides robust asynchronous and synchronous communication. Specifically, it exports three services in parallel: asynchronous discovery, synchronous unicast, and simultaneous broadcast. Each service has a different wakeup time interval that either yields no overlap with the others or or overlaps with low probability; this is achieved via pseudo-random slot selection per service.

We formulate the problem of optimal neighbor discovery for a duty cycled network, and provide a class of optimal wakeup schedules that achieves neighbor discovery with minimum energy. Our $3$-state schedules consume $1/\sqrt{2}$ of energy required by other approaches to achieve neighbor discovery in a discovery frame.

Also, to maintain an optimal duty cycle in the presence dynamic traffic, we provide a duty cycle adaptation mechanism that is based on feedback from channel utilization and collisions. We identify a metric, the **Activity Ratio**, using which we transform the duty cycle optimization problem into a fixed point control problem.

Experimental and simulation results show that our protocol provides robustness via asynchronous discovery, as well as higher energy efficiency by using synchronous communication combined with duty cycle adaptation.

### C. *Related Work*

In [8], radio communication is revealed as the dominant power consumer among all components. There are a large number of protocols to schedule radio wakeup:

*1) Synchronous Protocols:* The simplest synchronous protocol wakes up all the nodes at the same time. S-MAC [9] and T-MAC [10] are notable variants of this approach. At a time, only one receiver can receive a unicast message while the other receivers idle listen.

The concept of receiver-centric power management protocols was first introduced in [1]. In receiver-centric protocols, receivers are scheduled to wake up in different slots. Ideally, only one receiver wakes up at a slot and potential sender only need to wakeup to transmit in that slot; idle listening is avoided. The key feature of the protocol is that it avoids the vast majority of collisions by staggering or scheduling receiver on times rather than staggering or scheduling transmission times. Although, a global schedule can obtain optimal energy efficiency, OMAC [1] eschews the difficulty of implementing and maintaining a global schedule and achieves near optimal energy efficiency by exploiting a pseudo-random wakeup schedule. In [3], another receiver-centric MAC protocol called Crankshaft is presented, wherein node ID is used to decide wakeup schedule. The duty cycle in Crankshaft MAC is fixed and not easily changed.

*2) Asynchronous Protocols:* Asynchronous approaches have been extensively studied and adopted in MAC layer protocols, in part because they assume less about node coordination and time synchronization and this reduces system complexity. A well-known approach uses Low Power Listening (LPL) whereby nodes wake up periodically and independently to check channel activity, and when a node wishes to send a message it sends out a long preamble first to wake up the receiver [11], [7].

Theoretical results on efficient, deterministic wakeup-sleep schedules for $802.11$ networks are presented in [5] and [6]. To minimize energy consumption, all nodes wake up according to a schedule with only $\sqrt{N}$ wakeup slots out of total $N$ slots. The schedule guarantees that, for any two nodes, there is a slot during which both nodes are awake, no matter what time shift exists between the two schedules. In [5], a dynamic scheduler is developed that uses only $\sqrt{2N}$ wakeup slots to get different duty cycles by changing parameters, based on a torus quorum system.

## II. Energy Efficient Protocol Design for Mobile Networks

### A. *Major Components in Protocol*

Our protocol has four major components in our protocol, as shown in Figure 1.

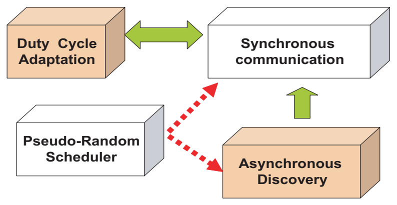

Fig. 1. The major components in the protocol

*1) Synchronous Communication:* Once nodes have discovered each other (including their ID), they share sufficient information to send and receive synchronous communications with respect to neighbors. Recall that our protocol supports synchronous unicasts and simultaneous broadcast.

Specifically, the unicast protocol performs the following tasks at each node: it maintains at each node a pseudo-random unicast-receiver schedules for every neighbor; it buffers messages; it schedules transmission to correspond to the time when the intended neighboring recipient's receiver is on; it explicitly ACKs each message, if the ACK mode is enabled; and it deals with collisions by employing a low complexity back-off scheme. (We note that unicast protocol allows a synchronous broadcast to be simulated by sending one unicast to each neighbor. Such a broadcast is invoked by identifying the receiver ID as $U\_BCAST$.)

The simultaneous broadcast protocol is used when a broadcast must serve as a synchronization barrier. (If a broadcast is performed merely to get data to all of a nodes neighbors, we advocate simulating the broadcast with a series of unicasts as described above.) In the simultaneous broadcast case, instead of requiring the transmitter to accommodate the receiver's schedule, the receiver is required to accommodate the transmitters schedule. That is, each node maintain a pseudo-random broadcast Schedule for each of its neighbors; during its broadcast slot, all of its neighbors are required to listen. Such a broadcast is invoked by identifying the receiver ID as $S\_BCAST$.

*2) Asynchronous Discovery:* Asynchronous discovery acts as the base line of communication. It uses a wakeup schedule that we describe in Section III. The discovery beacons enable each node to know all of its neighboring nodes within a single discovery frame length with high probability, and thus make the network robust against partitioning that may be induced by mobility and link dynamics.

The asynchronous discovery protocol also provides an interface for asynchronous broadcast communication for services such as time synchronization which must work even when the network is not synchronized. In this broadcast is invoked by identifying the receiver ID as $A\_BCAST$, the content of the broadcast is implemented as an overlay atop the discovery beacon messages of the protocol.

*3) Duty Cycle Adaptation:* This component deals with cases where the traffic is not well characterized at compile time. It uses a mechanism based on feedback control, described in Section IV, to adjust the duty cycle at each node according to traffic changes experienced locally.

*4) Pseudo-Random Scheduler:* The purpose of the pseudo-random selection of slots is to avoid systematic conflicts between (a) the schedules of neighbors for each of the communication services and (b) the schedules of the communication services of each node.

Specifically, a pseudo-random number generator is used in synchronous communication to select the slot during which a receiver node will listen within each frame. It is necessary for any neighboring node that wishes to send a message to the node to be able to figure out this slot. This requires that the sending node be able to reproduce the same pseudo-random slot selection that was used by the receiver node. This can be achieved by transferring 1) the last slot assignment, 2) the current frame length, and 3) the seed of the receiver to the sender via the discovery beacons.

A pseudo-random number generator is also used in asynchronous discovery for randomizing the discovery wakeup schedule. If all nodes wakeup at exactly the same time their wakeup and beacon slots will be identical and no discovery will occur. Randomizing the schedule avoids this "zero-shift" issue that is inherent to any deterministic schedule. Details will be discussed in Section III-G.

Figure 2 shows the communications between neighbors.

## B. Key Protocol Parameters

*1) Synchronous Unicast Frame Length:* The duty cycle of the synchronous communication is determined by the communication rate of the system. If the frame length is $N$, the duty cycle is $1/N$. Typically, $N$ is $\in [50, 500]$ for a low duty cycle application.

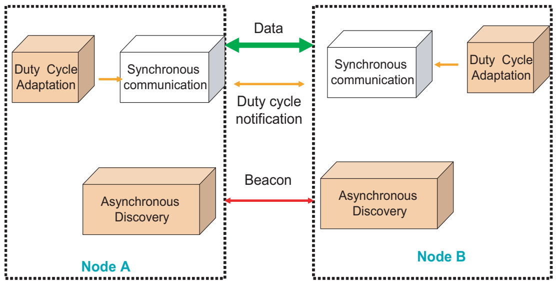

Fig. 2. The communications between neighbors

*2) Broadcast Frame Length:* A key difference between unicast frame length and broadcast frame length is that, because broadcast events are high energy events compared to unicasts (i.e., all of the nodes neighbors must be awake for the event), the broadcast frame length will typically be significantly longer than the frame length used in synchronous unicasts, e.g. $500$ to $10,000$ slots.

*3) Discovery Frame length:* The duty cycle of the discovery protocol is determined by the latency requirements for discovery. Typically these requirements can be fixed at compile time, but if they were to change in situ or to be different from region to region, it would be possible to change the frame length dynamically. The appropriate discovery frame length is much larger than normal data frame length. Typically, it is above $10,000$ slots for a low duty cycle application.

*4) Duty Cycle Adaptation Parameters:* The details of these parameters are described in Section IV.

## C. Relationship with Other Components

The relationship between our MAC protocol and other network protocols is shown in Figure 3.

A key decision is whether to subsume time synchronization within the MAC. On the one hand, time synchronization requires neighbors to exchange time information without assuming synchronous communication, and on the other hand, synchronous MAC protocols needs time synchronization. This chicken-and-egg dilemma suggests that subsuming time synchronization is simple; however, this is undesirable as it greatly limits the portability of the MAC across network platforms. We avoid the subsumption simply by letting time synchronization messages be exchanged via asynchronous broadcasts.

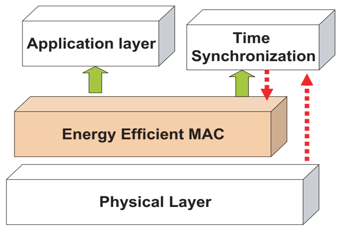

Fig. 3. The Relationship between MAC and other componets

## III. Continuous Discovery for Low Duty Cycle Mobile Networks

### A. System Model

The network consists of a number of mobile nodes. The operation of each radio can be viewed as a sequence of frames, each consisting of a constant number $N$ of constant time length slots. We assume initially the slot boundaries across different nodes are aligned, but then relax this assumption later. In each slot, a radio can be in one of three states: beacon, listen, or sleep. (This model represents the constraints of WSN radios, such as the $CC2420$ radio which conforms to the $802.15.4$ standard.) Each node $u$ follows a schedule $S_u$ that dictates its state per slot. $S_u$ is represented as:

$$S_u(j) = \begin{cases} 0 & \text{if node } u \text{ sleeps in slot } j \\ 1 & \text{if node } u \text{ beacons in slot } j \\ 2 & \text{if node } u \text{ listens in slot } j \end{cases}$$

Let $u$ and $v$ range over the network nodes. We denote the number of sleeping, beaconing, and listening slots per frame as $n_u^s, n_u^b, n_u^l$ and $n_v^s, n_v^b, n_v^l$ respectively.

### B. Problem Statement

Our goal is that each node wakes up as little as possible, while being able to discover new nodes as quickly as possible. Before defining the problem formally, we distinguish two types of neighbor discovery: unidirectional discovery and mutual discovery.

**Definition 1: (unidirectional discovery)** We say $u$ and $v$ achieve unidirectional discovery iff for any integer shift $T \in [0, N-1]$, $\exists i, j$ such that $(S_u(i+T) = 1 \wedge S_v(i) = 2) \vee (S_u(j+T) = 2 \wedge S_v(j) = 1)$.

Unidirectional discovery implies that at least one node can discover the other node, but there is no guarantee that both nodes can find each other. However, mutual discovery guarantees discovery in both directions.

**Definition 2: (mutual discovery)** We say $u$ and $v$ achieve mutual discovery iff for any integer shift $T \in [0, N-1]$, $\exists i, j$ such that $(S_u(i+T) = 1 \wedge S_v(i) = 2) \wedge ((S_u(j+T) = 2) \wedge S_v(j) = 1)$.

We now define the optimal SBL (Sleep-Beacon-Listen) problem for unidirectional discovery and mutual discovery.

**Definition 3: (Optimal SBL for unidirectional discovery)** Given a fixed $N$, design $S_u$ and $S_v$ so as to achieve (1) $\min\{n_u^b + n_u^l + n_v^b + n_v^l\}$ and (2) unidirectional discovery.

The SBL problem is thus to minimize the number of active slots (beacon slots and listen slots) while guaranteeing unidirectional discovery. The SBL problem for mutual discovery is defined likewise as:

**Definition 4: (Optimal SBL for mutual discovery)** Given a fixed $N$, design $S_u$ and $S_v$ so as to achieve (1) $\min\{n_u^b + n_u^l + n_v^b + n_v^l\}$ and (2) mutual discovery.

### C. Optimal Bound for Deterministic Schedule

In this section, we characterize the minimum number of active slots required to achieve unidirectional discovery and mutual discovery.

**Theorem 1:** To achieve unidirectional discovery, the following condition must hold:

$$n_u^b \cdot n_v^l + n_u^l \cdot n_v^b \geq N$$

*Proof:* Since both $u$ and $v$ may be out of sync with any shift, without loss of generality, we fix the schedule of node $u$ and only shift the schedule of $v$. In other words, we let $S_v(i+T)$ be $S_u(i)$, where $T$ is the shift and $0 \leq i < N$. For any beacon slot $j \in [0, N-1]$ in $u$, the total number of listening slots that $j$ overlaps during the shift is $n_v^l$. For all beacon slots in $u$, the total number of beacon-listen overlapping pairs is $n_u^b \cdot n_v^l$. Similarly, the total number of listen-beacon overlapping pairs is $n_u^l \cdot n_v^b$. None of these pairs repeat during the shift $T \in [0, N-1]$. Then the total number of "discovery" slots, $N_d$, is $n_u^b \cdot n_v^l + n_u^l \cdot n_v^b$. On the other hand, by unidirectional discovery, at least one beacon-listen or listen-beacon pair is guaranteed for every shift. So the total number of "discovery" slots, $N_d$, is at least $N$. We then have: $n_u^b \cdot n_v^l + n_u^l \cdot n_v^b = N_d \geq N$ ∎

**Theorem 2:** To achieve mutual discovery, the following condition must hold:

$$n_u^b \cdot n_v^l + n_u^l \cdot n_v^b \geq 2N$$

*Proof:* The proof is similar to Theorem 1, the only difference being that mutual discovery achieves at least one beacon-listen and one listen-beacon pair per every shift. So the total number of "discovery" slots, $N_d$, is at least $2N$. Thus we have: $n_u^b \cdot n_v^l + n_u^l \cdot n_v^b = N_d \geq 2N$ ∎

For ease of deploying WSNs, it is convenient to use the same wakeup schedule for all nodes. We call such schedule design symmetric. We get the following corollary by using Theorem 1 and Theorem 2.

**Corollary 1: (Bound for symmetric solution of the SBL problem)** Given $N$, $n_u^b = n_v^b = n_b$, and $n_u^l = n_v^l = n_l$, any SBL solution must satisfy:

$$2n_b \cdot n_l \geq N \text{(for unidirectional discovery)}$$
$$n_b \cdot n_l \geq N \text{(for mutual discovery)}$$

For an optimal symmetric design, the minimum number of total active slots are determined by the following corollary 2.

**Corollary 2: (Bound for optimal symmetric solutions of SBL problem)** Given $N$, $n_u^b = n_v^b = n_b$, and $n_u^l = n_v^l = n_l$, the optimal SBL solution must satisfy[1]:

$$\min\{n_u^b + n_u^l + n_v^b + n_v^l\} \geq$$
$$\begin{cases} 2\sqrt{2N} & \text{(for unidirectional discovery)} \\ 4\sqrt{N} & \text{(for mutual discovery)} \end{cases}$$

To achieve this bound, $n_b$ and $n_l$ must satisfy:

$$\begin{cases} n_b = n_l = \sqrt{2N}/2 & \text{(for unidirectional discovery)} \\ n_b = n_l = \sqrt{N} & \text{(for mutual discovery)} \end{cases}$$

*Proof:* $\min\{n_u^b + n_u^l + n_v^b + n_v^l\} = 2 \cdot \min\{n_b + n_l\} \geq 4\sqrt{n_b \cdot n_l}$. By Corollary 1, $\sqrt{n_b \cdot n_l} \geq \sqrt{2N}/2$ holds in the

[1]Note: For ease of exposition of our theorems and proofs, we round $\sqrt{N}$ to the nearest larger integer when it is not an integer.

unidirectional discovery case, and $\sqrt{n_b \cdot n_l} \geq \sqrt{N}$ holds in the mutual discovery case. So, $\min\{n_u^b + n_u^l + n_v^b + n_v^l\} \geq 4\sqrt{n_b \cdot n_l} \geq$

$$\begin{cases} 2\sqrt{2N} & \text{(for unidirectional discovery)} \\ 4\sqrt{N} & \text{(for mutual discovery)} \end{cases}$$

To make the first "≥" equal, $n_b$ and $n_l$ must satisfy $n_b = n_l$, so we can get the value of $n_b$ and $n_l$ in Corollary 2. ■

### D. Optimal Deterministic Schedule

In this section, we describe schedules that achieve the optimal bound. First, we introduce the concept of a block. We divide a frame into blocks, each with $X$ slots such that there is at least one active (beacon or listen) slot per block. Let $N = X \cdot Y$ slots.

*1) Optimal Unidirectional Discovery Schedule:* Although unidirectional discovery only guarantees discovery in one direction, it consumes less energy than mutual discovery. In addition, if one node discovers the other one, it can notify the other one about its schedule by using a synchronous ACK.

**Theorem 3:** The schedule based on the following rules achieves unidirectional discovery:

1) $S(i \cdot X) = 1$ for all integer $i \in [0, Y-1)$.
2) $S(X \cdot (Y-1) + j) = 2$ for all integer $j \in [0, X/2]$. (Note: by this rule, $S(X \cdot (Y-1)) = 2$)
3) $S(X \cdot (Y-1) + (X/2+1)) = 1$.

One instance of these schedules is shown in Figure 4. Generally speaking, this schedule requires node beaconing

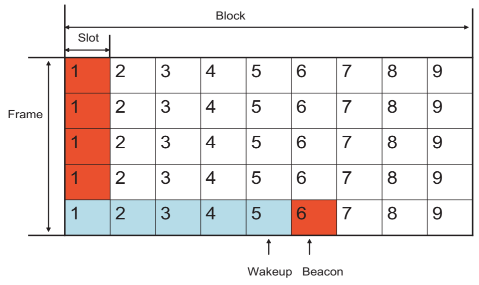

Fig. 4. Optimal wakeup schedule for unidirectional discovery

over the slots in one column (except for one), and listening in more than half of the slots in the excepted row. The proof validates that unidirectional discovery holds for every possible shift.

*Proof:* For any two nodes $u$ and $v$, without loss of generality, let the time $v$ be shifted wrt $u$ by $\Delta$, where $\Delta$ is an integer expressed as: $\Delta = a \cdot (X \cdot Y) + b \cdot (X) + c$ and $(0 \leq b < Y, 0 \leq c < X)$. So their schedules are:

$$S_u(j) = S(j)$$
$$S_v(j) = S(j+\Delta) = S(j + b \cdot X + c)$$

**Case 1** ($c = 0$)
$S_u(j) = S(j)$ and $S_v(j) = S(j + b \cdot X)$. Apparently, if $b = 0$, then $S_u(j) = S(j) = S_v(j)$, both nodes use the same wakeup schedule and are in sync. In the following, we assume $b \neq 0$. When $j = (Y-1-b)X$, $S_u(j) = S((Y-1-b)X), S_v(j) = S((Y-1)X)$. According to rule 1, and $(Y-1-b) \in [0, Y-1)$, so $S_u(j) = S((Y-1-b)X) = 1$. According to rule 2, $S_v(j) = S((Y-1)X) = 2$. So, node $v$ can listen the beacon of node $u$ at slot $j = (Y-1-b)X$.
**Case 2** ($c \neq 0, b = 0$)
$S_u(j) = S(j)$ and $S_v(j) = S(j+c)$.
If $c \leq X/2+1$, let $j = X \cdot (Y-1) + (X/2+1-c)$. Since $0 \leq (X/2+1-c) \leq X/2$, $S_u(j) = S(X \cdot (Y-1) + (X/2+1-c)) = 2$, based on rule 2. In addition, $S_v(j) = S(j+c) = S(X \cdot (Y-1) + (X/2+1)) = 1$, according to rule 3. So, $u$ can listen to the beacon of $v$ in slot $j = X \cdot (Y-1) + (X/2+1-c)$.
If $c > X/2+1$, let $j = X \cdot (Y-1) + (X-c)$. Since $0 \leq (X-c) \leq X/2$, $S_u(j) = S(X \cdot (Y-1) + (X-c)) = 2$, based on rule 2. In addition, $S_v(j) = S(j+c) = S(X \cdot (Y-1) + X) = S(0) = 1$, according to rule 1. So, $u$ can listen to the beacon of $v$ in slot $j = X \cdot (Y-1) + (X-c)$.
**Case 3** ($c \neq 0, b \neq 0$)
$S_u(j) = S(j)$ and $S_v(j) = S(j + b \cdot X + c)$.
If $c \leq X/2+1$, let $j = X \cdot (Y-1) - b \cdot X$. $S_u(j) = S(X \cdot (Y-1-b)) = 1$, based on rule 1. In addition, since $c \leq X/2+1$, $S_v(j) = S(j + b \cdot X + c) = S(X \cdot (Y-1) + c) = 2$, according to rule 2. So, $v$ can listen to the beacon of $u$ in slot $j = X \cdot (Y-1) - b \cdot X$.
If $c > X/2+1$, let $j = X \cdot (Y-1) + (X-c)$. Since $0 \leq (X-c) \leq X/2$, $S_u(j) = S(X \cdot (Y-1) + (X-c)) = 2$, based on rule 2. In addition, If $b \neq Y-1$, $S_v(j) = S(j + b \cdot X + c) = S(X \cdot (Y-1) + X + b \cdot X) = S(b \cdot X) = 1$, according to rule 1. So, $u$ can listen to the beacon of $v$ at slot $j = X \cdot (Y-1) + (X-c)$. If $b = Y-1$, let $j = X \cdot (Y-1) + (X/2+1)$. So, $S_u(j) = 1$, based on rule 3. In addition, $S_v(j) = S(j + b \cdot X + c) = S(X \cdot (Y-1) + (Y-1) \cdot X + 1 + X/2 + c) = S((Y-1)X + (c+1-X/2)) = 2$, according to rule 2. So, $v$ can listen to the beacon of $u$ at slot $j = X \cdot (Y-1) + (X/2+1)$. ■

Based on the schedule described in Theorem 3, we achieve the optimal bound, as is shown in the following corollary.

**Corollary 3:** If $X = 2 \cdot Y$, the schedule in Theorem 3 is optimal.

*Proof:* When $X = 2 \cdot Y$, $X = \sqrt{2N}$, so the total number of active slots is $2 \cdot X = 2\sqrt{2N}$, which is the optimal bound for unidirectional discovery. ■

*2) Optimal Mutual Discovery Schedule:* Mutual discovery is required for many applications, especially in mobile networks. In addition, it also has inherent robustness against unaligned slot boundaries, as shown in Theorem 6. The following theorem defines an optimal mutual discovery schedule.

**Theorem 4:** The schedule based on the following rules achieves mutual discovery:

1) $X = Y$
2) $S(i \cdot X) = 2$ for all integers $i \in [0, Y-3]$.
3) $S(X \cdot (Y-1) + j) = 1$ for all integers $j \in [1, X-1]$.
4) $S((Y-2) \cdot X) = 1$
5) $S(X \cdot (Y-2) + 1) = 2$
6) $S(X \cdot (Y-4) + 1) = 2$.

This schedule requires node beaconing in the slots of one column and listening in the slots of one row, with only a few exceptions as mentioned in the schedule. Figure 5 shows one instance of an optimal schedule for mutual discovery.

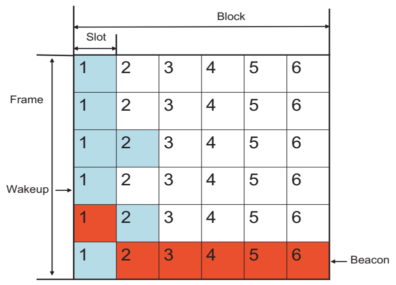

Fig. 5. Optimal wakeup schedule for mutual discovery

*Proof:* The proof is similar to that Theorem 3, it involves checking that mutual discovery holds under every possible shift. ■

## E. Optimal Schedule without Slot Alignment

In this section, we relax the assumption that the slots of different nodes are aligned. A successful communication happens when a receiver hears the full preamble and detects the start of frame delimiter (SFD), as shown in figure 6. After SFD is detected, the receiver can stay active during the slot to receive the data packet. Since the preamble length (denoted as $T_p$) is small compared with the slot length, we let $T_p \approx 0$. In this case, we say discovery occurs when a receiver wakes up for one slot and a preamble is received during that slot.

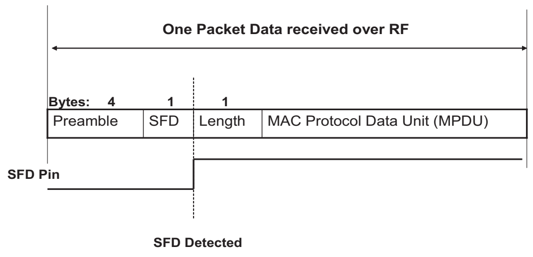
One Packet Data received over RF
Bytes: 4 1 1
Preamble
SFD
Length
MAC Protocol Data Unit (MPDU)
SFD Pin
SFD Detected

Fig. 6. Receiving a packet in CC2420

### 1) Unidirectional Discovery Schedule:

**Theorem 5:** The schedule defined by Theorem 3 achieves unidirectional discovery even when the slots of different nodes are unaligned.

*Proof:* The proof is similar to Theorem 3, except we validate that discovery occurs for all possible real number shifts. We omit the details here. ■

We note that not all schedules that achieve unidirectional discovery schedule in aligned slots case suffice unaligned slots case. A counter example is shown in Figure 7. This schedule is very similar to the schedule in Theorem 3, except that its beacon and listen slots are switched. It is easily checked that this schedule achieves unidirectional discovery in the aligned slots case but not in unassigned slots case.

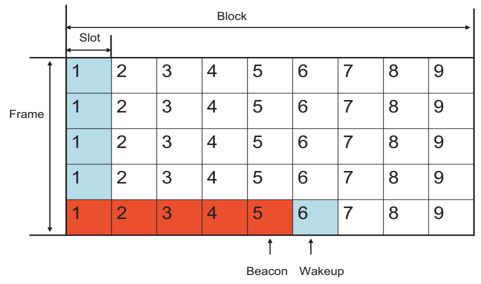

Fig. 7. The wakeup schedule for unidirectional discovery

In contrast, mutual discovery schedules have a robust feature that guarantees discovery even when slots are unaligned.

### 2) Mutual Discovery Schedule:

**Theorem 6:** Any algorithm that achieves mutual discovery when the slots of different nodes are aligned also achieves mutual discovery when those slots are unaligned.

*Proof:* For any two nodes $u$ and $v$, without loss of generality, let the time of $v$ be shifted w.r.t $u$ by $\Delta$, where $\Delta$ is a real number, which can be expressed as:

$$\Delta = a \cdot (X * Y) + b * (X) + c + \delta$$
$$0 \leq b \leq Y-1,\ 0 \leq c \leq X-1,\ 0 < \delta < 1$$

When $\delta = 0$, the slots are aligned. According to mutual discovery, there exists a slot $i$ in $u$ that hears a beacon from $v$. When $\delta = 1$, there exists a slot $j$ in $u$ that hears a beacon from $v$. In other words, $S_u(j) = 2, S_v(j) = 1$. When $\delta \in (0,1)$, because $u$ continues listening at slot $j$, $S_u(j+\delta) = 2$; also $v$ beacons at the shifted time $j+\delta$, i.e., $S_v(j+\delta) = 1$. So, $u$ hears a beacon at time $j+\delta$. Similarly, a $v$ also always hears a beacon from $u$, which means mutual discovery is guaranteed even when slots are not aligned. ■

This theorem implies the following corollary:

**Corollary 4:** The schedule in Theorem 4 achieves optimal mutual discovery even when the slots of different nodes are unaligned.

## F. Comparison with Previous Work

The schedule defined in Theorem 4 is inspired by the schedules in [5] and [6]. But instead of having two states, active and sleep, it has three states, beacon, listen, and sleep; and it is designed so that for all shifts except for the zero shift, there are exactly two overlapping activities: one where the first nodes listens and the second node beacons and the other where the first nodes beacons and the second node listens. To use the optimal two states schedule proposed by [5] and [6], a node has to randomly select to beacon or listen during wakeup. To achieve mutual discovery, a 2 states schedule requires on average 8 discovery frames. This is because:

- In every frame, one node can discover the other one with probability 0.5.
- To have mutual discovery, the probability for every two discoveries is 0.5.

For a frame length $N$, to guarantee mutual discovery in 8 frames, our scheme requires $2\sqrt{8N} = 4\sqrt{2N}$, while optimal

2-states schedule needs $8\sqrt{N}$ on average. So our optimal 3-states schedule uses only $1/\sqrt{2}$ of energy that the 2-states schedule uses. In addition, our 3-states schedule guarantees mutual discovery, while the 2-states schedule only achieves this only with probability 0.5.

### G. Implementing Continuous Neighbor Discovery

The asynchronous discovery protocol is described as follows:

- Every node uses the schedule defined in Theorem 4, shifted by an amount that is decided by the pseudo-random generator, whose seed is (say) the hash of the node ID. One example is shown in Figure 8.
- When two nodes discover each other, they exchange time synchronization information and the keys used to determine the schedule used for synchronous communication.

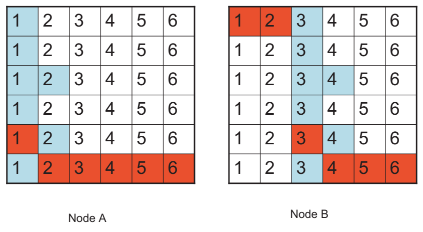

Fig. 8. Two Different Shifts of the Schedule.

The odds of any pair of nodes using the same shift are $1/N$, and $N$ will typically be from $10^3$ to $10^5$. Even though the odds of non-discovery between nodes is small, such problems will tend to persist. This can be dealt with by adding a slow random walk to the schedule shift. On each frame, we randomly increase the shift or decrease the shift by one slot. Shifts of only one slot have only a $1/N$ change of causing a new problem and always are sufficient to fix any existing problem.

Note also that the discovery mechanism implicitly provides a heartbeat or link monitoring capability even for nodes already in the mobile network.

## IV. Duty Cycle Adaptation

For any particular traffic level, there is an optimal duty cycle that maximizes the energy efficiency. In this section, we focus on adapting duty cycle to that optimal point.

### A. Receiver Based Collision Detection

It is important to detect collisions at the receiver to estimate the incoming traffic. CC2420 provides CCA (Clear Channel Arbitration) status, by which we can detect collisions. Traditionally, CCA is used for implementing CSMA at the sender. However, it also allows the receiver to be aware of unreadable preambles caused by overlapping transmissions. Consistent collisions indicate an under-provisioned communication system, while rare collisions indicate over-provisioning. The rate of receiver collision provides sufficient information for receiver to adjust its duty cycle.

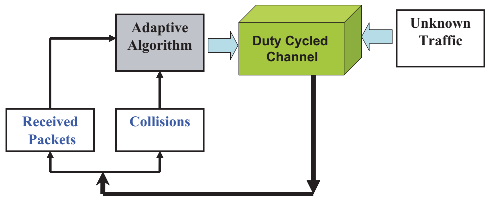

Fig. 9. The structure of adaptive duty cycle control system.

### B. Activity Ratio as the Control Metric

Short term variations in traffic may be absorbed by the buffering mechanism implemented in the protocol. But slightly longer term traffic variations require adaptation of the duty cycle. We have opted to use the control structure shown in Figure 9. There are two interesting points to note: 1) there is no attempt to directly send state information from the sender, and 2) control is based on knowledge of the number of collisions and the number of received packets. In addition the control algorithm attempts to deal with interfering traffic from an unknown origin.

Let $n_i$ be the total number of idle listening slots, $n_c$ be the total number of collision slots, and $n_r$ be the total number of successfully received slots. Then the **activity ratio**, defined as

$$r_a = \frac{n_r + n_c}{n_r + n_c + n_i},$$

captures the performance of the link over a wide range of environments. In particular, the optimal duty cycle depends strongly on the number of neighbors that send data do it and the amount of the other traffic in its vicinity. In practice, these metrics are hard to estimate and prone to large error. However, the activity ratio corresponding to the efficiency only, and only weakly depends on these hard to estimate aspects of the environment.

As a partial justification of this assertion, consider a uniform traffic pattern.

**Theorem 7:** In a network with uniform traffic, the optimal receiver energy efficiency is achieved when the activity ratio is in $[0.64, 0.75]$, no matter how many neighbors are transmitting.

*Proof:* Let $\eta$ define the number of nodes sending to the receiver of interest. We denote the sender's duty cycle by $d_s$ and the receiver's duty cycle by $d_r$. To provide sufficient bandwidth for all senders, $d_r \geq \eta \cdot d_s$. The probability $p$ that a sender transmits when the receiver is on is $p = d_s/d_r$.

The receiver energy efficiency is defined as the ratio of energy spent on successful transmission to the total energy spent by the receiver. In this situation, the energy efficiency is:

$$E = \eta \cdot p(1-p)^{\eta-1}$$

It follows that the maximal energy efficiency occurs when,

$$p_0 = 1/\eta \qquad \Rightarrow \qquad d_r = \eta * d_s$$

As already mentioned, both $\eta$ and $d_s$ are unknown to the receiver and difficult to estimate accurately. However, the activity ratio in this case is:

$$r_a = 1 - \frac{n_i}{n_r + n_c + n_i} = 1 - P_{idle} = 1 - (1-p)^\eta.$$

The receiver energy efficiency is maximized when

$$r_{a\_opt} = 1 - (1-p_0)^\eta = 1 - (1 - \frac{1}{\eta})^\eta$$

The value of $r_{a\_opt}$ is shown in Figure 10. Note that the value depends only weakly on $\eta$ and is in the range $[0.64, 0.75]$. ∎

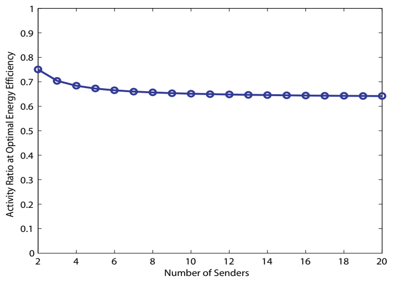

Fig. 10. Activity ratio at optimal duty cycle for different number of senders

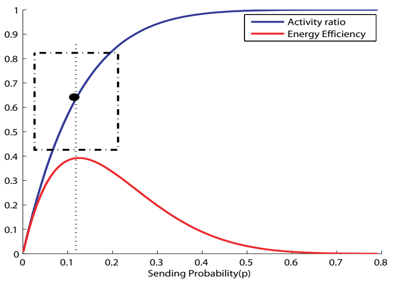

Fig. 11. Activity ratio and energy efficiency as a function of traffic level.

Figure 11 shows performance as a function of receiver duty cycle for a network with 8-nodes sending to a single receiver. Note that the activity ratio is monotonic with significant slope around the optimal point. This implies that this parameter can be usefully controlled in the vicinity of the optimum. The box around the figure shows the operating region resulting from controlling $r_a \in [0.64, 0.75]$.

By using the activity ratio, a duty cycle optimization problem has been transformed into a fixed point control problem. Defining a feedback mechanism that controls the value of $r_a$ to keep it in the range $[0.64, 0.75]$ works well.

### C. A Generic Control Algorithm

Figure 12 presents a simple generic control algorithm. Let:

- $d_r$ be the receiver duty cycle,
- $A_{max}$ be the maximum activity ratio,
- $A_{min}$ be the minimum activity ratio,
- $\alpha$ be duty cycle increasing rate, and
- $\beta$ be duty cycle decreasing rate;

```
**Input:** $r_a, d_r(k)$
**Output:** $d_r(k+1)$
**Parameter:** $A_{min}, A_{max}, \alpha, \beta$
**if** $(r_a > A_{max})$
    $d_r(k+1) = d_r(k) + d_r(k) * \alpha;$
**else if** $(r_a < A_{min})$
    $d_r(k+1) = d_r(k) - d_r(k) * \beta;$
**end**
```

Fig. 12. Basic adaptive duty cycle algorithm

This algorithm converges quickly in most cases. It will settle on a good, but sub-optimal, rather than spend energy on incremental improvements.

Our previous derivation assumes uniform traffic from all the neighbors. However, in real deployment, traffic is frequently unbalanced. Ironically, unbalanced sending traffic increases energy efficiency by avoiding collisions. In the extreme case, with only one sender transmits at a time, the optimal energy efficiency is obtained when the activity ratio is $1$. However, as the control point for the activity ratio approaches $1$, the feedback becomes unstable. In order to maintain some duty cycle gain margin, our simulations indicated that the control point should not be above about $0.85$. The worst case occurs when uniform traffic from a large number of neighbors. In the limit as the number of neighbors goes to infinity, the activity ration for uniform traffic is $0.64$. Setting the control point below this value only introduces more idle listening, without any corresponding benefit.

The parameters $\alpha$ and $\beta$ control the speed of convergence. Larger values improve the MAC efficiency by more quickly reaching the optimal dusty cycle, but too large a value will lead to instability. In order to estimate a reasonable maximum value of $\alpha$, we consider an ad-hoc criterion that starting inside the desired operating box, shown in Figure 11, we should not overshoot landing outside the desired operating box. This suggests the following relationships.

$$\alpha < 5 \cdot (r_a - A_{max}) = 5 \cdot (1 - 0.85) = 0.75$$
$$\beta < 1 \cdot (A_{min} - r_a) = 1 \cdot (0.64 - 0) = 0.64$$

In summary, here are the initial values of parameters:

- $A_{max} = 0.85$
- $A_{min} = 0.64$
- $\alpha$: $5 \cdot (r_a - A_{max})$
- $\beta$: $1 \cdot (A_{min} - r_a)$

### D. Stabilization of Feedback Control Algorithm

Although MIMD achieves better convergence and energy efficient, stability is an issue for this method. If $\alpha$ and $\beta$ are not chosen carefully, it is possible that activity ratio $r_a$ may oscillate from below $A_{min}$ to above $A_{max}$ as shown in Figure 13.

To prevent oscillation, we add stabilization into the basic feedback control algorithm as shown in Figure 14. The main idea is that when a transition from $r_a > A_{max}$ to $r_a < A_{min}$

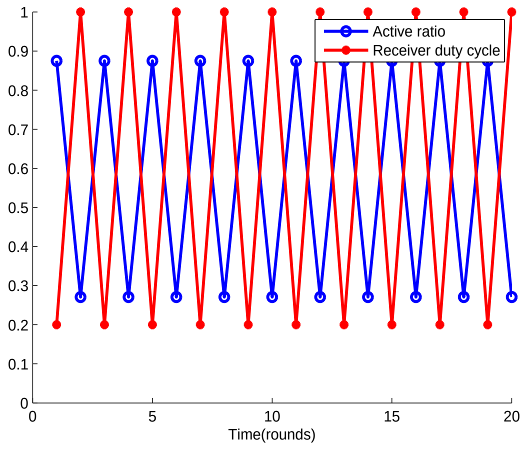

Fig. 13. Oscillation in adaptive duty cycle control system

or from $r_a < A_{min}$ to $r_a > A_{max}$ happens, the rate of receiver duty cycle change is decreased. In this algorithm, variable $lstate$ is used to indicate different states:

- $INC$: duty cycle increasing
- $DEC$: duty cycle decreasing
- $OVER$: transition from $< A_{min}$ to $> A_{max}$ happens
- $BELOW$: transition from $> A_{max}$ to $< A_{min}$ happens
- $NOR$: steady state

Variables $\delta_i, \delta_d$ are used for denoting the step size of increasing or decreasing duty cycle.

This algorithm provides guarantee of stabilization as stated in the following theorem:

**Theorem 8:** When the incoming traffic is steady, the adaptive duty cycle algorithm described in Figure 14 stabilize to a certain activity ratio in $[A_{min}, A_{max}]$, by which optimal efficiency is obtained.

*Proof:* For a network with $\eta$ senders, every node transmits with a certain duty cycle $d_s^i$. Note the duty cycle of different senders may be different, we use a vector $D_s$ to represent:

$$D_s = [d_s^1, d_s^2, \ldots\ldots, d_s^\eta]$$

The relationship between receiver activity ratio $r_a$ and receiver duty cycle $d_r$ can be expressed as function:

$$r_a = f(D_s, d_r)$$

When $D_s$ is fixed, function $f(D_s, d_r)$ is a decreasing function. In other words, when receiver duty cycle increases, activity ratio decreases when incoming traffic is fixed. For instance, in the case of random traffic,

$$r_a = f(D_s, d_r) = 1 - (1 - p_i)^\eta$$

$$where \quad p_i = \max\{1, \frac{d_s^i}{d_r}\}$$

We use Lyapunov theorem to prove the stability. The Lyapunov function is:

$$E_l = \min\{\|r_a - A_{min}\|, \|A_{max} - r_a\|\}$$

```
**Input:** $r_a, d_r(k)$
**Output:** $d_r(k+1)$
**Parameter:** $A_{min}, A_{max}, \alpha, \beta$
**State:** $lstate, \delta_i, \delta_d$
if $(r_a > A_{max})$
    if $(lstate == INC) \| (lstate == NOR)$
        $d_r(k+1) = d_r(k) + d_r(k) * \alpha;$
        $\delta_i = d_r(k+1) - d_r(k);$
    if $(lstate == DEC) \| (lstate == OVER)$
        $\delta_d = \delta_d/2;$
        $d_r(k+1) = d_r(k) + \delta_d;$
        $lstate = OVER;$
    if $(lstate == BELOW)$
        $\delta_i = \delta_i/2;$
        $d_r(k+1) = d_r(k) + \delta_i;$
else if $(r_a < A_{min})$
    if$(lstate == DEC)$
        $d_r(k+1) = d_r(k) - d_r(k) * \beta;$
        $\delta_d = d_r(k) - d_r(k+1);$
    if $(lstate == OVER)$
        $\delta_d = \delta_d/2;$
        $d_r(k+1) = d_r(k) - \delta_d;$
    if $(lstate == INC) \| (lstate == BELOW)$
        $\delta_i = \delta_i/2;$
        $d_r(k+1) = d_r(k) - d_r(k) * \delta_i;$
        $lstate = BELOW;$
    $lstate = DEC;$
else
    $lstate = NOR$
end
```

Fig. 14. The adaptive duty cycle algorithm with stabilization

When no transition happens, either $\|r_a - A_{min}\|$ or $\|A_{max} - r_a\|$ is a decease function, guaranteed by decreasing function $r_a = f(D_s, d_r)$. When transition happens, the algorithm guarantees infinite smaller step size of duty cycle is added or subtracted. Function $E_l$ is still a decreasing function. By Lyapunov theorem, this algorithm stabilizes to a static point. Given the continuity of function $r_a = f(D_s, d_r)$, activity ratio stabilize to a point in $A_{min}, A_{max}$. ■

### E. Implementing Duty Cycle Adaptation

Once a node changes its duty cycle, all its neighbors should be notified. In our protocol, simultaneous broadcast serves this purpose effectively, i.e, whenever a node's duty cycle is changed, a simultaneous broadcast is issued to the neighbors.

## V. Protocol Evaluation

The asynchronous discovery protocol and synchronous communication with duty cycle adaptation have been implemented on the TelosB platform, using TinyOs 1.1. In this section, we show their performance in experiments in a realistic mobile environment. In addition, we also evaluate the performance of the adaptation algorithm in simulations.

## A. Asynchronous Discovery Protocol

This experiment was based on the mutual discovery protocol schedule described in Theorem 4. We used $9$ TelosB motes; one attached to a laptop represented an established network, while the other $8$ represented mobile nodes simultaneously attempting to join the network. The experiment was conducted in an engineering building, which is a noisy RF environment, including at least 3 802.11 access points in the vicinity of the nodes. All nodes were placed within communication range of the receiver. To guarantee random shift of our clock, all nodes were reset after every experiment. The parameters of the experiment are:

| | |
|---|---|
| Slot length | 10 ms |
| Frame length | 2500 slots |
| Packet length | 20 bytes |

This implies that the frame length is about $25s$ and the duty cycle is $101/2500 \approx 4\%$. Figure 15 shows the time required to discover the nodes. The figure reveals that time taken to discover all the $8$ nodes does not exceed the frame length in 240 trials.

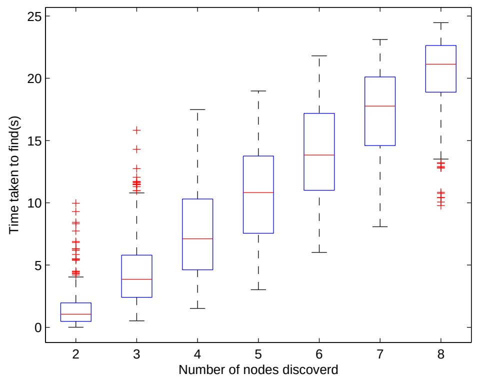

Fig. 15. Time taken to discover neighbor

## B. Synchronous Protocol with Duty Cycle Adaptation

The experiments described in this subsection were conducted in the same environment as the asynchronous discovery experiment. We use a larger slot length $20ms$. Initially, nodes (receivers) run with frame length of $50$ slots, by which we can compute the duty cycle to be $1/50 = 0.02$. A time synchronization protocols also running on those nodes. We conduct two experiments:

| | |
|---|---|
| Slot Length | 20 ms |
| Initial Frame Length | 50 slots |
| Packet Length | 30 bytes |

**Experiment 1: (varying traffic)** In this experiment, we use $10$ senders and $1$ receiver. The duty cycle for each sender is $3/256 \approx 1.2\%$, i.e., for each frame the probability of a transmitter attempting to transmit when the receiver is on is $0.012 \cdot N$. The receiver duty cycle, i.e., the receiver frame length, is updated every $32$ frames, which we'll call a round. The initial aggregate transmission rate is $0.012 \cdot 10 = 12\%$. After $100$ rounds, the number of transmitters drops to $5$ (i.e., about $6\%$), and after $140$ round it drops to $2$ nodes, (i.e., about $2.4\%$). Figure 16 shows the throughput and the receiver duty cycle. Initially, receiver duty cycle is far below incoming traffic, so it increases exponentially to reach the steady state. When incoming traffic decreases, the receiver duty cycle matches the incoming traffic fairly well. The collision rate and activity ratio are shown in Figure 17. The algorithm maintains a steady activity ratio and low collision rate.

**Experiment 2: (communication interference)** The previous

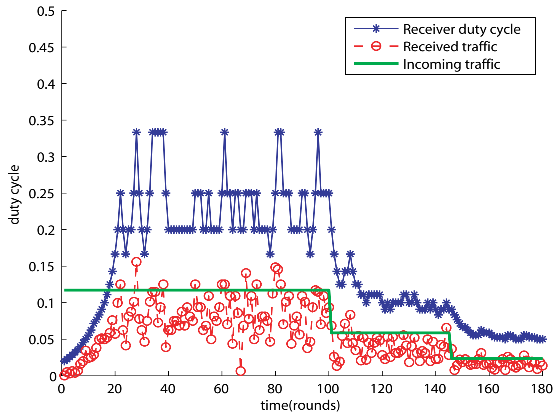

Fig. 16. Experiment 1: duty cycle adaptation under varying traffic

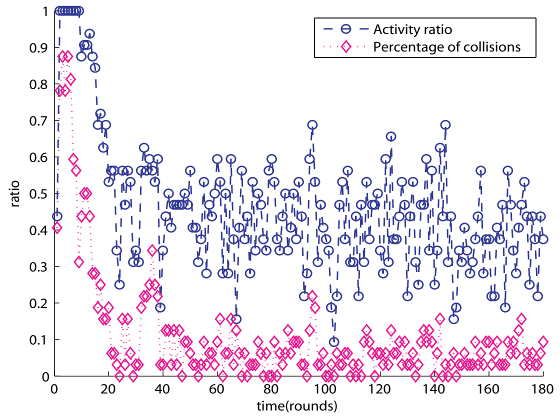

Fig. 17. Experiment 1: activity ratio and collisions

experiment used only one receiver. This experiment is designed to show the effect of our protocol with several receivers in the region. Here we only have $5$ nodes sending to the instrumented receiver, but the other $5$ nodes send to a large number of other receivers. All transmitters potentially interfere with each other.

The $5$ nodes that are transmitting to the instrumented node use a duty cycle of $1.2\%$, as before. However, the $5$ nodes that are transmitting to the other nodes use a higher duty cycle of $2\%$. As Figure 18 shows, the instrumented receiver's duty cycle converges to the rate of the incoming traffic. The instrumented receiver also occasionally overhears messages intended for other receivers and fails to hear its own messages due to collisions with messages destined for the other receivers. However, Figure 19 shows fewer collisions occur than in the previous experiments; this is because the other receiver's slot rarely coincides with the reception slot of the instrumented receiver.

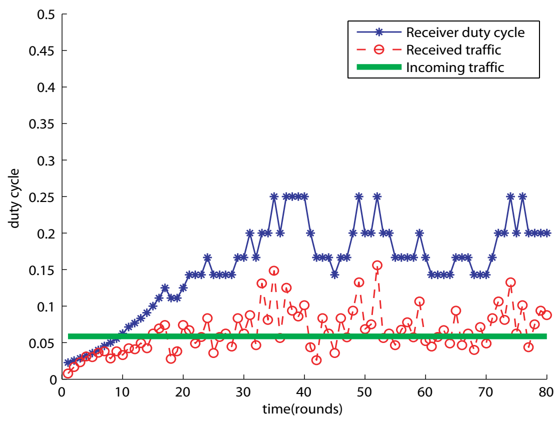

Fig. 18. Experiment 2: duty cycle adaptation under interfering traffic

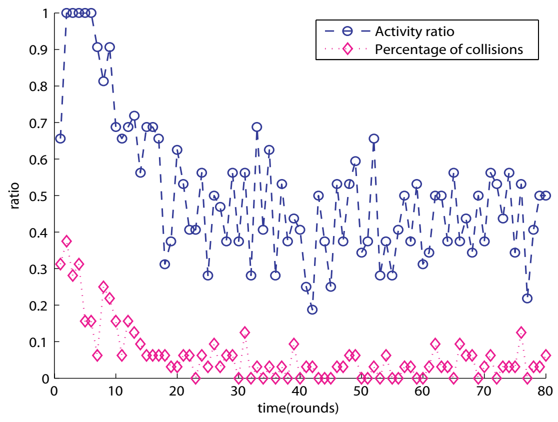

Fig. 19. Experiment 2: activity ratio and collisions

To further verify the algorithm, we have run several simulations.

**Simulation: (Increasing traffic without collisions)** In this simulation, two nodes are within communication range, and adaptive duty cycle algorithm is running on the receiver. The sender sends out packets at each second with probability $p$, which is initiated as $0.01$ and linearly increased to $0.2$. As Figure 20 shows, receives duty cycle increases linearly with incoming traffic. In steady state, a slight higher receiver duty cycle is provided to compensate the randomness of incoming traffic.

**Simulation: (Increasing traffic with collisions)** There are 2 senders and one receiver in the network. All the senders can send message directly to the receiver. At every second,

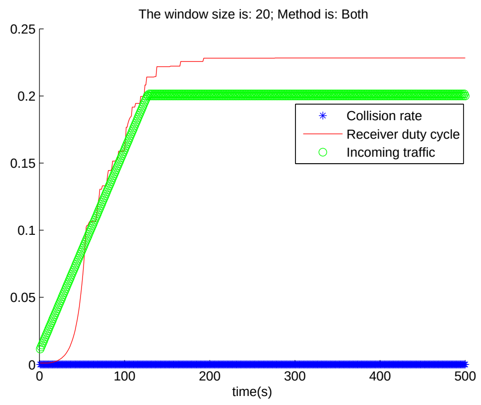

Fig. 20. Duty cycle adaptation for increasing traffic without collisions

each node sends out packets with probability $p$. The traffic for those three senders are:

- Sender 1: $p : 0.01 \longrightarrow 0.2$
- Sender 2: $p : 0.2$

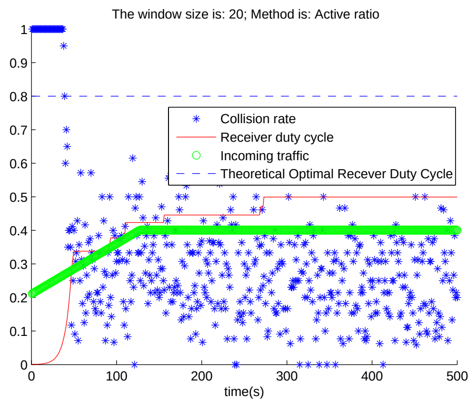

Fig. 21. Duty cycle adaptation for increasing traffic with collisions

Results are shown in Figure 21. Initially, the receiver duty cycle $(0.02)$ is below the required duty cycle $(0.21)$. By using activity ratio information, the receiver duty cycle can response duty cycle change quickly and stably.

**Simulation: (Decreasing traffic without collisions)** The sending probability $p$ is initiated as $0.02$ and linearly decreased to $0.01$. When traffic decreases, our algorithm can follow it closely as shown in Figure 22.

**Simulation: (Decreasing traffic with collisions)** We uses the same setting as before, but the traffic of those three senders are decreasing:

- Sender 1: $p : 0.2 \longrightarrow 0.02$
- Sender 2: $p : 0.2$

**Simulation: (SeeSaw traffic)** In this simulation, there are $10$ senders and $1$ receiver within communication range. Every sender generates traffic randomly. The average rate starts at $0.001$ packets per second and linearly increases to $0.05$ packets per second and then decreases to back to the initial rate before repeating the cycle.

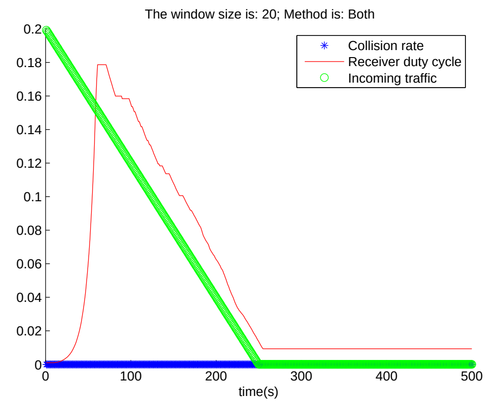

Fig. 22. Duty cycle adaptation for decreasing traffic without collisions

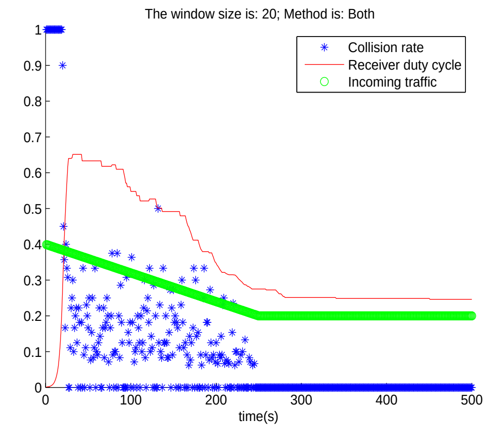

Fig. 23. Duty cycle adaptation for decreasing traffic with collisions

- Sender $i$: $p : 0.001 \rightarrow 0.05 \rightarrow 0.001 \cdots$

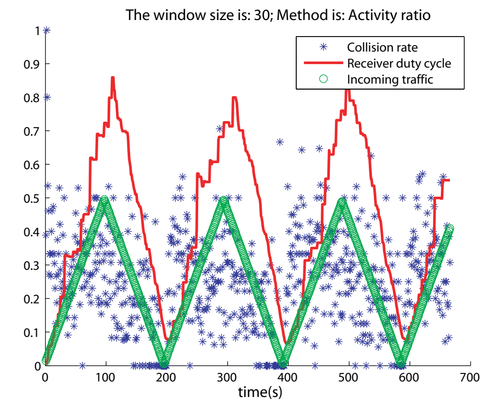

Fig. 24. Simulation of duty cycle adaptation in SeeSaw traffic

As Figure 24 shows, even for a $10$ node, highly varying traffic network, our adaptive algorithm works well.

## VI. Conclusion

System level analysis often suggests that WSNs should strive for duty cycles on the order of $1\%$ while actually communicating at a duty cycle above $0.1\%$. In practice there are very few examples of deployments that achieve this level of performance. The key reason is the difficulty of getting the receiver off most of the time. Synchronous MACs can achieve this level of performance, but have not been widely adopted largely because of the system implications for discovery and bootstrapping. These problems have seemed insurmountable for mobile networks.

This paper shows an approach for getting a high efficiency asynchronous discovery protocol to co-exist with an even higher efficiency synchronous communication protocol. The result should allow $1\%$ to $5\%$ duty cycles to be achieved, even for mobile networks. This in turn should allow for a whole new class of applications based on long life, battery powered, mobile nodes. The result could be as substantial as increasing the battery life from a few days to a year or from a couple of weeks to several years.

## References

[1] H. Cao, K. W. Parker, A. Arora. O-MAC: a receiver centric power management protocol, Proceedings of the 14th IEEE International Conference on Network Protocols (ICNP), Nov. 2006
[2] W. Ye, F. Silva, and J. Heidemann. Ultra-Low duty cycle MAC with scheduled channel polling, Proceedings of the 4th ACM Conference on Embedded Networked Sensor Systems (SenSys), Boulder, Colorado, USA, Nov. 2006.
[3] G. Halkes and K. Langendoen. Crankshaft: An energy-efficient MAC-protocol for dense wireless sensor networks,European conference on Wireless Sensor Networks (EWSN), Delft, The Netherlands, Jan. 2007.
[4] Y. Li, W. Ye, and J. Heidemann. Energy and latency control in low duty cycle MAC protocols. In Proceedings of the IEEE Wireless Communications and Networking Conference, New Orleans, LA, USA, Mar. 2005.
[5] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh. Power-saving protocols for IEEE 802.11-based multihop ad hoc networks. In Proc. of INFOCOM, 2002.
[6] R. Zheng, J. C. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks., in Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc), 2003, pp. 35-45.
[7] G. Yee, E. Anderson, R. Han. X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks, Proceedings of the 4th ACM Conference on Embedded Networked Sensor Systems (SenSys), Nov. 2006.
[8] L. Doherty, B. A. Warneke, B. E. Boser, K.S.J. Pister. Energy and performance considerations for smart dust, International Journal of Parallel Distributed Systems and Networks, Volume 4, Number 3, 2001, pp. 121-133.
[9] W. Ye, J. Heidemann, and D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Jun. 2002.
[10] T. van Dam and K. Langendoen, An adaptive energy-efficient mac protocol for wireless sensor networks, in Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys), 2003, pp. 171-180.
[11] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys), Baltimore, MD, Nov. 2004.