

# An Energy Efficient and Accurate Slot Synchronization Scheme for Wireless Sensor Networks

Lillian Dai      Prithwish Basu      Jason Redi  
BBN Technologies, 10 Moulton St., Cambridge, MA 02138  
ldai@bbn.com      pbasu@bbn.com      redi@bbn.com

**Abstract**—Existing slotted channel access schemes in wireless networks assume that slot boundaries at all nodes are synchronized. In practice, relative clock drifts among nodes cause slot misalignment over time and can result in catastrophic data loss in such systems. We propose a simple network-wide slot synchronization scheme (Slot-Sync and Slot-Resync) suitable for duty-cycling wireless sensor networks. The proposed scheme attains high accuracy by circumventing dominant sources of error inherent in traditional time synchronization protocols. At moderate duty-cycle frequencies, the proposed scheme also has the unique advantage of eliminating re-synchronization (re-sync) overhead completely, thereby achieving slot re-sync essentially for free. We provide an energy efficient slot guard time and re-sync interval design for the proposed scheme and analyze several spanning tree structures for slot-alignment message propagation. In addition, we derive upper bounds on the synchronization (sync) error for a family of trees. Through simulations, we compare the spanning trees we propose to those used for time sync in literature and show up to 80% reduction in sync error and up to 70% reduction in energy needed for slot-alignment message propagation by choosing appropriate tree structures.

**Keywords**—wireless sensor network, slot synchronization, guard time, spanning tree, synchronization error, clock drift.

## I. INTRODUCTION

In recent years, there has been phenomenal growth in the development and deployment of battery operated wireless sensor networks (WSN). Since battery energy is a limited resource, many systems intelligently conserve energy by turning the radio transceivers off through a process termed *duty-cycling* [1-3]. Both asynchronous [2] and synchronous [1, 3] medium access control (MAC) schemes have been proposed for such WSNs. In general, synchronous MAC with time slotting and some amount of transmission coordination can achieve higher throughput than asynchronous schemes; hence are more suitable for fixed sensor networks with moderate amount of traffic load [2-3]. However, implicit in all synchronous MAC schemes is the assumption that slots of neighboring nodes are aligned within a reasonable level of precision at all times. In practice, clocks in different nodes suffer from random drifts, causing slot misalignment over time. This, in turn, may result in catastrophic data loss in duty-cycling WSNs as nodes are relying on the underlying slotted structure for higher layer processes.

---

Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

In this paper, we investigate the slot synchronization (sync) problem in multi-hop WSNs that employ synchronous MAC. One way to achieve slot sync is to sync the clocks (*time-sync*) as proposed in [4-8]. By virtue of having attained a common time within a certain degree of accuracy, slots at neighboring node pairs are temporally synchronized with bounded offsets. There are various sources of error associated with time-sync protocols, which stem, to a large part, from variable software delays for transmission and reception of time-stamped messages [5, 6]. We note that many slotted networks do not have an explicit requirement for attaining a common time in different nodes [3]. Instead, it is only necessary that the slot boundaries in different nodes are synced and that nodes agree on which slot they are currently transmitting or receiving in. This affords an opportunity to significantly reduce the sync errors inherent in time-sync protocols due to time-stamping.

We propose a slot sync scheme (termed *Slot-Sync and Slot-Resync*) that does not exchange time-stamped information, hence eliminating variable software delays associated with channel access, packet queueing, transmission, and reception. Instead of using time-stamps, our scheme (described in detail in Section III) relies on the inherent property of the slotted MAC structure and an added feature that determines the signal arrival time at each receiver. Since each node transmits information at the beginning of a time slot, each data transmission informs other nodes of a node's slot boundaries. Hence, re-sync is achieved for free each time a data transmission occurs. To achieve initial sync in the network, *slot-align messages* are propagated in the network along spanning trees starting from a reference node. In effect, with the proposed Slot-Sync scheme, clocks in two neighboring nodes may still report different times; however, the nodes agree on slot boundaries and can proceed to exchange information in a common slot.

We perform detailed analytical and simulations studies of the proposed scheme. Our key contributions are:

1. Slot-Sync and Slot-Resync that can achieve greater accuracy and energy savings compared to time-sync.
2. Energy-efficient design for guard time and re-sync period.
3. A message-efficient slot formation and maintenance scheme that uses an appropriately chosen rooted tree structure for slot-align message propagation.
4. An upper bound on the maximum sync error.
5. Simulation results that show up to 80% improvement in sync error and up to 70% improvement in energy efficiency for slot-align message propagation of certain minimum-hop based trees over general level-based trees such as the rapidly constructed tree proposed in [5].

Note that contributions (2-5) are also relevant for time-sync schemes as applied to slotted wireless systems. In that sense, our guard time and tree analyses are of broader interests.

This paper is organized as follows. Section II provides an overview of related work. Section III describes the slot sync problem. Section IV provides analysis of the energy tradeoffs between guard times, re-sync periods, and slot utilization. Section V focuses on the problem of achieving network-wide slot sync by propagating slot-align messages along spanning trees. Section VI presents a comparative simulation analysis of various sync trees. Section VII concludes the paper and outlines possible research directions.

## II. RELATED WORK

The slot sync problem has been studied in the context of special wireless networks such as Bluetooth. In a Bluetooth piconet (with a maximum of 8 nodes), a master node provides the time slot reference [9]. A slave node responds to a master's "poll" in the second half of the received slot, thus achieving slot sync. Adjacent frequency hopping piconets are interconnected by bridge nodes which could play master/slave or slave/slave roles. Since Bluetooth performs slot sync within a piconet alone, such bridges often have to switch between piconet time references (and thus waste many slots) during data forwarding. In fact, efficient scheduling of such slots and coordinating transmission at these bridge nodes is a hard problem and a topic of intense research [9].

Ebner *et al.* also propose a scheme which syncs the slots at each node to those of a nearest neighbor node, forming locally synced node pairs on demand [10]. However, their dynamic slot sync architecture is designed for short lived communications for highly mobile vehicular systems, not for general purpose or duty cycling WSNs in which nodes rely on the synced slot structure to sleep and wake up.

Realizing the limitations of the classical Network Time Protocol (NTP) [11] from the perspective of energy efficiency, Elson *et al.* proposed Reference Broadcast Synchronization (RBS) for time-sync [4]. In RBS, nodes send beacons to neighbors using wireless broadcasts. These beacons do not contain explicit timestamps; instead, receivers use the arrival time of a beacon as reference for comparing their clocks. RBS achieves reasonable sync accuracy at the expense of excessive message exchanges between receivers.

Ganeriwal *et al.* [5] proposed a network-wide time-sync protocol for WSNs called Timing-sync Protocol for Sensor Networks (TPSN). TPSN has two phases: level discovery and time sync. The level discovery phase assumes that there exists a known root node in the network. Such a node is assigned a level 0 and the level discovery protocol begins at this node. The level-based sync tree used in this work has the advantage of rapid tree construction at the expense of large sync error as well as long delay and high energy consumption in propagating the sync messages. We believe that since the sync tree would be used often for re-sync, one with better properties is more desirable for achieving significant reduction in sync-error and improvement in message (energy) efficiency.

Some researchers proposed schemes that assume each clock can be approximated by an oscillator with fixed frequency [12]. They derived upper and lower bounds for constraints on the relative drift and relative offset between node pairs. Others proposed schemes that aim at minimizing the complexity of achieving time-sync instead of minimizing sync error [7]. Li and Rus proposed a localized diffusion-based method in which nodes exchange and update clock reading information locally with neighbors [8]. Their algorithm can adapt to node failure, mobility or channel errors, but takes much longer time to converge than tree based sync protocols.

## III. SLOT SYNCHRONIZATION FUNDAMENTALS

Slotted networks require two fundamental primitives for effective operation: (a) accurate alignment of slot boundaries between neighboring nodes, and (b) coordinated slot usage between nodes in the network for collision free operation, and/or duty cycling. While a failure to perform (b) effectively can result in loss of network throughput due to collisions and/or dropped packets, a failure to perform (a) accurately can result in catastrophic data loss over a period of time. In this paper, we focus exclusively on (a).

Two nodes X and Y can align their slot boundaries in several ways. If both have GPS receivers, their clocks are synchronized to within 200ns of UTC. In the absence of GPS, X and Y can run a time-sync protocol [4,5,8,12] to synchronize their clocks. Slot sync is obtained simply by virtue of two nodes having obtained a common time. If time-sync schemes are used, the misalignment of slot boundaries at X and Y is directly proportional to the error in time-sync. Most time-sync protocols in the literature exchange messages containing time-stamped information generated in a higher layer of the protocol stack to estimate relative offset, drift and propagation delay. The dominant source of error in these protocols is the variability in the times spent by this time-stamped message in the sender's MAC and the receiver's OS. These delays are generally much greater than the signal propagation delay [5].

The availability and use of physical layer timestamps at the receiver node can help to eliminate the delay in the receiver's OS. However, most time-sync schemes with the exception of RBS [4] still suffer from the "access time" delay on the sender's side. RBS, as proposed, performs no time stamping at the sender and will only suffer from the error due to receiver OS delay. If RBS is augmented with the capability of physical layer time-stamping at the receiver, the most dominant source of error that would remain is propagation delay. Our scheme is similar to RBS augmented with PHY layer time-stamping and without the exchange of receiver timestamps between nodes. There are several advantages of doing this: (a) the only major source of error is propagation delay; (b) no exchange of sender (as in [5]) or receiver (as in [4]) timestamps is necessary as slot sync can be achieved in one shot in a neighborhood; and (c) no exchange of timestamps is necessary for re-sync at future times.

Our proposed mechanism achieves slot sync between X and Y without the intermediate step of explicit time-sync. Assume that a periodic slot structure is maintained by the physical layer module (PHY). At local time  $t$ , a “slot-align” message is generated by a higher layer module at X and contains the slot number the message should be sent in. This message is transmitted by the PHY in the appropriate slot at time  $t'$ . The slot-align message reaches Y at its local time  $t''$ . Y then begins its PHY layer time slotting starting at time  $t''$ . At this point X and Y have achieved slot sync without any further information exchange. This scheme is shown in Fig. 1. The only error incurred in this process is the propagation delay. Due to the variable time delay from the time the message is generated to the transmission time, the slot number contained in the packet may be expired at time  $t'$ . If this occurs, PHY notifies the upper layer processes so that a new slot-align message can be generated with more lead time.

Starting from a reference node, slot-align messages are propagated through the network by nodes that have received the slot-align message previously along the edges of a sync-tree (discussed in greater detail in Section V). Each node maintains a fixed slot duration and rebroadcasts the message at its slot boundary, i.e., after waiting for time equal to an integer multiple of slot durations. Upon receiving a message from a designated parent in the tree, a child node whose slot boundary has drifted relative to the parent’s, re-syncs its slot to the time of message reception. This process is illustrated in Fig. 2.

Due to imperfections in manufacturing processes, changes in ambient temperatures, material aging, and other external conditions [13], a typical clock used in sensor nodes may drift up to tens of parts per million (ppm) which is equal to a few seconds per day. Since communication equipments typically operate on the orders of microseconds, clock drifts can significantly degrade the performance of synchronous communications. Hence resync needs to be an integral part of any slotted network system. To protect against data loss before slots are re-synced, small amounts of guard times are usually added before and after the data portion of the slot. If there is enough traffic flowing through the network, re-sync can be achieved for free under our proposed scheme. Otherwise, explicit re-sync messages are transmitted (See Section IV).

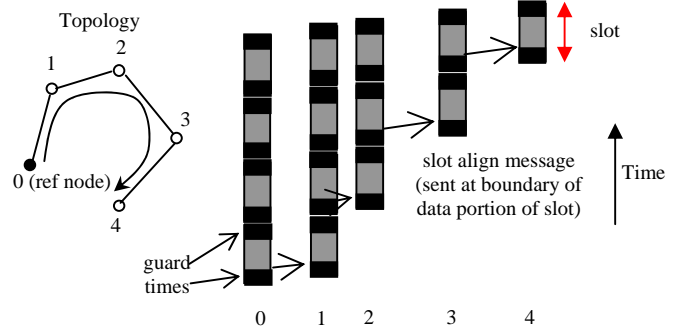


Fig. 2 Slot-sync Message Propagation and Alignment

#### IV. ENERGY EFFICIENT SLOT DESIGN

In a wireless network, it is unnecessary to align the slot boundaries of non-neighboring nodes since they do not participate in data exchange. For each neighboring node pair ( $tx/rx$  pair), data loss may occur due to relative clock drift, signal propagation delay, and initial slot offset (*sync error*) as shown in Fig. 3. All the pertinent parameters are defined in Table I. To mitigate data loss, guard times need to be provisioned in each slot as shown in Fig. 3. Although the amount of time available for data transmission is reduced in each slot by provisioning guard times, data loss due to slot misalignment can be eliminated for a number of slots. As long as slots of neighboring nodes are re-synced before data loss occurs (which may not incur any overhead as discussed in Section III.B), data loss due to slot misalignment can be eliminated completely. The network is said to have achieved network-wide sync (*net-sync*) if the slot boundaries between every neighboring node pair are synced in every slot such that data loss is eliminated.

First, we would like to design the guard times such that net-sync is achieved with minimum re-sync frequency, while maintaining a *desired slot utilization*,  $U$ , which measures the fraction of time available for data transmission in each slot. This can be posed as a constrained optimization problem. We provide a simple rule-of-thumb result in scenarios where the maximum clock drift  $\theta_{max}$  and the maximum sync error  $\Delta$  are small. More details are presented in [14]. In this case,

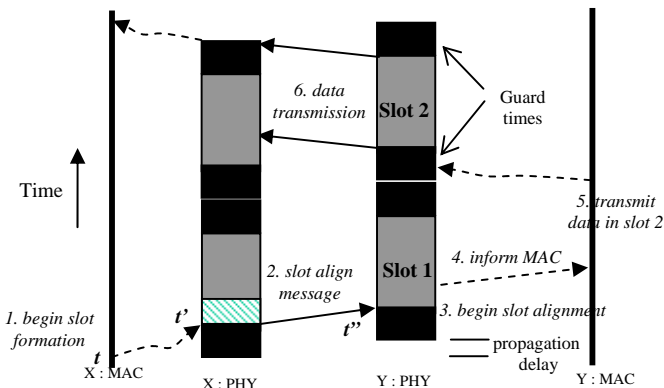


Fig. 1 Pair-wise Slot Formation and Slotted Data Transmission

TABLE I. LIST OF PARAMETERS FOR SLOT DESIGN

$\psi$	Slot duration [sec]
$\theta_{AB}$	Relative clock drift between nodes A and B
$d_{AB}$	Propagation delay between nodes A and B [sec]
$\Delta_{AB}$	Initial slot offset (sync error) [sec]
$s$	Data slot duration [sec]
$g_{init}$	Initial guard time [sec]
$g_{end}$	End guard time [sec]
$T_{max}$	Max number of slots before data loss occurs (re-sync period)
$\hat{T}$	Re-sync period in [sec] = $T_{max}\psi$
$U$	Slot utilization = $s/\psi$
$\theta_{max}$	Maximum clock drift relative to ideal clock [ppm]
$\Delta$	Maximum sync error across all tx/rx pairs [sec]
$d$	Maximum propagation delay across all tx/rx pairs [sec]

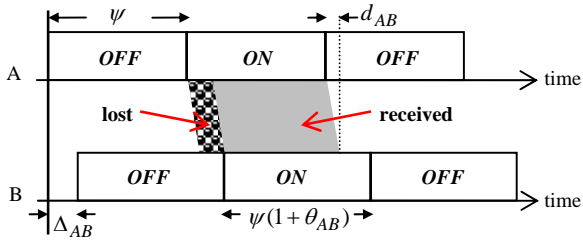


Fig. 3 Transmission from node A to B with loss. B has a slower clock than A ( $\theta_{AB} > 0$ ). All times shown are times reported by the clock in node A. (Propagation delay and sync error are exaggerated for illustration purposes).

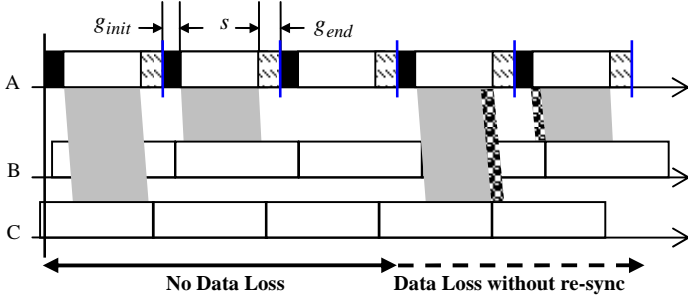


Fig. 4 Transmissions from A to B and A to C where, compared to A, B has a slower clock and C has a faster clock. After a number of lossless transmission slots, relative clock drifts induce data loss

$$g_{init} \approx \frac{s}{2}(U^{-1} - 1) - d, \quad g_{end} \approx \frac{s}{2}(U^{-1} - 1) + d, \quad T_{max} \approx \frac{1-U}{4\theta_{max}} \quad (1)$$

In terms of re-sync interval  $\hat{T}$  in time, a simple (appx) expression can be obtained for the total guard time in a slot

$$g_{init} + g_{end} \approx 4\theta_{max}\hat{T} \quad (2)$$

This result is fairly intuitive since for small  $\theta_{max}$  and  $\Delta$ , slot boundaries drift apart by at most  $2\theta_{max}\hat{T}$  in duration  $\hat{T}$ . A  $g_{init}$  of  $2\theta_{max}\hat{T}$  is needed for transmission from a node with a slow clock to a node with a fast clock. Similarly, a  $g_{end}$  of  $2\theta_{max}\hat{T}$  is needed for transmission from a node with a fast clock to a node with a slow clock. Exact derivations of these quantities have been omitted due to paucity of space and can be found in [14]. Large  $\Delta$  results in larger guard times.

Now we address the issue of optimal slot utilization for energy efficient operation of the Slot-Sync scheme. For a given slot utilization, (2) shows that smaller guard times result in a shorter re-sync period. This implies that less energy is wasted at receivers in each slot where a transmission takes place, at the expense of potentially more energy consumed to perform re-sync. The number of data and re-sync transmissions depend on the traffic rate (in an ideal duty-cycling network, this also corresponds to the duty-cycle frequency). Let the arrival of ON slots be a Poisson process with arrival rate  $\lambda$ . If a data transmission occurs prior to  $\hat{T}$ , then no extra energy need to be expended for re-sync; otherwise, for every  $\hat{T}$  interval that does not have a data transmission,  $m$  number of slots are needed to transmit re-

sync messages. To first order, the expected amount of energy wasted due to guard times and re-sync is proportional to

$$\lambda \left( g_{init} + g_{end} + m\psi \frac{e^{-\lambda\hat{T}}}{1 - e^{-\lambda\hat{T}}} \right) \quad (3)$$

Substituting (1-2) into (3), one can numerically solve for the optimal utilization that minimizes the amount of energy wasted. Fig. 5 shows a typical curve for wasted energy. For higher expected traffic volume, it is more energy-efficient to have a larger slot utilization with smaller guard times and re-sync period.

Note that implicit in the above derivations, we assumed that a node transmits to a receiver in one slot and either turns off or transmits to a different receiver in the subsequent slot. In general, a node can transmit to the same receiver in consecutive slots and need not terminate the transmission in each slot. If pair-wise transmissions typically span several slots, say  $n$  slots, then one may consider redefining a new data slot interval  $s' = n\psi - g_{init} - g_{end}$ . All previous derivations carry through for the expanded data slot. One may argue that the guard time at the end of each slot is unnecessary since a receiver can intelligently turn itself off only after an end-of-transmission signal is received from the transmitter. However, depending on the duty-cycling and MAC protocols, turning a receiver off past its slot boundary may cause collisions in the subsequent slot. In our design, we provide guaranteed slot boundaries to higher layers, although cross-layer design is possible and is subject to future research.

## V. NETWORK-WIDE SLOT SYNCHRONIZATION

In this section, we investigate techniques and propose algorithms to achieve net-sync. Two important network wide metrics are examined:

1. *Maximum sync error* ( $\Delta$ ): maximum slot misalignment between any pair of neighbors after termination of the net-sync algorithm taking across all node pairs in the network.
2. *Message efficiency*: the number of message exchanges needed to achieve net-sync (this directly translates to energy efficiency).

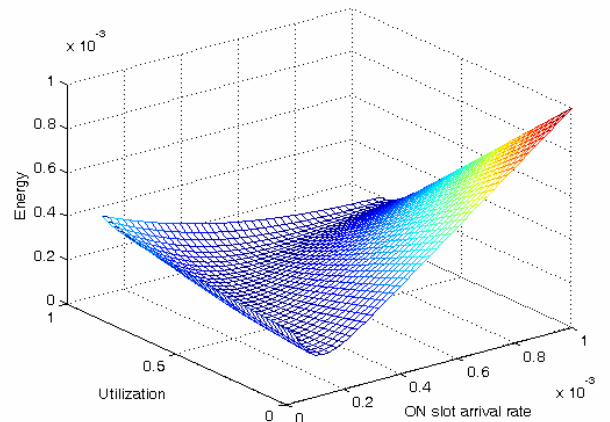


Fig. 5 Wasted Energy for varying ON slot arrival rate  $\lambda$  and slot utilization  $U$

The proposed basic Net-Sync algorithm is as follows:

- One of the nodes in the network provides a reference slot boundary and transmits a *slot-align message* at the beginning of a data slot (after  $g_{init}$ ).
- Neighboring nodes receive the packet after some propagation delay, set their slot boundaries to the time at which the packet is received minus  $g_{init}$ , and retransmit the slot-align message at subsequent data slot boundaries.
- The slot-align message is propagated in the network through multiple hops until every node's slots are aligned.

As can be expected, the choice of the reference node as well as the paths taken by slot-align messages has significant impact on the maximum sync error and message efficiency to achieve net-sync. Although many time-sync papers provide performance analyses on the various time-sync protocols, they assume certain sync message propagation structures without much justification. Furthermore, most error analyses are performed for nodes along a single sync path only.

#### A. Model for Net-Sync and Problem Statement

Consider a sensor network graph  $G=(V,E)$  where  $V$  is a set of nodes and  $E$  is a set of edges representing bidirectional communication links (with maximum transmission range  $\rho$ ). The reference node (or *root node*) is indexed by  $r$ . Function  $d:E\rightarrow\mathbb{R}^+$  denotes the propagation delay along each edge of  $G$ , and is directly proportional to the distance between the endpoints of the edge. Let  $S$  denote a sync structure which is a *spanning subgraph* of  $G$  – the net-sync messages are propagated along the edges of  $S$ , starting at node  $r$ . Clearly, a spanning tree of  $G$  is a minimal structure along whose edges the sync messages can propagate. Let  $P_S^r(w)$  denote the shortest path from  $r$  to  $w$  using edges in  $S$  in terms of the following total distance metric:

$$D_S^r(w) = \sum_{(x,y)\in P_S^r(w)} d(x,y) \quad (4)$$

This is the minimum delay path from  $r$  to  $w$  using edges in  $S$ . The sync error between two neighboring nodes,  $u$  and  $v$  is the difference between the total delays accumulated by the respective slot-align messages when they reach  $u$  and  $v$  from  $r$  along best paths in the sync structure (assuming no delay in forwarding at intermediate nodes). It is also desirable that message propagation along  $S$  should require the minimum number of broadcast transmissions  $b(S)$  for reaching all nodes in  $G$  starting at root node  $r$ . This message efficiency metric leads to the conservation of energy for performing net-sync.

The decision version of the corresponding constrained optimization problem that minimizes the two aforementioned metrics can be formulated as follows:

**Input:**  $G=(V,E)$ ;  $d:E\rightarrow\mathbb{R}^+$ ;  $B\in\mathbb{Z}^+$

**Desired outputs:** spanning tree  $T$ ; reference root node  $r$

**Objective:**  $\min_T \max_{(u,v)\in E} |D_T^r(u) - D_T^r(v)|$

**Constraint:**  $b(T)\leq B$

Clearly, this is a very complex optimization problem that is NP hard for general instances of  $G$ . Besides, pair wise propagation delay information  $d$  may not be known (the propagation delays, if known, could easily be compensated for in a pair wise manner, and there would be little sync error.). Hence, in this paper, we propose to construct heuristic propagation structures using connectivity information while keeping the two over-arching metrics in consideration. Moreover, for common propagation structures, such as a rapidly-constructed level-based tree [5], analyzing sync error between nodes synchronized along a path is inadequate since the maximum sync error typically occurs for neighboring nodes that are synchronized along *non-intersecting paths*. This is because although slot-align messages are propagated along the spanning tree, neighboring nodes in the original graph  $G$  may not have parent/child relationships in the spanning tree.

#### B. Heuristic Sync-tree structures for Net-sync

We first discuss elements of the sync structure that impact maximum sync error and message efficiency of net-sync. Then, we provide detailed analyses and simulation results for several candidate sync message propagation structures and propose a particular level-based tree structure that has good sync error and message efficiency properties. For this sync-tree, we derive upper bounds on the sync error. Fig. 6 shows an example of such a level-based spanning tree. We use standard graph terminologies *parent*, *child*, *leaf node*, and *non-leaf node* to refer to node relationships in the tree.

Maximum sync error between neighboring node pairs is an important metric for a net-sync propagation structure. As mentioned earlier, our slot-sync scheme eliminates dominant sources of error inherent in time-sync protocols. The dominant source of error now is propagation delay, which may be significant for networks with long links. If nodes do not have information about the propagation delay between all neighboring node pairs, then it is impossible to construct an optimal spanning tree that minimizes the maximum sync error using any algorithm. Hence, we can only use hop-based algorithms to approximate this performance. On the other hand, if nodes indeed possess information about pair wise propagation delay, then sync error can almost be eliminated by an additional handshaking step in the slot-sync protocol.

The number of slot-align message broadcasts is the other important metric for the net-sync propagation tree, and it is equal to the number of non-leaf nodes in the tree. Hence, the message efficiency (and in turn the amount of energy) for

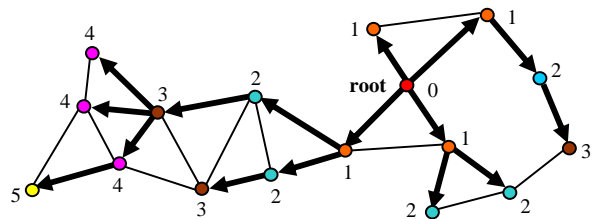


Fig. 6 Level-based Spanning Tree for Slot-align message Propagation. Nodes align slot boundaries to parent nodes pointed to by the arrows

achieving net-sync is proportional to the number of non-leaf nodes; therefore, minimizing the number non-leaf nodes in the spanning tree is a desirable goal. It turns out that this is equivalent to finding the minimum connected dominating set (MCDS) of  $G$ . This has been shown to be NP-hard [15]; however, several heuristic algorithms have been proposed for the MCDS problem and the best heuristic we know of in finding the minimum number of non-leaf nodes is the breadth-first-search (BFS) spanning tree. It has been shown to be within a constant factor of 8 from the optimal [16]. From an implementation perspective, a BFS spanning tree lends itself to simple construction in wireless sensor networks. We derive performance bounds for BFS spanning trees and show considerable gains over the rapidly constructed level tree used in [5] both in terms of maximum sync error and number of non-leaf nodes, particularly if certain attributes of the BFS tree are improved upon.

First we review standard graph theoretic terminology. Let the length of the shortest path (in terms of hop count) between nodes  $x$  and  $y$  be  $h(x,y)$ . The *eccentricity* of a node  $i$  is  $e_i = \max_{j \in V} h(i,j)$ . The *diameter* and *radius* of graph  $G$  are  $D_G = \max_{i \in V} e_i$  and  $R_G = \min_{i \in V} e_i$  respectively. The *center* of graph  $G$  is a set of vertices that has node eccentricity equal to the radius. A *central Point* is a node in the set of graph centers. For any graph  $G$ ,  $R_G \leq D_G \leq 2R_G$ .

For a BFS spanning tree with an arbitrary root node  $r$  (default level 0), the number of hops from  $r$  to any other node is the minimum possible. Let the eccentricity of  $r$  be  $e_r$ , it is easy to show that the number of levels of any BFS spanning tree rooted at  $r$  is equal to  $e_r$ . If  $r$  is a central point, then the number of levels is equal to the radius of  $G$ , and hence is equal to the minimum number of levels possible. We call such a BFS spanning tree that is rooted at a central point a minimum hop-level spanning tree (MHLST). The BFS spanning tree has a few nice properties: it is relatively easy to construct; the maximum sync error is upper bounded by a linear function of the number of levels of the spanning tree; and the number of non-leaf nodes is at most a factor of 8 from the optimal. Additionally, by choosing a *central point* as the root, the number of levels is minimized and sync error upper bound may be reduced by a factor of two.

**Theorem 1.** The maximum net-sync error for a network  $G$ , that is synchronized using a BFS spanning tree rooted at node  $r$  with  $e_r$  number of levels, is upper-bounded by:

$$\Delta = \begin{cases} d(\frac{e_r+1}{2}), & \text{if } e_r = \text{odd} \\ d(\frac{e_r+2}{2}), & \text{otherwise} \end{cases},$$

where  $d$  is the maximum propagation delay ( $d = \rho/c$ , where  $\rho$  is the maximum transmission range, and  $c$  is the speed of light). This bound is tight.

*Proof:* Refer to Fig. 7, one can show that such networks, with  $\phi > 60$  degrees, can be constructed and meet the bound [14].

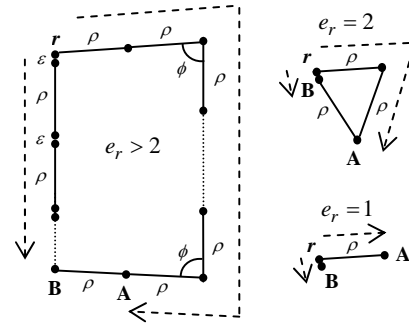


Fig. 7 Type of Graphs that Achieve the Maximum Sync Error Upper Bound

**Corollary 1.** The maximum sync error in a graph  $G$  is upper bounded by  $d(\frac{R_G+1}{2})$ , if  $R_G = \text{odd}$ ;  $d(\frac{R_G+2}{2})$ , otherwise.

*Proof:* The minimum number of levels in a BFS spanning tree is equal to the graph radius  $R_G$ . Hence for a network synced using a MHLST, the upper bound can be written for  $e_r = R_G$ . This can serve as an upper bound on the maximum sync error for a network synced using any sync structure.

**Corollary 2.** MHLST can reduce the maximum sync error by at most one-half compared to a BFS spanning tree not rooted at a central point.

*Proof:* A BFS spanning tree has a maximum number of levels that is equal to the graph diameter  $D_G$ , with maximum sync error upper bounded by  $(D_G+1)d/2$ . Since  $D_G \leq 2R_G$ , the upper bound can be written for  $e_r = 2R_G$ . This is less than a factor of two times the upper bound for a MHLST.

We investigate a plethora of spanning trees for distributing the slot align message from the root to all nodes in the network. They fall in three broad categories:

(1) **Rapidly-constructed Spanning Tree** [5] The root node starts by broadcasting a sync message; neighbors that hear this message rebroadcast it after a random delay. This process is continued until all nodes are synchronized. The resulting tree is level-based, but could potentially have a large number of levels as well as a large number of broadcasting nodes.

(2) **Spanning trees based on min hop metric** If network connectivity (or propagation delay) information was widely available, Dijkstra's shortest path algorithm can be used to calculate shortest path trees rooted at a given node.

(3) **BFS Spanning trees** We investigate two optimizations on the basic BFS spanning tree:

*Node-sort Spanning Tree* The root node starts by propagating a slot-align message. Neighbors propagate this packet to their neighbors to form a BFS spanning tree. At each level, nodes send the slot-align message in decreasing order of the number of possible children nodes. This tends to result in fewer parent nodes than a general BFS tree rooted at the same root node.

*Edge-sort Spanning Tree* This tree is similar to the BFS Node-sort Spanning Tree. The difference is that at each level, nodes send the slot-align message in decreasing order of the number of edges connecting its possible children. Since children that

have the same parent tend to have lower relative sync error, this tree tends to have better average sync error performances than other BFS trees.

The choice of the root node is orthogonal in each of the above trees and has its set of associated tradeoffs. Choosing a *central point* as the root (denoted by CPR) is desirable for reducing the maximum sync error, but computing this is usually a non-trivial protocol step. On the contrary, choosing a root node heuristically or randomly (denoted by RR) may be faster but may yield suboptimal sync error and message efficiency performance.

## VI. SIMULATION RESULTS

Our simulation setup consists of a network of 100 nodes uniformly and randomly distributed in a unit-area square region. Fig. 8 plots the pair-wise sync error for all simulated trees normalized by multiplying by the speed of light<sup>2</sup>. Through more simulations, we observe that in general, BFS based trees and the min-hop Dijkstra tree have comparable sync error and are significantly better than the rapidly-constructed tree both in the maximum sync error and in the mean. The min-distance Dijkstra tree achieves even lower sync error; however, since distance information is not known to the network, this tree cannot be constructed in practice. In terms of non-leaf nodes, the rapid level-based tree in [5] and the min-distance Dijkstra tree have the worst performance in general, followed by the BFS tree. Node-sort and Edge-sort trees have the least number of non-leaf nodes in most cases. In general, the CPR versions of trees generate much fewer non-leaf nodes than the corresponding RR versions.

To verify these observations, we simulate 500 random topologies for a set of transmission ranges. The average number of non-leaf nodes and the average maximum normalized sync error are plotted in Fig. 9 (the average is taken over the topologies). Figs. 9 also includes plots of percentage savings in average number of non-leaf nodes and average maximum sync error over those generated by the rapidly-forming tree for performance comparisons (the plots show percentage savings for CPR only; RR generates similar plots). As expected for geometric random networks, there exists a critical transmission range threshold, below which the network is disconnected with high probability. We observe a second phase transition threshold, beyond which the network is almost fully connected with graph radius of less than 2. For the 100 node network we simulate, this threshold occurs at a transmission range of approximately 0.6. For most WSNs, the interesting region lies between the two phase transition regions where the network is connected, but not almost fully connected. Hence, the plots we generate are over the transmission range of 0.14 to 0.6 only.

Fig. 9 shows that the rapidly-forming tree and the min-distance Dijkstra tree generate significantly more non-leaf nodes than BFS based trees. Node-sort and Edge-sort trees generate the least number of non-leaf nodes in general. When

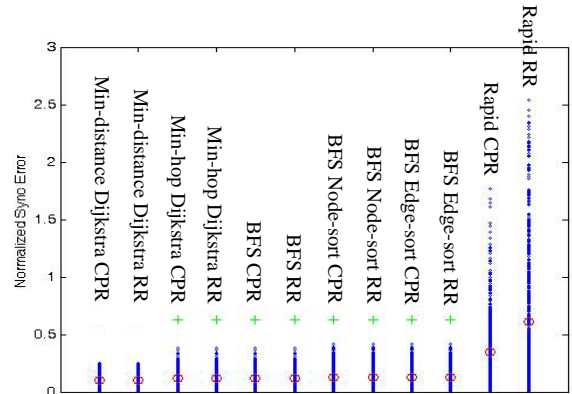


Fig. 8 Normalized Sync Error

Each dot represents sync error of a node pair in the network. Circles mark avg. sync error over all node pairs; '+' signs mark upperbounds

rooted at central points, up to 70% savings can be achieved when compared to the rapidly-forming tree rooted at the same nodes. On average, all BFS based trees have comparable sync error performances. Compared to the rapidly-forming level tree, up to 80% sync error reduction can be achieved using BFS based trees. We also simulated networks with fewer and more nodes as well as over rectangular regions [14]. Similar curves have been obtained. For all the types of trees simulated, choosing a central point as the root node reduces the number of non-leaf nodes significantly compared to choosing a random node as the root. Up to 40% reductions is achievable for BFS trees. An additional 25% reduction over BFS trees can be achieved if Node-sort or Edge-sort trees are used [14].

Our proposed techniques are applicable to the network wide time-sync problem as well. The tradeoffs between maximum sync error and message efficiency in achieving sync are fundamental in nature and are applicable to both slot sync and time sync problems. Also, the various tradeoffs studied for slot guard time design in Section IV are applicable to any slotted network system that cares about energy efficiency.

## VII. CONCLUSION AND FUTURE WORK

In this paper, we presented a new technique for achieving slot alignment in wireless sensor networks termed Slot-Sync. First, we provided an energy efficient slot design for guard times and re-sync periods. We then presented an energy-efficient slot formation and maintenance scheme that uses an appropriately chosen rooted spanning tree structure known as the sorted-MHLST for slot-align message propagation throughout the WSN. Through simulations, we showed that the choice of the sync tree and that of the root node are very important both in minimizing the worst-case sync-error and the energy consumption of the sync/re-sync process. Up to 80% reduction in sync error and up to 70% reduction in energy for sync message propagation can be achieved over the rapidly constructed tree used in [5]. Using graph theoretic approaches, we derived an upper bound on the maximum sync error in WSNs and showed that the node pairs that have the maximum sync error typically do not have parent/child relationship in the sync tree.

<sup>2</sup> The transmission radius is 0.2. If the operating area and tx radius are scaled by 1000, then a normalized max sync error of 2.5 (Rapid tree) corresponds to about 8.3  $\mu$ -sec. For longer haul links, this sync error could be much worse.

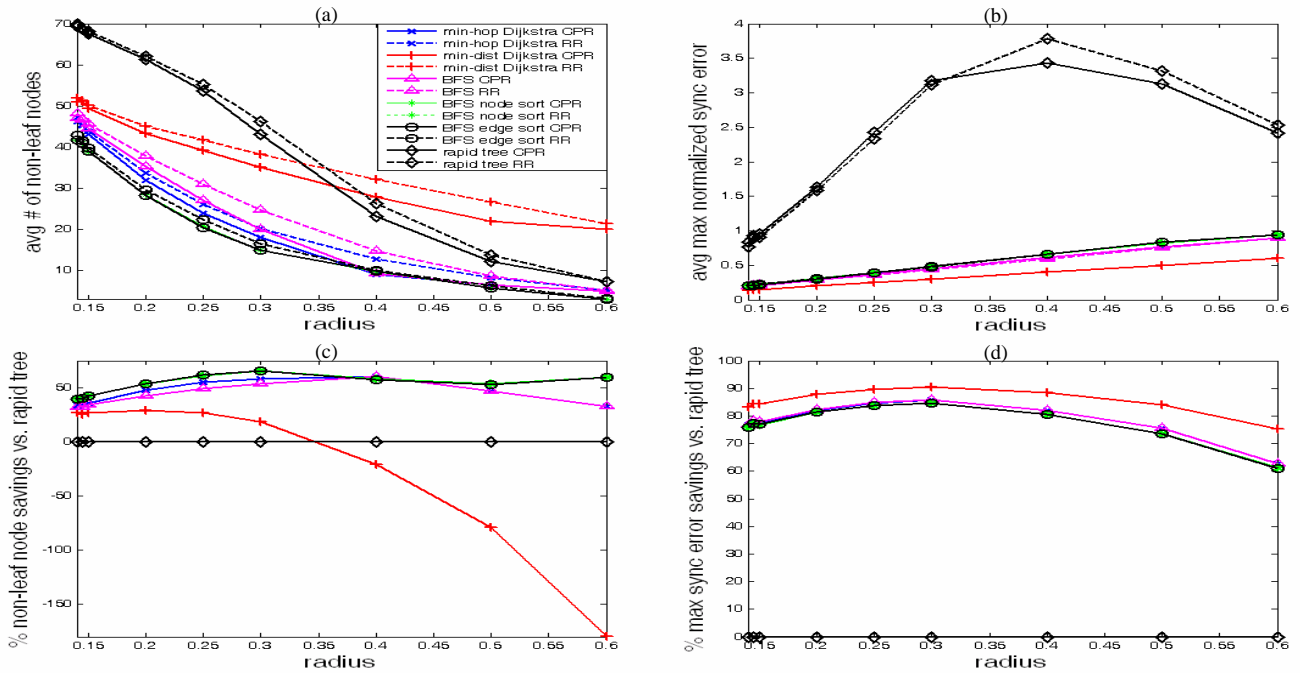


Fig. 9 Simulation results for 100 node networks in unit area square region; for each transmission radius, 500 random topologies were generated. (a) Average number of non-leaf nodes; (b) Average maximum normalized sync error; (c) Percentage reduction in non-leaf nodes over the rapidly-forming tree (d) Percentage reduction in sync error over the rapidly-forming tree

Although we presented our Slot-Sync scheme in the context of WSNs, our approach is more general and can be applied to general multi-hop wireless networks with moderate traffic load and low mobility. We are investigating solutions to the slot misalignment problem for networks with greater mobility. In particular, we are developing distributed algorithms for achieving net-sync in unreliable networks where partitions form and heal dynamically.

We are also working on energy comparisons of the proposed slot-sync scheme when applied to well known synchronous MAC schemes with asynchronous MAC schemes that do not require sync. Duty cycling presents additional challenges to the protocol for maintaining net-sync – the time to propagate a message through the network increases as a function of the duty cycling period. We are currently investigating this issue actively.

Additional future research includes a stochastic analysis of guard time and re-sync period in contrast to the worst-case analysis presented in this paper. Our simulations indicated that the mean sync error can be substantially lower than the worst case error. Furthermore, the mean clock drift is much lower than the worst case clock drift reported by the manufacturer. Hence, for duty-cycling networks that can tolerate occasional data loss, stochastic design of guard times and re-sync period may achieve substantial energy savings.

## REFERENCES

[1] K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie, "Protocols for Self-organization of a Wireless Sensor Network," *IEEE Personal Communications*, Oct. 2000.

[2] J. Hill and D. E. Culler, "MICA: A Wireless Platform for Deeply Embedded Networks," *IEEE Micro*, 22(6):12-24, Nov. 2002.

[3] J. Redi, I. Castineyra, C. Partridge, K. Manning, R. Rosales-Hain, R. Ramanathan, and S. Kolek, "Joint Architecture Vision for Low Energy Networking (JAVeLEN) – DARPA-ATO Connectionless Networking Program, Phase I", BBN Technical Report 8408. Dec. 2004.

[4] J. Elson, L. Girod, and D. Estrin, "Fine-grained Time Synchronization using Reference Broadcasts," *Proc. USENIX OSDI*, Boston, MA, Dec. 2002.

[5] S. Ganeriwal, R. Kumar, and M. Srivastava, "Timing Sync Protocol for Sensor Networks," *Proc. ACM SenSys*, Los Angeles, CA, Nov. 2003.

[6] M. L. Sichitiu and C. Veerarittiphan, "Simple, Accurate Time Synchronization for Wireless Sensor Networks," *Proc. IEEE WCNC*, 2003.

[7] J. Van Greunen and J. Rabaey, "Lightweight Time Synchronization for Sensor Networks," *Proc. ACM Int'l. Conf. Wireless Sensor Networks and Applications*, San Diego, CA, Sept. 2003.

[8] Q. Li and D. Rus, "Global Clock Synchronization in Sensor Networks," *Proc. IEEE Infocom*, 2004.

[9] T. Salonidis and L. Tassiulas, "Performance issues of Bluetooth scatternets and other asynchronous TDMA ad hoc networks," *Proc. MoMuC 2003*.

[10] A. Ebner, H. Rohling, M. Lott, and W. Halfmann, "Decentralized Slot Synchronization in Highly Dynamic ad hoc Networks," *Wireless Personal Multimedia Communications*, vol. 2, pp. 494 – 498, Oct. 2002.

[11] D. Mills, "Internet Time Synchronization: The Network Time Protocol," *IEEE Trans. Communications*, 39(10), pp. 1482-1493, Oct. 1991.

[12] M. L. Sichitiu and C. Veerarittiphan, "Simple, Accurate Time Synchronization for Wireless Sensor Networks," *Proc. IEEE WCNC*, 2003.

[13] Symmetricom white paper, "Symmetricom SmartClock Technology - Improving Oscillator Long-term Stability for Synchronization Applications." <http://www.symmetricom.com>

[14] L. Dai, P. Basu, and J. Redi, "Energy Efficient Slot Synchronization Techniques for Wireless Sensor Networks," BBN Technical Report, Jul. 2005.

[15] M. R. Garey and D. S. Johnson, "Computers and Intractability – A Guide to the Theory of NP-completeness," W. H. Freeman, San Francisco, CA, 1979.

[16] K. M. Alzoubi, P.-J. Wan, and O. Frieder, "Distributed Heuristics for Connected Dominating Sets in Wireless ad hoc Networks," *Journal of Communications and Networks*, 4(1), Mar. 2002.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.