EBS: Decentralised Slot Synchronisation for Broadcast messaging for Low-power Wireless Embedded Systems

Poonam Yadav, Julie A. McCann
Department of Computing
Imperial College London
(pooyadav,jamm)@doc.ic.ac.uk

ABSTRACT

In this paper, we present a decentralised scheme that facilitates reliable network wide broadcast messaging without the requirement of strict time synchronisation, for duty-cycled low-power wireless embedded systems. In this emergent broadcast slot (EBS) scheme, devices coordinate their wake-up periods with their neighbours to exchange schedule information locally. This leads to the emergence of local slot synchronisation without the need for either network-wide synchronisation or a centralised time synchronisation element. We theoretically show that this scheme converges faster than similar emergent and gradient-based approaches, which we confirm by evaluation on real test-beds. We also show that our scheme exhibits lower overheads while being more tolerant to disturbances caused by faulty nodes, wireless link failures, contention and interference in presence of deterministic propagation delays.

1. INTRODUCTION

In recent years, low-power wireless embedded systems have gained wide attention due to their suitability for monitoring complex physical world phenomena. The real world deployment and maintenance of these embedded systems [1, 2] is a challenging task because of their resource constraints and dynamic nature. Furthermore, many of these systems involve battery powered devices (hereafter nodes) that make use of low-power radios such as Bluetooth [3] and Zigbee [2] for wireless communication. However, many low-power radios have a limited communication range of up to 100 meters, so, to ensure messages are propagated from source to sink, multi-hop networking has been adopted. To optimise battery power usage and elongate the network's lifetime, duty-cycling is used; whereby nodes utilise idle time by inducing regular sleep periods. Duty-cycled nodes with multi-hop networking (DCM) have proven themselves many times and in recent years many differing schemes have been proposed for low-power devices [4]. Beyond energy constraints, efficient and reliable data delivery remains a challenge due to the presence of unreliable wireless links and limited bandwidth in low-power radios. Therefore, efficient route management mechanisms are of paramount importance to DCM networks. Route management mechanisms typically use control messages that are exchanged among nodes to maintain and optimise the route structures. This data may be piggy-backed on top of data traffic so that it effectively captures the dynamic nature of such networks, for example node failures etc., can be identified and routed around. Here, we describe these control messages as broadcast messages that are propagated either to local neighbourhoods or network-wide.

Clever route management and DCM techniques aim to maximise network's lifetime and operation but at a cost. The control messages constitute extra overhead traffic in the network, which consumes a non-trivial level of bandwidth. Further, duty-cycling incurs management overheads to ensure nodes synchronise sleep-awake schedules to receive broadcast messages. Relay nodes in multi-hop networking must receive and forward data traffic to propagate broadcast messages. Due to the half-duplex nature of low-power wireless radios, relay messages cannot be sent and received simultaneously, which effectively halves bandwidth and throughput. This is exacerbated by the hidden terminal problem, and contention resolution mechanisms must address this to avoid message loss due to collisions. As of now, there are no simple straight-forward solutions that handle control messages in DCM networks by taking all these issues into account. Consequently, in this paper we present schemes that aim to address this problem by efficiently handling control messages (broadcast messages) in DCM networks so that they achieve the required throughput.

Normally, for a node to broadcast control messages to all (or some) nodes of the network, some form of local synchronisation is required to ensure that the receiving nodes are awake to receive the control messages being sent to it. Therefore, to achieve this time coordination among nodes, a number of *centralised* time synchronisation algorithms have been proposed that are optimised for low-power wireless networks [5]. However, they are not suitable for DCM networks because of issues pertaining to maintenance overheads, synchronisation de-

lays, and single points of failure [6].

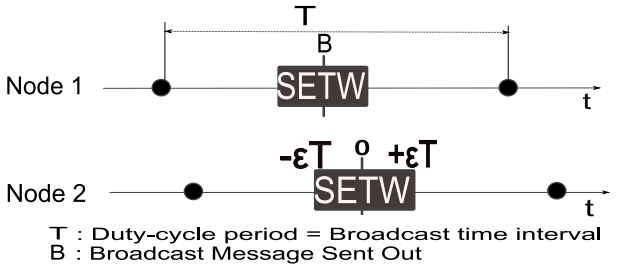


Figure 1: EBS mechanism emerges overlapped SETWs for broadcast messages exchange. Once EBS converges, nodes awake only in their respective SETWs.

To this end, alternative decentralised approaches have been proposed that achieve local synchronisation but are not optimised for DCM networks. In this paper, we propose an Emergent Broadcast Slot (EBS) scheme, which is inspired by firefly-based synchronisation to permit broadcast message delivery. In EBS, a Synchronisation Error Tolerance Window (SETW) of a node emerges in such a way that it partially overlaps the SETWs of its neighbourhood nodes (see Fig. 1). A node uses its SETW as a time slot for broadcast message exchanges with its 1-hop neighbourhood. We accordingly describe this as loose synchronisation , as network-wide synchronisation is not required. Therefore, this reduces the number of synchronisation messages, which in turn reduces overheads and contention in the network. A node transmits its broadcast message half-way through this window, which is in turn received by the nodes which are currently also in their SETWs. A node has to ensure that it synchronises with its one-hop neighbourhood nodes before switching to duty-cycle mode. Note that here synchronisation does not mean the actual time synchronisation but overlapped SETWs for efficient broadcast message exchanges.

Due to the presence of unreliable and asymmetric wireless link qualities, the neighbourhood size of a node changes and we found that this affects the convergence and stability of EBS. To solve this problem, we relax the requirement of total neighbourhood synchronisation; instead a node must synchronise only with a sub-set of its neighbourhood nodes before switching to duty-cycle mode. We therefore introduce the notion of a 'degree' of Synchronisation. Once a node reaches a predefined synchronisation threshold it switches to duty-cycle mode; whereby it subsequently awakens during its SETW. The error tolerance window plays an important role for the nodes that span two or more neighbourhoods in DCM networks, as it allows those nodes to maintain a sufficiently long wake-up-time period to allow coordination with *all* of their neighbourhoods while maintaining only one wake-up interval within a broadcast message time interval; thus minimising the effects of hidden terminals. In summary, the contributions of this paper are as follows:

- We present an Emergent Broadcast Slot scheme, which allows nodes to maintain a single broadcast slot to exchange messages with its one-hop neighbourhood nodes.
- This scheme converges quickly to small broadcast slots as compared to other bio-inspired and gradient scheme because of its use of *Refractory Periods* and *Synchronisation Thresholds*.
- EBS is self adaptive to broadcast messages intervals and local connectivity of the node in the network.
- The *Synchronisation Threshold* allows EBS to cope with node failures and lossy wireless links.
- To the best of our knowledge, this is the first slot synchronisation scheme for DCM networks.

The organisation of the paper is as follows: In Section 2, we present related work on decentralised time synchronisation schemes and firefly based synchronisation approaches. In Section 3, we present an overview of the system model, overview of EBS, and its configuration parameters and their effects are analysed in Section 3.3. We present two implementation of EBS in Section 4 and discuss their suitability and convergence in the presence of random delays. We briefly discuss experiment setup in Section 5. The EBS performance and comparative results are presented in Section 6. We then conclude summary along with future work in Section 7.

2. RELATED WORK

In general, low-power nodes consist of relatively inexpensive hardware components, meaning that clocks typically drift, which enforces regular network wide resynchronisation, increasing overheads. Therefore, centralised time synchronisation mechanisms are not ideal for low-power devices. Furthermore, the presence of lossy wireless links means that these synchronisation packets may be dropped and may need to be sent multiple times. Therefore, distributed coordination is a more appropriate solution, which provides local synchronisation that allows broadcast message exchange within a local one-hop neighbourhood. Given that the local synchronisation is more efficient in DCM networks than globally centralised synchronisation, there have been a number of local synchronisation mechanisms proposed such as gradient-based [7] and bio-inspired algorithms [8].

In the literature, many believe that bio-inspired algorithms have higher overheads than gradient based.

However, gradient based schemes are not very agile and are unable to cope with dynamism such as node failures [9]. On the other hand, the bio-inspired nature of Firefly based synchronisation shows that it can cope with failure. In 1990 Strogats and Mirollo (M&S) presented a mathematical model M&S [10] based on the theory of pulse-coupled oscillators (PCO) to capture the synchronisation behaviour observed in nature, for example, when fireflies flash in unison. Subsequently, [10] produced a model explaining the appropriate configurations under which a fully connected PCO system can achieve synchronisation; the phase advancement function presented therein can be simplified to Eq. 1. It represents the phase advancement of a PCO, which observes firing [1] from other PCO at time t.

$$\phi(t^+) = \begin{cases} \phi(t) + \alpha\phi(t) & \text{if } \phi(t) + \alpha\phi(t) < 1 \\ 0, & \text{Otherwise} \end{cases} \quad (1)$$

Here, $\phi(t) \in [0, 1]$, is the phase of an oscillator at time t, and α is coupling factor. Also note that, when $\phi(t) + \alpha\phi(t) \geq 1$, $\phi(t^+)$ resets to 0 (i.e. it does not fire immediately) instead of 1 (which means it fires instantly, and where propagation delays exist between the PCOs, delaying the reception of the fire messages, leads to both PCOs chasing each other). Conversely, this is overcome if the PCOs are weakly coupled (α is smaller), but the system takes longer to converge. To adapt this model to wireless systems two concepts were introduced; *reach-back* [8] and *refractory periods* [11]. Werner-Allen et al. [8] presented a *reach-back* Firefly Algorithm (RFA) that accounts for communication latencies by modifying the original firefly model to allow nodes to use information from the past to adjust the future firing [1] phase. In this algorithm, instead of firing immediately, the node keeps a record of the phase advancement of the current period and advances its phase immediately at the start of the next time period, T. The new phase after this advancement, is calculated using the following Eq. 2.

$$\phi(t^+) = \begin{cases} \phi(t) + \alpha\phi(t) & \text{if } \phi(t) + \alpha\phi(t) < 1 \\ 1, & \text{Otherwise} \end{cases} \quad (2)$$

Here, $\phi(t^+)$ is the new phase value, $\phi(t)$ is the previous phase value. Note that, when $\phi(t) + \alpha\phi(t) \geq 1$, $\phi(t^+)$ resets to 1 unlike in [10] (refer to Eq.1).

Degesys et al. [11] presented the concept of *refractory periods*. The *refractory periods*, represented by $\phi(t)_R \in [0, 1]$, are time periods that start just after a node fires, and during this, the firing node ignores fire messages from other nodes; the phase advancement function is shown in Eq. 3.

$$\phi(t^+) = \begin{cases} \phi(t) & \text{if } \phi(t) < \phi(t)_R \\ \phi(t) + \alpha\phi(t) & \text{if } \phi(t) > \phi(t)_R \,\&\&\, \phi(t) + \alpha\phi(t) < 1 \\ 0, & \text{Otherwise} \end{cases} \quad (3)$$

Global stable synchronisation, using inhibitory pulse coupling in systems with delay, has been proposed by Klinglmayr et al. [12]. Here the phase adjustment is calculated using $\phi(t^+) = (1+ \alpha)\phi(t)$ (as in RFA) but when the PCO fires at $\phi(t) = 1$, instead of resetting the ϕ to 0, it resets to $1 - |\alpha|$ which shortens its cycle to a length of $|\alpha|$. In other words, the firing node also advances its own phase when $\phi(t) = 1$, along with all other nodes that observe this fire message.

Tyrrell et.al [6] proposed Meshed Emergent Firefly Synchronisation (MEMFIS), which is a modified and extended synchronisation model based on the PCO model; it relies on the detection of a synchronisation indicator common to all nodes, which is embedded into each packet along with the payload data. The time-stamp of the received synchronisation indicator serves as input to slot synchronisation algorithms at the physical layer that are based on the original M&S model.

Cui and Wang [13] proposed RFA with a Late Sensitivity Window (LSW) in which nodes do not advance their phase. This improves the percentage of nodes synchronised, and also reduces the time to synchronise as well as the corresponding broadcast message overheads.

So bio-inspired decentralised synchronisation has been explored to some extent, however, they have yet to take duty-cycling (DCM networks) into account; i.e. the re-synchronisation required after a sleep period, nor have they explored the effects of asymmetric wireless links in dense networks that means many of the synchronisation messages may be missing.

Therefore, we propose EBS, a distributed firefly based synchronisation scheme, which is simple and guarantees fast synchronisation within a given error tolerance window in the presence of asymmetric and lossy wireless links in dense networks. We also evaluate and analyse the optimisation parameters of EBS for DCM networks.

Though EBS is inspired by MEMFIS [6], *reach-back* and *refractory periods*, it exhibits better performance for DCM networks in terms for low duty-cycle and throughput achieved; shown further in Section 6.

3. EMERGENT BROADCAST SLOT SCHEME

In this section, analyse the EBS scheme by presenting the system model and individual analysis of EBS parameters, which effects the EBS performance and its convergence.

3.1 System Model

We model the DCM wireless network as a graph, G(S, L), that is composed of S nodes represented by the set

[1]Firing refers to "synchronisation message broadcast" in Firefly based synchronisation algorithms

S = (1, 2,. . . , S) where L is a set of all transmission links between these S nodes, i.e.;

$$\forall s \in \mathrm{S} \qquad L = \bigcup_{s_i, s_j \in \mathrm{S}} \langle s_i, s_j \rangle \tag{4}$$

Here, $\langle s_i, s_j \rangle$ define a link iff the node s_i is within the transmission range of node s_j.

The m-hop neighbourhood N^m is composed of the node itself and its m-hop neighbours. Thus, the 1-hop neighbourhood of the node d_i is given by:

$$N_i{}^1 = \{s_i\} \bigcup \left(\bigcup_{\langle i,j \rangle \in L} \{s_j\} \right) \tag{5}$$

The local degree of connectivity of node s_i can be represented by $d_i = \| N_i{}^1 \|$. The average degree of connectivity for the whole network can be represented as:

$$\bar{d} = \frac{\sum_{i=1}^{\mathrm{S}} \| N_i{}^1 \|}{N} \tag{6}$$

We assume that every node sends a broadcast message asynchronously after a periodic time interval, T. We define the periodic function $f(t+T) = f(t)$ with a periodic time period T and $\phi(t) = f(t/T)$, where $\phi(t) \in [0, 1]$ is the phase that has elapsed since the previous control message was broadcast by the node.

We define $\phi'(t) = (1 - \phi(t))$ as the remaining phase left, after which a node is scheduled to broadcast its next control message.

Definition 1. *The Synchronisation Error Tolerance Window (SETW) is the period within the regular broadcast time interval, T, such that SETW $= [-\varepsilon T, \varepsilon T]$, $\varepsilon \in (0, 0.5)$ and its width, $W = 2 \times \varepsilon \times T$.*

Definition 2. *The Synchronicity, S(%) , of a node in a network is the percentage of the total number of neighbours whose broadcast messages it can hear during its SETW.*

Definition 3. *The Synchronicity Threshold, $S_{Th}(\%)$, of a node is defined as the minimum percentage of nodes from its neighbourhood that it must hear during its SETW.*

3.2 EBS Overview

In a DCM network, each node has to exchange broadcast messages with its 1-hop neighbourhood nodes (i.e. the nodes in its communication range). For this purpose, each node remains awake only for its SETW (broadcast time slot)(refer to Definition 1) within every periodic time interval, T. Recall that the SETWs of the synchronised neighbourhood nodes should be partially overlapped in such a way that each node can exchange broadcast messages with its 1-hop neighbourhood nodes during this SETW.

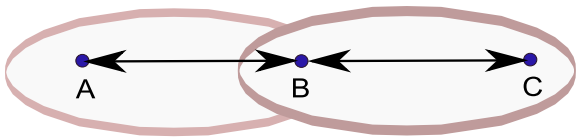


Figure 2: Node B is part of two overlapped neighbourhoods such that it can hear both A and C, whereas A and C could not hear each other.

In a fully connected network topology, where all nodes are in communication range of each other, this window can be fully overlapped, but in multi-hop networks where a node belongs to more than one neighbourhood, as shown in Fig. 2, it becomes necessary that its SETWs partially overlap with the SETWs of the other nodes that are in its neighbourhoods. The SETW of a node also acts as a *refractory period* within T. The *refractory period* is a time window in which a node does not advance its phase if it hears any broadcast message from its 1-hop neighbourhood nodes. The EBS scheme consists of an *Initialisation phase* and *Duty-cycled phase*. The EBS scheme does not have synchronisation message overhead as it piggy-back the broadcast messages with EBS byte. However, in absence of the broadcast messages, it can broadcast its own EBS messages to achieve synchronisation.

In the *initialisation phase*, all nodes in the network are 100% duty-cycled and they remain awake until they establish their SETWs and converge. For a node to declare itself as synchronised [2], each node has to keep a record of its *Synchronicity*, $S(\%)$ (refer to Definition 2) value for its SETW. Once the node's $S(\%)$ value goes above or equals to the pre-set $S_{Th}(\%)$ (refer to Definition 3), the node declares itself as synchronised. To achieve the required $S(\%)$, each node in the EBS scheme uses the phase advancement function in the *initialisation phase* that can be is described as:

$$\phi(t^+) = \begin{cases} \phi(t) & \text{if } \phi(t) < \varepsilon \,||\, \phi(t) > (1-\varepsilon) \\ 1 - g(1-\phi(t)) & \text{if } \varepsilon < \phi(t) < (1-\varepsilon) \end{cases} \tag{7}$$

Here, $\phi(t^+)$ is the new phase value, $\phi(t)$ is the previous phase value, $g(1-\phi(t))$ is the phase advancement function, $g(1-\phi(t)) = \sigma(1-\phi(t))$, $\sigma \in (0,1)$ and $\varepsilon \in (0, 0.5)$(refer to Definition 1). Here, $2 \times \varepsilon$ is the node's wake-up time fraction in the duty-cycle time period, T.

In terms of $\phi'(t)$, the Eq. 7 can be presented as:

[2]The synchronisation decision is local to every node.

$$\phi'(t^+) = \begin{cases} \phi'(t) & \text{if}((1-\varepsilon) \leq \phi'(t)) \,||\, (\phi'(t) \leq \varepsilon) \\ g(\phi'(t)) & \text{if}(\varepsilon \leq \phi'(t) \leq (1-\varepsilon)) \end{cases} \tag{8}$$

Once synchronised, the nodes switch to the *duty-cycle phase* and wake-up only for their respective SETWs to exchange messages. In the *duty-cycle phase* also, all nodes keep updating their $S(\%)$ values. The nodes whose $S(\%)$ value falls below the $S_{Th}(\%)$, switch back to the *initialisation phase*; returning themselves to 100% duty-cycling. The various reasons for a drop in a node's $S(\%)$ value can be due to clock drift, a change in network density, etc. This method of switching back has the advantage that, unlike gradient based schemes [7], one node's disturbance does not perturb the whole network; therefore, our scheme is both agile and tolerant to change. Moreover, in EBS, a node which is part of more than one overlapped neighbourhood, for example node B in Fig. 2, exchanges broadcast messages with all its 1-hop neighbours during the SETW; that means only one broadcast slot within each period, T. EBS only requires the node to wake-up once within the period, T, rather than multiple times. Recall that multiple wake-ups within a T, cause higher energy consumption in radio state transitions, which occurs in other distributed or centralised slot or time synchronisation schemes in multi-hop networks [14, 15].

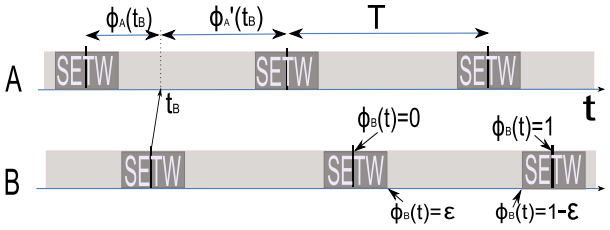


Figure 3: Two nodes A and B are 100 % duty-cycled in the initialisation phase. This figure shows the phase of the node A just before it receive broadcast message from B at time t_B.

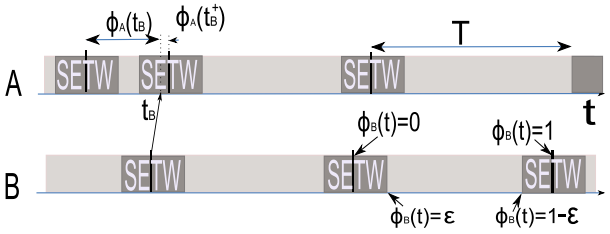


Figure 4: This figure shows the phase of the node A after phase advancement due to the received broadcast message from B at time t_B in their initialisation phase.

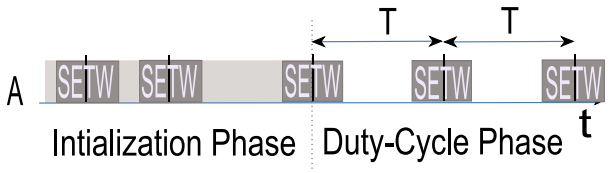


Figure 5: A node is 100% Duty-cycled in the Initialisation phase and wake-up only in its SETW during the Duty-cycled phase.

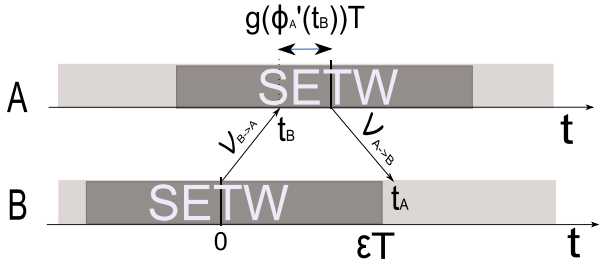


Figure 6: Overshooting condition occurs when node B receives broadcast message from node A at its time t_A, which is out of its SETW.

Suppose, a given node A receives a broadcast message from another node B at time t_B. It checks the time left to transmit its next broadcast message that is represented by $\phi'_A(t_B)T$. According to the Eq. 8, if $(\varepsilon \leq \phi'_A(t_B) \leq (1-\varepsilon))$(see Fig. 3), then it shortens its phase $\phi'_A(t_B)$ in such a way that it either transmits its broadcast message immediately (by setting $\phi'_A(t_B^+)$ to 0) or calculates new $\phi'_A(t_B^+)$ (see Fig. 4). To achieve rapid synchronisation, the node transmits its broadcast message immediately and after transmission it resets the next broadcast message transmission time to T by setting $\phi_A(t_B) = 0$ and $\phi'_A(t_B) = 1$. However, recall that in low-power wireless networks, all nodes within the 1-hop neighbourhood are able to listen to the same broadcast message and if they were to broadcast their messages on receipt of this message at same time, then there will be many messages in the local area; leading to packet collisions. To avoid such collisions, nodes delay their broadcast message transmission by $(\phi'(t^+))$ which is equal to $g(\phi'(t))$.

The appropriate values of ε and σ depend on a number of factors such as the number of nodes in a 1-hop and 2-hop neighbourhoods, the value of T, and message propagation delays (discussed further in Section 3.3). We call $\phi'(t^+)$ the remaining time to the next broadcast message, so the new $\phi'(t) = \phi'(t^+)$ and $\phi(t) = 0$ since $\phi(t)$ resets after the synchronisation advancement.

3.3 Effects of EBS Parameters

The performance of the EBS scheme depends on the

values of its three configuration parameters: ε, σ and $S_{Th}(\%)$. The appropriate values of these parameters are based on a few network parameters such as network density and uniformity of the nodes in the network. Other issues that affect the performance of EBS scheme are the periodic interval (T) and the message propagation delays (ν). For a given T and average degree of connectivity of a network, we first analyse ε, σ, and $S_{Th}(\%)$ respectively.

3.3.1 *Effects of* ε

The ε parameter plays an important role in the EBS scheme because it is directly proportional to the duty-cycle of the DCM network. Once the *initialisation phase* is over, in *duty-cycle phase*, the nodes remain awake only for their respective SETWs to reduce energy consumption as shown in Fig. 5. The size of the SETW is defined in time units as $||SETW|| = 2\varepsilon T$. Hence, the duty-cycle ($\triangle$) of the network in the *duty-cycle phase* is given as:

$$\triangle = \frac{||SETW||}{T} \times 100 \tag{9}$$

Therefore, the primary trade-off is to minimise the $||SETW||$ as much as possible to reduce the energy consumption while maintaining the required throughput. However, regarding the convergence of the EBS scheme, the size of the SETW plays a reverse role, that is, the larger the $||SETW||$, the quicker the EBS scheme converges. Therefore, this requires us to find the minimum size of SETW that can guarantee quick convergence without overshooting as shown in Fig. 6. Th overshooting situation occurs when a node A receives a broadcast message from node B after a propagation delay ($\nu_{B\to A}$) (see Fig.6). Then node A delays its own broadcast message transmission by $g(\phi'(t_B))T$ according to Eq. 8. Node B receives this message from node A at its local time t_A, after a propagation delay ($\nu_{A\to B}$). This is the overshooting situation; whereby t_A lies outside of the refractory period which makes node A to advance its phase again, so node B then follows and so on; they begin to chase each other. This condition only happens in the *initialisation phase* due to the small size of the SETW. To avoid this overshooting condition, we can limit the lower bound of the εT to:

$$\varepsilon \times T \geq (\nu_{A\to B}) + g(\phi'(t_B))T + (\nu_{B\to A}) \tag{10}$$

Assuming that ($\nu_{A\to B}$) and ($\nu_{B\to A}$) are approximately equal, we replace them with ν, so the Eq. 10 can be represented as:

$$\varepsilon \times T \geq g(\phi'(t_B))T + 2\nu \tag{11}$$

$$\varepsilon \geq \frac{g(\phi'(t_B))T + 2\nu}{T} \tag{12}$$

Also, recall that $||SETW|| = 2\times\varepsilon\times T$ and the maximum size of the SETW in one time period could be T, so, $||SETW|| \leq T$, which leads to $\varepsilon \leq 0.5$. Therefore, Eq. 12 can represented as:

$$0.5 \geq \varepsilon \geq g(\phi'(t_B)) + \frac{2\nu}{T} \tag{13}$$

Theoretically, in Eq. 13, we can set $g(\phi'(t_B)) = 0$ (when we consider the immediate transmission of broadcast messages) and also, when $T \gg 2\nu$, we can approximate $\frac{2\nu}{T} \simeq 0$ (means no overshooting), which leads to $\varepsilon \simeq 0$. But, in practice ($\frac{2\nu}{T} > 0$) and so the $\varepsilon > 0$. So, the lower value of ε is bounded by $\frac{2\nu}{T}$ for a given value of T[3]. The optimal value of ε is chosen by considering duty-cycle requirements and density of the network. To account for this, we assume the maximum time for a node to receive and process a broadcast message from its neighbourhood is β; assuming that each neighbourhood node sends a single message in T. So, the maximum time a node is required to be awake to receive and process messages from its neighbourhood of size $\bar{d}$ is: $\beta \times \bar{d}$; thus the size of SETW should be $||SETW|| = \beta \times \bar{d} + 4\nu$.

$$2 \times \varepsilon_{opt} \times T \geq \beta \times \bar{d} + 4\nu \tag{14}$$

$$\varepsilon_{opt} \geq \frac{\beta \times \bar{d} + 4\nu}{2T} \tag{15}$$

3.3.2 *Effects of* σ

The σ is a coupling parameter that defines the rate of convergence in the *initialisation phase*. The phase advancement function that is first defined in Eq. 8 and Eq. 7 is, $g(1-\phi(t)) = \sigma(1-\phi(t))$ or $g(\phi'(t)) = \sigma\times\phi'(t)$, here $\sigma \in (0, 1)$. Recall that a large value of σ slows the convergence of the EBS scheme, which leads to a longer *initialisation phase*, resulting in high energy consumption because the nodes are 100% duty-cycled. For faster synchronisation, smaller values of σ are chosen. It is also possible to set $\sigma = 0$, but that leads to message collisions in the network if it is dense. This is because, $\phi'(t^+) = 0$, that means a node immediately transmits its broadcast message, and if all the nodes in the local neighbourhood do the same, then it leads to message collisions. To overcome these message collisions, we set $\sigma > 0$, but its maximum value depends on the lower bound of $||SETW||$ (refer to Eq. 10) to take advantage of the SETW's *refractory period* for achieving one step convergence. We derive the maximum value (σ_{max}) or upper bound of σ for the given T and ε by considering the Eq. 13. From Eq. 7, we can see that $g(\phi'(t)) = \sigma \times \phi'(t)$. The maximum possible value of $\phi'(t)$ can be $(1-\varepsilon)$, also shown in Fig. 3.

$$\varepsilon \geq \sigma_{max}(1-\varepsilon) + \frac{2\nu}{T} \tag{16}$$

[3]Recall that $\varepsilon \in (0, 0.5)$ represents the phase interval

$$\sigma_{max} \leq \frac{\varepsilon - \frac{2\nu}{T}}{(1-\varepsilon)} \quad (17)$$

Hence, $\sigma_{max} > \sigma \geq 0$ for given T and ε.

3.3.3 *Effects of Synchronicity Threshold*

The synchronicity threshold, $S_{Th}(\%)$, is used in the phase transition decision, that is, when a node decides to switch from the *initialisation phase* to the *duty-cycle phase*. Theoretically, a node switches to the *duty-cycle phase* from the *initialisation phase* only when it can hear broadcast messages from all its 1-hop neighbourhood during its SETW. But due to the presence of wireless interference, asymmetrical and temporal wireless links, it might possible that a node will not able to hear all its 1-hop neighbourhood in its SETW. Hence, the *Synchronicity Threshold* defines the percentage of 1-hop neighbourhood nodes that a given node *should* hear to switch phases.

4. EBS IMPLEMENTATION DETAILS

We implemented the EBS scheme under the UPMA [16] framework in TinyOS 2.x for the CC2420 radio, compliant to *IEEE 802.15.4* standards with a data rate of 250 kbps. The UPMA framework is built on the CSMA MAC and we chose to couple the EBS components to the MAC layer instead of the application layer to avoid the buffer delays of the intermediate stack queues and buffers. We implemented two versions of the EBS scheme to understand its behaviour. The two implementations are: (a) EBS without Reach-back to enable stand-alone synchronisation support in the absence of upper-layer broadcast messages, and (b) EBS with partial Reach-back in which the broadcast messages those are generated from an application or routing layer are piggybacked with synch-byte to use them as EBS broadcast messages.

4.1 EBS without Reach-back

In this implementation of the EBS scheme, when a node receives the broadcast message from other nodes, for example, messages received at time t_2 and t_3 in Fig. 7, it advances its phase according to EBS phase advancement function (refer to Eq. 8). The node's EBS layer transmits broadcast messages every time when $\phi'(t) = 0$ (refer to Eq. 8); shown as small circle and outgoing arrow at t_2 and t_3 in Fig. 7. It means, if there is any pending broadcast message from the upper layer at $\phi'(t) = 0$, EBS piggy-backs this message with synch-byte and transmits it as EBS broadcast message. If there is no pending message at the time when $\phi'(t) = 0$, then EBS layer transmits its own message. The advantage of this scheme is that nodes converge quickly in the *initialisation phase* by exchanging messages without the need of *reach-back*. However, this scheme has a high cost when the overshooting condition occurs. Here the nodes keep chasing each other's broadcast messages without achieving stable synchronisation. As a result, the whole network becomes congested with EBS broadcast messages, and in presence of a high $S_{Th}(\%)$, the network may never converge.

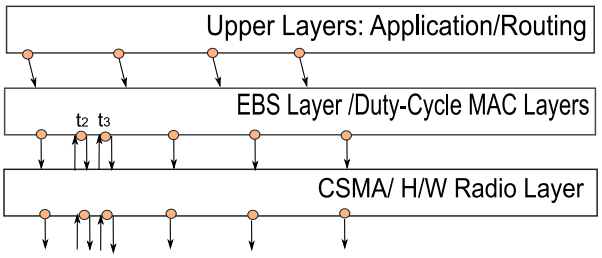


Figure 7: EBS without Reach-back, here when a node receives a EBS brodcast message from its neighbour at time t_2 and t_3, it advances its phase and transmits its broadcast message accordingly without waiting for broadcast message from upper layers.

4.2 EBS with Partial Reach-back

The EBS scheme is implemented at the MAC layer (see Fig. 8); the EBS layer receives a broadcast message from the application or routing layer and transmits it after piggy-backing as EBS broadcast message. To

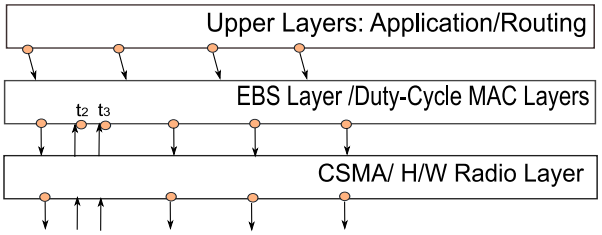


Figure 8: EBS with Partial Reach-back, here when a node receives a EBS brodcast message from its neighbour at time t_2 and t_3, it advances its phase and doesn't transmit its broadcast message at the end of the phase if there is no message from upper layers.

establish coordinated SETWs during the *initialisation phase*, a node advances its phase a few times according to Eq. 8 as shown in Fig. 8 at t_2 and t_3, but it transmits broadcast messages only when it has a new pending broadcast message from the upper layers. The advantage of this scheme is that it avoids the conditions that occur due to the overshooting shown in Fig. 6. However, this leads to nodes flapping between synchronised and non-synchronised states in both the *initialisa-*

tion and the *duty-cycled phases*, shown experimentally in Fig. 9-a. In Fig. 9-a, $T = 10$s and we set fixed $\varepsilon = 0.01$ that is smaller than the ε_{opt} derived from Eq. 15, and it pushes a node with a large connectivity to flap between synchronised and non-synchronised states. That is, when a node is in its synchronised state, it switches to *duty-cycle phase*, however, in non-synchronised state it is 100% duty-cycled. Further, when we increase the $\varepsilon = 0.05$, and then we see that that flapping between synchronised and non-synchronised states reduce significantly as shown in Fig. 9-b.

4.3 Neighbourhood Size

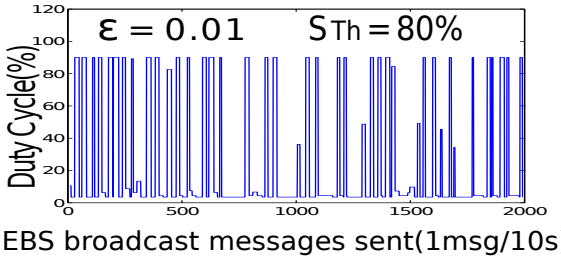
To include the synchronicity concept into the EBS implementation, EBS requires information regarding the neighbourhood size to calculate a node's S_{Th}. The neighbourhood size can be calculated at the start by keeping all nodes awake for initial few periods T, thereafter using this value during the EBS run time. However, to make the EBS scheme more agile and dynamic, we further suggest to keep updating this value during the EBS *duty-cycled phase*. Note that, EBS is not maintaining its own neighbourhood table but only uses the neighbourhood size that is calculated on the basis of number of broadcast messages received in single broadcast message interval, T, in the *initialisation phase*. However, the routing layer's link estimation mechanisms can be used to make precise and accurate neighbourhood size estimation.

5. EXPERIMENTS SETUP

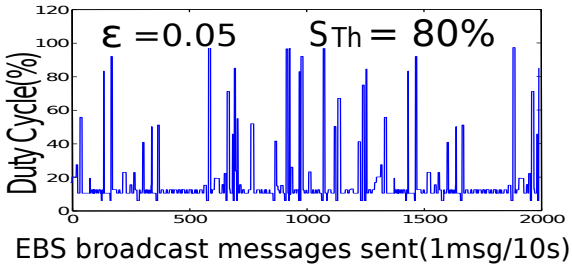
We perform our experiments on two different size of testbeds: Motelab (87 Telosb nodes) [17], and our local lab (10 MicaZ nodes). Telosb and MicaZ are small embedded devices with low-power Zigbee radios [2]. We begin our experiments by fixing the value of the broadcast time interval, T and varying the $S_{Th}(\%)$, and measured duty-cycle of the network after every second interval. We present in this paper the results of the experiments that are executed on Motelab with a broadcast interval time (T) set at 10 seconds, 20 seconds, and 30 seconds (representing typical routing control messages interval). Recall the SETW width $W = 2 \times \varepsilon \times T$, ε_{opt} varies according to the Eq. 15 for the given experiments. For each run (either varying T or S_{Th}), we measured the duty-cycle of each node, their wake-up times, as a percentage. In the resulting graphs, we present the average duty-cycle and average throughput of the network by varying the $S_{Th}(\%)$ from 95% to 20%.

6. RESULTS

In this section, we present EBS performance results showing both convergence parameters effects and comparative performance with the nearest state-of-the-art protocol.



(a) The node keep flapping between synchronised and non-synchronised states when ε is set smaller than ε_{opt}



(b) The node's flapping improved with increasing the ε to 0.05 from 0.01

Figure 9: Individual behaviour of a node, which has average connectivity 20 in Motelab Testbed, where all nodes running EBS with reach-back

6.1 Calculation of Deterministic Propagation Delay

From Eq. 15, it is clear that the value of ε_{opt} is proportional to propagation and processing delays (C). However, these are non-deterministic in the real environment. But, for EBS to converge in the *initialisation phase*, we do require the value of these delays. To get these values, we first performed our initial experiments to observe the behaviour of the EBS with prefixed ε varying from 0.01 to 0.1. To observe the flapping behaviour, we chose EBS with the partial *reach-back* implementation that is described earlier in Section 4.2. We found that nodes flapping with low value of ε was more prominent as compared to large values, shown in Fig. 9. We represent Eq. 15 in terms of C as:

$$\varepsilon_{opt} \geq \frac{C}{T} \tag{18}$$

We run initial experiments to derive the value of C by varying ε and T and found that it should be $C \geq 50$ms for the network, which has an average connectivity of 2-3, to converge. The adaptive value C_i of a node i is based on density and other factors that can be further

explored. However, for the experiments in this paper we choose EBS (without *reach-back* implementation) with adaptive C_i given as follows:

$$C_i = (50 \times d_i \times S_{Th}(\%)/100)ms \quad (19)$$

Note that the initial experiments shown in Fig. 9, are performed with the consideration that $\sigma_{max} \leq \frac{\varepsilon}{(1-\varepsilon)}$ by assuming $2\nu/T = 0$ in Eq. 17.

6.2 Effects of $S_{Th}(\%)$

We study the effect of $S_{Th}(\%)$, that is, the degree of synchronicity on the convergence in the *initialisation phase* and its subsequent effect on the broadcast message throughput in the network. For these experiments, we set $T = 30s$ and vary S_{Th} from 20 % to 95%, the results are shown in Fig. 10.

Fig. 10-(a) shows the Motelab topology on which we run our experiments; we present the duty-cycle of a node A with different values of $S_{Th}(\%)$ w.r.t the running time of EBS in Fig. 10-(b)A. For this experiment, we use the fixed value of $C = 50$ ms and each node calculates its neighbourhood during only one initial period T. We can see that in the initial 5 mins, the duty-cycle of the node is nearly 100% when $S_{Th} = 95\%$ as compared to other lower values of S_{Th}. This clearly shows that, to achieve synchronicity with high threshold the node must remain awake slightly longer (here nearly $10T$) as compared to T or $2T$ with low threshold in the *initialisation phase*. Also note that, in this set of experiments, we perform neighbourhood size calculation in only first T period and we observed that due to lossy wireless links this value might not be the actual neighbourhood size. Due to this reason, for our next experiments, we selected an average of neighbours by keeping nodes awake 100% for the first $5T$.

Fig. 10(b) is composed of two figures (B and C), showing the comparative duty-cycle(%) and throughput(%) achieved respectively. The Fig. 10(b)-B presents an average duty-cycle with a fixed $C = 50$ms for all nodes, node A's average duty cycle with a fixed $C = 50$ms, and average duty-cycle with variable C which is calculated by each node using Eq.19 of all nodes in the given topology. We found that the adaptive C shows a regular pattern in terms of both duty-cycle and throughput results. Also, adaptive C improves the throughput significantly by 40% at small cost of duty-cycle(2-3%) as shown in Fig. 10(b)-C.

6.3 Effects of Broadcast Message Interval, T

Fig. 11 presents the effects of T on duty-cycle(%) and throughput(%). We use three values of $T = 10$ s, 20 s, 30 s and varied S_{Th} from 20% to 95%. We found that duty-cycle increases nearly 0.1 % with every 10% increase in S_{Th} till $S_{Th} = 60$ % for all values of T. However, for $S_{Th} > 60\%$, its incremental effect on duty-cycle is slightly higher(1-3%) for low values of T as compared to high values of T. On the other hand, throughput is not much affected with T, higher values of T perform better for low S_{Th} and lower values of T perform better for high S_{Th}. However, there is gradual increase in throughput from 20-30% to 80-90% when $S_{Th}(\%)$ varies from 20% to 95%, respectively.

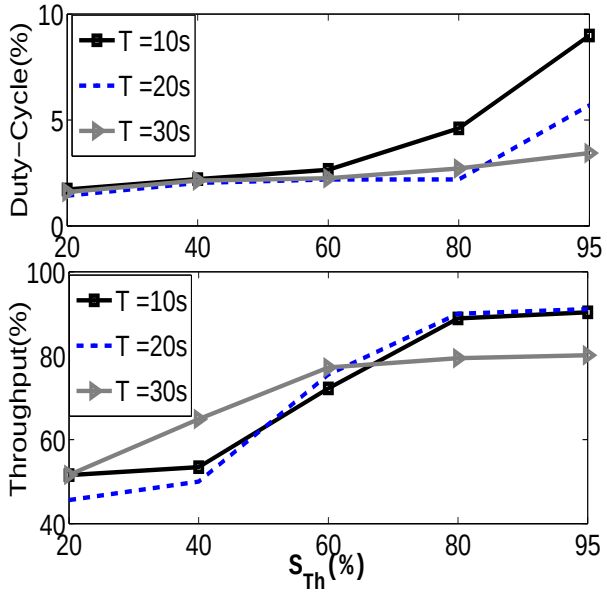


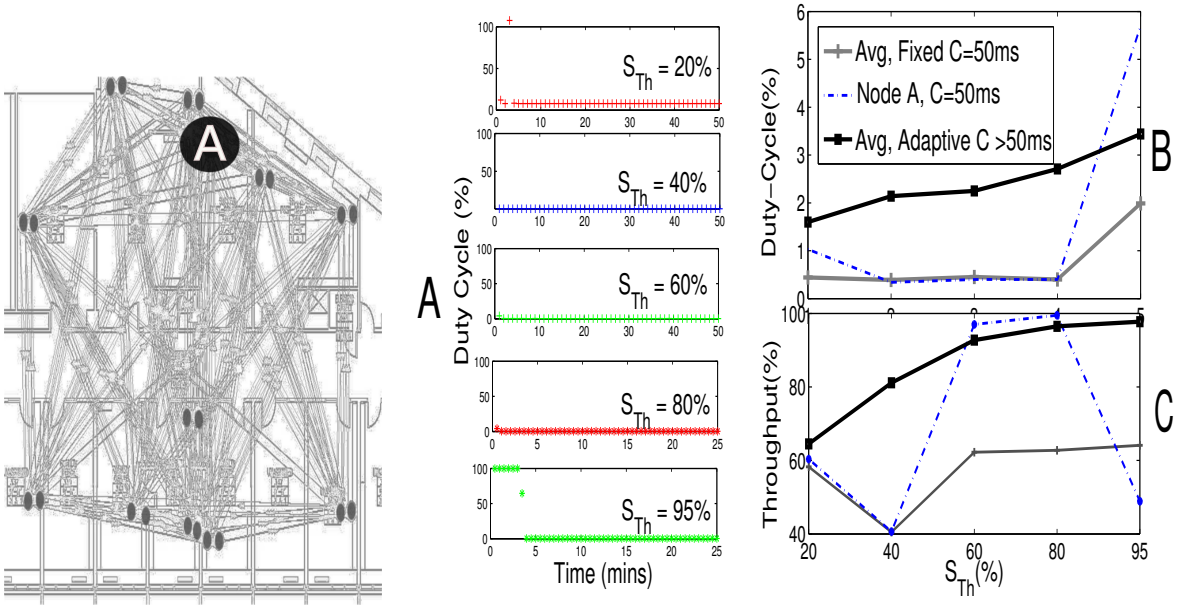
Figure 11: EBS Duty-cycle and throughput with adaptive C.

6.4 Effects of σ

If we choose $\varepsilon = \varepsilon_{opt} = \frac{C}{T}$ then $\sigma = \sigma_{max} = 0$, otherwise we consider Eq. 17. However, in this equation value of $2\nu/T$ is unknown. So, for simplification, we modify it as:

$$\sigma_{max} \leq \frac{\varepsilon - \frac{C}{T}}{(1-\varepsilon)} \quad (20)$$

Here, $\varepsilon > \frac{C}{T}$ and $C = 50$ms. We performed experiments by varying ε by setting it to $\frac{C+5ms}{T}$, $\frac{C+10ms}{T}$, and $\frac{C+15ms}{T}$ to observe the density effects. However, we do not observe much effect of this on throughput for low density network, but has significant throughput and convergence improvement for dense network. To include the density into consideration, instead of fixed C, adaptive C is considered (refer to Eq. 19) with the assumption that σ is included in the adaptive C. The results showing improvement in throughput by using adaptive C over fixed C are shown in Fig. 10(b)-C. The σ plays important role when C is very precisely calcu-



(a) (b)

Figure 10: Fig. (a) shows the nodes with average connectivity 7-8 on Motelab testbed topology; in Fig. (b)-A, we study the behaviour of individual node A in the given topology, where all nodes run EBS, with respect to time with different Synchronicity thresholds(S_{Th}(%)). In Fig. (b)-B and Fig. (b)-C, we analyse the average duty-cycle and average throughput percentages of all nodes when $C = 50$ms and when C is adaptive according to Eq. 19.

lated, but for our current experiments we have used the simplified version of adaptive C (refer to Eq. 19), that is large enough for avoiding the contention. Therefore, the precise value of σ can be further explored along with optimised value of C.

6.5 Comparative performance

We compare our result with *Refractory Periods*[11] because it is the closest of the bio-inspired approaches to ours. To make it suitable for duty-cycling, we use the $T_{Ref} = T/2$, as suggested in [6] for achieving quick loose synchronicity, we call this Modified Refractory Period (MRF). The phase advancement function of MRF is represented as:

$$\phi(t^+) = \begin{cases} \phi(t) & \text{if } \phi(t) < 0.5 \\ 1 & \text{if } \phi(t) > 0.5 \,\&\&\, \phi(t) < 1 \end{cases} \quad (21)$$

We show that EBS achieves almost similar throughput, but with large improvement in duty-cycle as shown in Fig. 12. From Fig. 11, it is clear that for our current network topology, setting $S_{Th} = 80\%$ achieves more than 85% throughput while keeping duty-cycle below 5%, which means saving nearly 80% energy by avoiding ideal listening. For comparing our results with MRF, we choose $S_{Th} = 80\%$. In Fig. 12, we use adaptive $C > 50$ms, however, if we put $C > 100$ms, we can achieve same or higher throughput as compared to MRF. Therefore, in this experiment we present the throughput achieved at minimum duty-cycle for adaptive $C > 50$ms which EBS can achieve while considering the propagation delays and processing delays which are caused by the lower layers (e.g. CSMA).

7. CONCLUSION AND FUTURE WORK

In this paper we present EBS, a fully decentralised scheme for wireless embedded systems that ensures network-wide data broadcast in duty-cycled networks without requiring costly global synchronisation. We have identified the challenges of providing such a mechanism, formalised the constraints therein and produced algorithms that are lightweight and robust to changing network topology or failures.

As far as we are aware, we are the first to provide support for broadcast messaging that uses local synchronisation within a duty-cycled network. Further, EBS has been fully implemented on a wireless sensor devices testbed to show both convergence parameters effects and comparative performance with the nearest state-of-the-art protocol (MRF). Here, we show that EBS achieves more than 80% throughput while keep-

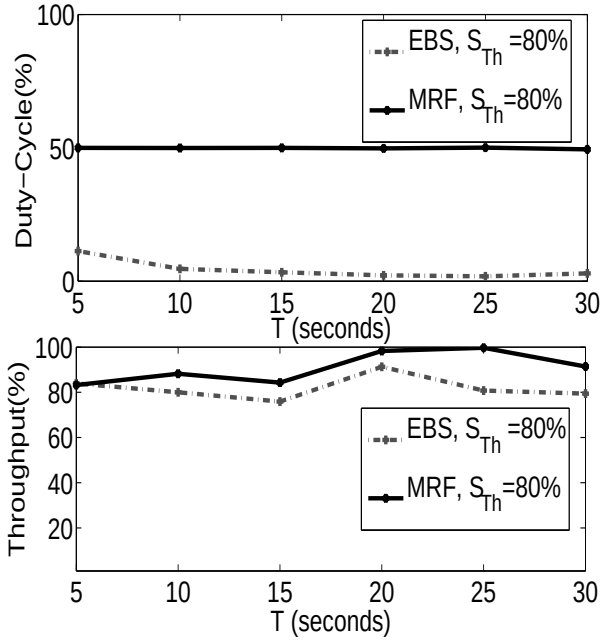


Figure 12: Comparative Duty-cycle and throughput of EBS(with adaptive C) and MRF.

ing the duty-cycle down to less than 5% (for broadcast message intervals, T >10s). Therefore, the results confirm that EBS can match the throughput of comparative schemes that are continuously awake (100% duty-cycle) at a considerable reduction of energy consumption network-wide. To achieve, this we present the notion of a *synchronicity threshold*, which allows us to trade-off the maximum throughput achievable versus the degree of duty-cycling (energy consumption) the system can tolerate. Further, we have shown that not only does this scheme converge to a steady state very quickly in the presence of deterministic delays, but is also tolerant to random delays even while in its *duty-cycled phase*.

Nevertheless, this scheme can be further improved. Through further analysis of propagation and processing delays, we can produce a better model of C, thus improving convergence times in the presence of random propagation and transmission delays in the *initialisation phase*. This improvement would allow the use of heterogeneous self-adaptive *synchronicity thresholds*, S_{th} (%), which are more appropriate to a given node's degree and network conditions. Therefore, this precise modelling of C further reduces the network-wide energy consumption in the presence of even large numbers of node failures.

8. REFERENCES

[1] *SunSPOT Datasheet. http://www.sunspotworld.com/.* Accessed on 19/09/08.

[2] *Micaz datasheet. http://www.xbow.com/Products/Product.* Accessed on 21/09/08.

[3] *BTnode datasheet. http://www.btnode.ethz.ch/pub/uploads/Main/.* Accessed on 17/08/08.

[4] A. El-Hoiydi and J.-D. Decotignie, "Wisemac: an ultra low power mac protocol for the downlink of infrastructure wireless sensor networks," in *ISCC '04: Proceedings of the Ninth International Symposium on Computers and Communications 2004 Volume 2 (ISCC"04)*, (Washington, DC, USA), pp. 244–251, IEEE Computer Society, 2004.

[5] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 39–49, ACM, 2004.

[6] A. Tyrrell, G. Auer, and C. Bettstetter, "Emergent slot synchronization in wireless networks," vol. 9, (Piscataway, NJ, USA), pp. 719–732, IEEE Educational Activities Department, 2010.

[7] P. Sommer and R. Wattenhofer, "Gradient clock synchronization in wireless sensor networks," in *IPSN '09: Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*, (Washington, DC, USA), pp. 37–48, IEEE Computer Society, 2009.

[8] G. Werner-Allen, G. Tewari, A. Patel, M. Welsh, and R. Nagpal, "Firefly-inspired sensor network synchronicity with realistic radio effects," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 142–153, ACM, 2005.

[9] A. Tyrrell, G. Auer, B. C, and R. Naripella, "How does a faulty node disturb decentralized slot synchronization over wireless networks?," in *IEEE Intern. Conf. on Communications (ICC)*, 2010.

[10] R. Mirollo and S. Strogatz, "Synchronization of pulse-coupled biological oscillators," *SIAM Journal on Applied Mathematics*, vol. 50, pp. 1645–1662, 1990.

[11] J. Degesys, P. Basu, and J. Redi, "Synchronization of strongly pulse-coupled oscillators with refractory periods and random medium access," in *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, (New York, NY, USA), pp. 1976–1980, ACM, 2008.

[12] J. Klinglmayr, C. Bettstetter, and M. Timme, "Globally stable synchronization by inhibitory pulse coupling,"

[13] L. Cui and H. Wang, "Reachback firefly synchronicity with late sensitivity window in wireless sensor networks," in *HIS '09: Proceedings of the 2009 Ninth International Conference on Hybrid Intelligent Systems*, (Washington, DC, USA), pp. 451–456, IEEE Computer Society, 2009.

[14] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *IEEE Infocom*, June 2002.

[15] W. Ye, F. Silva, and J. Heidemann, "Ultra-low duty cycle mac with scheduled channel polling," in *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 321–334, ACM, 2006.

[16] K. Klues, G. Hackmann, O. Chipara, and C. Lu, "A component-based architecture for power-efficient media access control in wireless sensor networks," in *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 59–72, ACM, 2007.

[17] *Harward Sensor Network Testbed: http://motelab.eecs.harvard.edu/.*