Asymptotically Optimal Time Synchronization in Dense Sensor Networks

An-swo Hu  Sergio D. Servetto
School of Electrical and Computer Engineering – Cornell University  
http://cn.ece.cornell.edu/

ABSTRACT

We consider the problem of synchronization of all clocks in a sensor network, in the regime of asymptotically high node densities. We formulate this problem as one in which all clocks must line up with the clock of an arbitrary node in the network (of course without assuming that this clock can be observed everywhere in the network, nor assuming that this node has any special hardware—this node could be any). We give a state-space description for the generation of observable data as a function of the ideal clock, and we derive an optimal estimator for determining the state of the ideal clock. A salient feature of our approach is that nodes collaborate to generate an aggregate waveform that can be observed simultaneously by all nodes, and that contains enough information to synchronize all clocks. This aggregate waveform effectively simulates the presence of a “super-node” capable of generating a high-power, network-wide time reference signal.

Categories and Subject Descriptors

C.2 [Computer-Communication Networks]: Network Architecture and Design; Wireless Communication

General Terms


1. INTRODUCTION

1.1 Problem Setup

Consider the time synchronization problem. \( N \) nodes are placed uniformly on the surface of a square of unit area. For simplicity of presentation, but without any loss of generality, we will assume one of the \( N \) nodes falls at the center of the square, and we will refer to this node as “node 1.” Each node \( i \) is equipped with a local clock \( c_i \) modeled as

\[ c_{i,t} = \alpha_i (t - \bar{\Delta}_i) + \Psi_i (t), \]

where \( \alpha_i \) is due to a constant frequency bias in the oscillator, \( \bar{\Delta}_i \) is a constant offset, and \( \Psi_i (t) \) is white noise (for a fixed variance, the noise model that makes prediction the most difficult). Node 1 will have an operational counter, \( s_1 (t) \), that increments at integer values of \( c_1 \). We will say that node \( i \) is synchronized to node 1 if the operational counter of node \( i \), \( s_i (t) \), which is a function of \( c_i \), increments at the same time as \( s_1 (t) \) in the deterministic case, or the mean squared error between \( s_1 (t) \) incrementing and \( s_i (t) \) incrementing is minimized in the random case. This setup is illustrated in Fig. 1. In the problem of global clock synchronization, the goal is to get all nodes in the network synchronized with the clock of node 1. We will focus primarily on the case in which we have a large number of nodes in the network (i.e., the asymptotic performance of clock synchronization techniques in the limit as \( N \rightarrow \infty \).

Figure 1: In the figure above, the sine waves are the oscillators of node 1 and node \( i \). Notice that even though the oscillator for node 1 runs twice as fast and with some offset as compared to that of node \( i \), the operational counters of node 1 and node \( i \) still increment at the same times.

1.2 Relevance of High-Density Asymptotics

In this work, we study the time synchronization problem under the assumption that the sensed area is finite, but letting the density of nodes grow unbound. Since these assumptions imply the use of point sensors with no mass in the limit, it is clear that some justification is needed for why this is an interesting regime to study. And the answer is given by scalability issues.

Whereas infinitely large networks consisting of nodes with zero mass are clearly not realizable by physical devices (if nothing else before, this would imply a violation of the laws of quantum physics), there is a trend towards miniaturization of these devices. For example, in recent work, a hardware simulation-and-deployment platform for wireless sensor networks capable of simulating networks with on the order of 100,000 nodes was developed [8]. As well, for many years now the Smart Dust project has been seeking to build cubic-millimeter motes for a wide range of applications [21]. With large numbers and small nodes, we face a situation involving networks operating at high densities. We expect that our analysis...
based on high-density asymptotics will provide useful insights on
the behavior of such networks at high but still finite densities—
basically, algorithms whose performance and complexity can be
shown to have favorable scaling laws we expect will perform well
in practical regimes well approximated by the limit behavior.

Note however that in setting up such high-density asymptotics,
we have to be careful to not trivialize the problem. For example,
in such a regime, it seems particularly inappropriate to assume that
each node is able to store/harvest a constant amount of energy. In
the limit, this would imply that the whole network has access to an
infinite supply of energy over a finite area, and therefore conclu-
sions derived from this model would be of questionable value—for
example, any communications aspect of the problem becomes triv-
ial, since infinite power would allow us to ignore any form of noise
in the channel. Furthermore, there is work in progress on the minia-
turization of power sources: one example of a radioactive battery
capable of powering 2-4 mm$^3$ nodes has been developed that is ca-
ble of delivering approx. 1-2 nanoWatts/sec [11]—this is several
orders of magnitude below the power output of chemical batteries
used for larger devices.

In summary, we consider in this work the problem of time syn-
chronization under the assumption of asymptotically high densities
and vanishing per-node power. We work under these assumptions
because we feel they enable analytical work that can address crit-
cal scalability issues in large-scale sensor networks. And we be-
lieve that the conclusions derived from our analysis are not just
"cute math, but irrelevant": we believe they provide valuable in-
sights for how to design and operate the network at high but still
finite densities. In the specific context of the time synchronization
problem, synchronization methods developed for locking a small
finite densities. In the specific context of the time synchronization
problem, synchronization methods developed for locking a small
numbers of nodes to an ideal clock would result in poor network
wide synchronization in dense networks, since errors accumulate as
nodes many hopes away from the ideal clock are synchronized. Yet
the scheme proposed in this paper has synchronization performance
that is independent of the number of nodes and/or of the distance
from any node to the reference source, and has complexity that is
independent of the number of nodes in the network.

1.3 The Sensor Reachback Problem

Our interest in synchronization arises from the idea of a dis-
tributed transmission array for the sensor reachback problem [1,
7]. In this problem, the goal is to move the field of observations
picked up by all sensors to a far receiver. What makes the reach-
back problem interesting is the fact that typically, each individual
sensor does not have enough resources to generate a strong infor-
mation-bearing signal that can be detected reliably at the far re-
ceiver. By synchronizing all nodes in the network, they can act as
a distributed transmission array and cooperatively generate a signal
that can be reliably detected at the receiver. This is illustrated in
Fig 2.

1.4 Main Contributions and Organization

The main contribution presented in this paper is the design and
performance analysis of a new family of clock synchronization
techniques which, in the limit as the number of nodes in the net-
work goes to infinity (but keeping the total power budget bounded),
achieves the same performance as if we had one “super node” gen-
erating a high powered reference signal heard, in one hop, by every
other node in the network simultaneously. This is accomplished by
means of cooperation among nodes, to collectively produce a wave-
form in the air which, as $N$ gets large, provides to each node all the
information needed to achieve globally optimal synchronization.
The key intuition behind our development here is that independent
timing errors across nodes cancel out, leading to a deterministic
aggregate waveform as a result of the law of large numbers.

The remainder of the paper is organized as follows. In Section 2
we describe our system model. In Section 3 we propose a syn-
chronization protocol. In Section 4 we formulate the optimality
criteria for network-wide synchronization, we develop an optimal
estimator to solve the synchronization problem within the broad-
cast domain of the node with the ideal clock, we extend this esti-
mator to solve the network-wide synchronization problem, and we
prove that as the density of the network grows unbound this proce-
dure yields optimal synchronization. This section concludes with a
discussion of “real world” issues. We present some discussion of
related work in Section 5, and concluding remarks in Section 6.

2. SYSTEM MODEL

2.1 Oscillator Instability Model

From [6] we derive a model for modelling frequency instabil-
ity. The instantaneous frequency in the $i$th oscillator, $f_i(t)$, will be
modelled as

$$f_i(t) = f_o + \Delta f + \bar{f}(t-t_o) + f_r(t),$$

where

- $f_o$ is the ideal frequency
- $\Delta f$ is the frequency bias or offset
- $\bar{f}$ is the frequency drift
- $f_r(t)$ is a random process for the unmodelled random fre-
quency errors

and $t_o$ is an arbitrary initial time reference with respect to the ideal
clock that the variable $t$ represents. Given $f_i(t)$ we can establish a
timing relationship. If we assume that the counter associated with
oscillator $i$ increases continuously with the oscillator and counts
from $k$ to $k+1$, where $k$ is an integer, every complete cycle of the
oscillator, then we have

$$t_i(t) - t_i(t_o) = \frac{1}{f_o} \int_{t_o}^{t} f_i(t) dt,$$

where $t_i(t)$ is the time at clock $i$, at the ideal time $t$.

By combining the two previous equations, we can obtain the ex-
pression for the time of clock $i$ at a given ideal time $t$ as

$$t_i(t) = t_i(t_o) + (t - t_o) + \frac{\Delta f}{f_o}(t-t_o) + \frac{\bar{f}}{2f_o}(t-t_o)^2 + \Psi_i(t),$$

Figure 2: Cooperation among sensors to reach back to a far receiver.
A good analogy to describe the role of synchronization in the context
of reachback is that of a symphonic orchestra. When all instruments
in the orchestra play independently, all we get is noise; but when they
can hear each other and cooperate, the music from all instruments is
combined into a coherent play. Synchronization of a sensor network is
what allows the nodes to “hear” each other and achieve cooperation.
where \( \Psi_i(t) = \frac{1}{T} \int_{s_{i-1}}^{s_i} f_i(t)dt \). In this work, we simplify the model by assuming there is no frequency drift. This yields the following simplified model

\[
t_i(t) = t_i(t_0) + (t - t_0) + \frac{\Delta}{f_0}(t - t_0) + \Psi_i(t).
\]

For simplicity, in this work we assume that \( \Psi_i(t) \) is a zero mean, white Gaussian process implying that samples \( \Psi_i(t_j) \sim \mathcal{N}(0, \sigma^2) \), for \( j \in N \), are independent and identically distributed. Note that \( \sigma^2 \) is defined in terms of the clock of node \( i \). We make this independence assumption to simplify notation for presenting the ideas of this work. The independence assumption actually leads to conservative estimates of the minimum achievable mean squared error since with correlated noise, the innovations sequence associated with the samples of \( \Psi_i(t) \) has variance less than \( \sigma^2 \) [19] and hence leads to an even lower mean squared error for our estimators.

### 2.2 Clock Model

We consider a sensor network with \( N \) nodes. The clock of one particular node in the network will serve as the ideal time and to this clock we wish to synchronize all other nodes. To make this point clear, the synchronization methods presented in this work synchronize the clocks of all nodes in a network to the clock of one particular node. This is done to make the synchronization scheme self-contained when the only clocks that the network has access to are the clocks of the nodes. If we want the network synchronized to “real-time”, then the node initiating synchronization would need to have access to it. According to the recommendations of Elson and Römer [3], we allow the local clock of each node to be free-running. We never adjust the local clock frequency or offset, but instead we seek to construct an “operational” clock on top of the free-running local clock. The operational clock of each node will be synchronized to the ideal clock and it will be defined in terms of that node’s local clock.

Without loss of generality, we will call the node with the ideal clock node 1. The clock of node 1 will be defined as \( c_{1,t} = t \). We also define the counter \( c_{1,t} = \lfloor t \rfloor \) where \( t \in [0, \infty) \). At any time \( t_0 \), \( c_1(t_0) \) is the number of ticks the counter of node 1 has made. From the expression for \( t_1(t) \), we can easily see that the clock of node 1 ticks on integer values of \( t \). We define the counter \( c_{1,t} \) to simplify the description of the synchronization procedure since all synchronization pulses are sent at integer values of \( t \).

Taking \( c_1 \) to be the ideal clock, we now define the clock of an arbitrary node \( i \) using the simplified oscillator frequency model that assumes no frequency drift. We define \( c_i \) as

\[
c_{i,t} = \alpha_i(t - \Delta_i) + \Psi_i(t)
\]

and its associated counter

\[
c_i(t) = \lfloor \alpha_i(t - \Delta_i) + \Psi_i(t) \rfloor,
\]

where \( \alpha_i = 1 + \frac{\Delta_i}{f_0} \) and \( c_0 = 0 \) when \( t = \Delta_i \). \( \Delta_i \) is a constant offset that models the fact that it is not known when \( c_i \) is started relative to \( c_x \).

As mentioned earlier, the clocks \( c_1 \) and \( c_i \) for all \( i \) will be free running clocks that will have a synchronized “operational” counter built on top of them. This operational counter is set up in the following manner. We first assume that node 1 at time \( t_0 \) decides it needs to synchronize the remaining \( N-1 \) nodes. Node 1 will increment its operational counter to a value of 1 at the next integer time \( t \). That is to say, the operational counter of node 1, denoted by \( s_1(t) \), will be \( s_1(t) = \lfloor t - n_0 \rfloor \) where \( n_0 = \lfloor t_0 \rfloor \). Our goal, ideally, will then be to construct at node \( i \) an identical operational counter \( s_i(t) = \lfloor t - n_i \rfloor \). We want the operational counter at the \( i \)th node to increment at integer values of \( t \) and hold a value equal to \( s_1(t) \).

### 2.3 Observation Model

Synchronization will be achieved by the transmission and observation of pulses. We first make the following assumptions about pulse transmission and reception:

- **no propagation delay**: We assume there is no delay between the time a pulse is transmitted and the time it is seen by other nodes. This is a reasonable assumption since the propagation time of radio waves travelling at the speed of light over small transmission distances is negligible. The transmission range of a node in our network will be small since we are considering extremely small nodes with very low power.
- **no transmission delay or time-stamping error**: We assume that a pulse is transmitted at exactly the time the node intends to transmit it. We make this assumption since there will not be any delay in message construction or access time [2] since our nodes broadcast the same simple pulse without worrying about packet collisions. Also, when a node receives a pulse it can determine its clock reading without delay since any time stamping error is small and can be absorbed into the random clock jitter.

Because pulses are exchanged among many different nodes, to clearly describe transmission and reception times in relation to different clocks, we define the following notation:

- \( s_{jk}^s \) is the time, with respect to clock \( c_k \), that the \( i \)th node sees its \( j \)th pulse. For example, \( s_{12}^s \) is the time, with respect to clock \( c_1 \), that the 3rd node sees its 2nd pulse.
- \( s_{jk}^a \) is the time of the \( m \)th transition of the operational counter \( s_i(t) \) with respect to \( c_k \).
- Let us also say that in general, any value or variable \( X_{i,j}^c \) means that we are considering the value of \( X \) in terms of the time scale of \( c_i \).

See Fig. 3 for an illustration.

![Figure 3: The above figure illustrates \( c_i \) and \( c_j \) as well as the operational counter of node 1, \( s_1(t) \). We assume node 3 is in the broadcast domain of node 1. In this illustration we assume \( \Psi_3(t) = 0 \) (no random clock jitter), \( \alpha_3 = 1 \), and \( \Delta_3 = 1.5 \). If \( s_1(t) \) increments, the second pulse will be transmitted at \( s_{2,1}^a = 4 \). Since node 3 can hear the pulses of node 1, this pulse will be the second pulse heard by node 3. This occurs at \( s_{3,2}^a = 2.5 = s_{2,1}^a \).](image-url)
i within the broadcast domain of node 1. We only consider this case because we will later show that this is the only important case. We recall that by definition, \( s^{C,1}_{n+1,i} \) will be an integer and at this time, a pulse will be transmitted. Because node i is in the broadcast domain of node 1, we can describe the pulse receive time at node i, with respect to the clock of node i, in terms of the pulse transmission time (or equivalently, the time at which the operational clock of node 1 increments) as the state equations

\[
\begin{align*}
    s^{C,1}_{n+1,i} &= s^{C,1}_{n,i} + 1 \\
    t^{C,1}_{n,i} &= \alpha(s^{C,1}_{n',i} - \Delta) + \Psi(s^{C,1}_{n',i}).
\end{align*}
\]

(2)

The second equation makes use of the clock model of node i (1). Under the assumption that node i being in the broadcast domain of node 1, \( n' = n \). However, this does not hold in general because in the multi-hop case the \( n' \)th pulse observed by the \( j \)th node does not necessarily correspond to the \( n' \)th pulse transmitted by node 1. So in general, if we assume \( n' = n \), then the expression is saying that the pulse seen by node j at \( t^{C,1}_{n,i} \) is occurring at \( s^{C,1}_{n+j,k} \).

3. A SYNCHRONIZATION PROTOCOL

We consider a network of \( N \) nodes, uniformly distributed over the \([0, 1] \times [0, 1]\) plane. We describe the mechanism for synchronizing this network to the clock of node 1. Note that this description only gives the overview of what nodes will do to achieve synchronization. Showing that such a procedure will actually result in an optimally synchronized network as \( N \to \infty \) will be left for Sections 4.2 and 4.3. Synchronization will be achieved in the following manner. Node 1 will start transmitting pulses and continue to transmit pulses every time the counter \( s_1(t) \) increments. After the initial \( m \) pulses, the set of nodes in the broadcast domain of node 1, not including node 1, will make an optimal estimate of the location of the \( m + 1 \)th pulse and transmit at that time. The set of nodes in the broadcast domain of node 1 will be called \( R_2 \). The nodes in \( R_2 \) will then use its most recent \( m \) observations to optimally estimate the time of pulse \( m + 2 \). The \( R_2 \) nodes will continue in this manner. The nodes that can hear the aggregate transmissions from \( R_2 \) and node 1, the \( R_3 \) nodes, will begin their own predictions and transmissions after observing \( m \) pulses. This propagation will then continue until all nodes in the network hear signals. Fig 4 illustrates this propagation.

When a new set of nodes first begin to transmit, say \( R_j \), they send out a packet of information following their first pulse that has the count of the operational counter. If we call this value \( v \), then when nodes \( R_{j+1} \) start transmitting, they will set their operational counter to \( v + m \) and send this value out in a packet after the first pulse. This will let all the operational counters to not only increment at the same time but also have the same value. The one other piece of information contained in the packet is a count, \( q \), of how many hops out from node 1 the synchronization has gone. For example, node 1 would send \( q = 1 \). After \( m \) pulses, node 1 would internally increment its value, \( q + 1 = 2 \). It would then continue to increment \( q \) every \( m \) pulses. When the \( R_2 \) nodes send their data packet, \( q = 2 \). They too would then increment their own internal value of \( q \) every \( m \) pulses. The importance of \( q \) is so that nodes can scale their signal amplitudes to keep the aggregate amplitude roughly constant. Nodes would scale their transmit amplitude by \( \eta_q \), where \( \eta_q \) is defined in Section 4.3. Since we consider finite area networks, the nodes would know approximately how many hops are required to cover the entire network.

It is important to note that the packet distribution overhead is only for the initial synchronization phase. After all nodes in the network have heard synchronization pulses, the network no longer needs to distribute information other than the synchronization pulses. In terms of energy consumption, the fact that only synchronization pulses are required means very little energy will be consumed to maintain synchronization. A pseudo-code description of the synchronization protocol is given in Table 1.

<table>
<thead>
<tr>
<th>TimeSync (observation length ( m &gt; 1 ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>observe pulse arrival time;</td>
</tr>
<tr>
<td>if (first observed pulse)</td>
</tr>
<tr>
<td>{ receive packet and set ( v, q ) values; }</td>
</tr>
<tr>
<td>while (pulse arrived)</td>
</tr>
<tr>
<td>{ if (( m ) or more arrival times in memory)</td>
</tr>
<tr>
<td>{ keep only ( m ) most recent and discard all other arrival times;</td>
</tr>
<tr>
<td>use last ( m ) arrivals to estimate next pulse time ( t_p );</td>
</tr>
<tr>
<td>transmit scaled pulse ( p(t) ) at time ( t_p );</td>
</tr>
<tr>
<td>if (first pulse transmitted)</td>
</tr>
<tr>
<td>{ transmit packet with ( v + 1, q ); }</td>
</tr>
<tr>
<td>}</td>
</tr>
<tr>
<td>observe pulse arrival time;</td>
</tr>
<tr>
<td>if (4 pulses received is multiple of ( m ))</td>
</tr>
<tr>
<td>{ set ( q = q + 1 ); }</td>
</tr>
<tr>
<td>set ( v = v + 1 );</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>

Table 1: Pseudo-code description of the synchronization algorithm executed by each node.

4. ASYMPTOTICALLY OPTIMAL CLOCK SYNCHRONIZATION

4.1 Optimality Conditions

Optimal Synchronization

The problem of synchronization is the challenge of having the \( t \)th node accurately and precisely predict when node 1 increments its operational counter. In our setup, the reception of a pulse by node \( i \) tells it of such an event. Recalling that \( \Psi(s^{C,1}_{n,i}) \sim \mathcal{N}(0, \sigma^2) \),
from (2) we see that the pulse receive time at node $i$, $t_{c_n,i}$, is a Gaussian random variable. Thus, to achieve synchronization node $i$ will try to estimate the random variable $t_{c_n,i}$ using a series of $m$ pulse receive times as observations. We define optimal synchronization as node $i$ making an estimate of $t_{c_n,i}$, called $\hat{t}_{c_n,i}$, that meets the following optimality criteria:

$$E(\hat{t}_{c_n,i}) = E(t_{c_n,i})$$

$$\arg\min_{\hat{t}_{c_n,i}} E(\|\hat{t}_{c_n,i} - t_{c_n,i}\|^2)$$

The first optimality condition comes from the fact that given a finite $m$, it is reasonable to want the expected value of the estimate to be the expected value of the random variable being estimated. The second condition is the result of seeking to minimize the mean squared error between the estimate and the random variable being estimated.

### An Optimally Synchronized Network

The above optimality conditions apply to all nodes trying to synchronize to the operational counter of node 1. It is, however, possible for a node $j$ to achieve the optimality conditions but have its minimum possible mean squared error over the area of the entire network. This means that even though node $j$ is optimally synchronized to $s_1(t)$, node $j$ may still achieve a greater mean squared error than the minimum possible for it over the area of the entire network using the given synchronization methods. For example, this could be because node $j$ is not in the broadcast domain of node 1 and better synchronization would be possible if it were in node 1’s broadcast domain. Because of this, we say that a network is optimally synchronized if all nodes in the network can achieve the optimality conditions of (3) and (4) and the mean squared error achieved in (4) for each node is the smallest possible mean squared error achievable for that node over the area of the network. Thus, the optimality condition for an optimally synchronized network is

$$E(\|\hat{t}_{c_n,j} - t_{c_n,j}\|^2) = \min_A E(\|\hat{t}_{c_n,j} - t_{c_n,j}\|^2)$$

where $A$ is the area of the network. This means that even though node $j$ is optimally synchronized to $s_1(t)$, node $j$ may still achieve a greater mean squared error than the minimum possible for it over the area of the entire network.

#### Observation Distributions

We study a node $i$ within the broadcast domain of node 1 and it will seek to estimate $t_{c_n,i}$ given $t_{c_{n-1},i}, \ldots, t_{c_{n-m},i}$. Using the past observations $t_{c_{n-1},i}, \ldots, t_{c_{n-m},i}$ node $i$ will try to estimate where its operational counter should next increment, i.e. calculate $s_{c_{n-1},i}$, where $m$ is the number of pulses observed before the $i$th node’s first pulse transmission.

From our state equations (2), we see that $T = [t_{c_{n-m},i}, \ldots, t_{c_{n-1},i}]^T$ is a jointly Gaussian random vector. Recall that we assume $\Psi(t)$ is a zero mean, white Gaussian process meaning that the samples $\Psi(t_j)$ are independent and identically distributed. As a result, from the second state equation we can say that the random variable $t_{c_{n-1},i}$ is Gaussian with $t_{c_{n-1},i} \sim \mathcal{N}(\alpha s_{c_{n-1},i} - \Delta_i, \sigma^2)$ for some $n' \in N$. It is true in this case that node $i$ being in the broadcast domain of node 1, $n' = 1$, but in general we assume $n' = n - k$. We also notice that

$$E(t_{c_{n+1},i}) = \alpha_i(s_{c_{n+1},i} + 1 - \Delta_i) = \alpha_i(s_{c_{n+1},i} - \Delta_i) + \alpha_i.$$
With these two conditions, from [12] we see that the optimal estimate for \( y_{m+1} \) will be the uniformly minimum variance unbiased (UMVU) estimator for \( E(y_{m+1}) = \theta_1 + m\theta_2 \).

Using the above linear model, from [10] we know the maximum likelihood (ML) estimate of \( \theta, \hat{\theta}_{ML} \), is given by

\[
\hat{\theta}_{ML} = (H^T \Sigma H)^{-1} H^T \Sigma^{-1} Y = (H^T H)^{-1} H^T Y.
\]

This estimate achieves the Cramer Rao lower bound, hence is efficient. The Fisher information matrix is \( I(\theta) = \frac{H^T H}{\sigma^2} \) and \( \hat{\theta}_{ML} \sim \mathcal{N}(\theta, \sigma^2 (H^T H)^{-1}) \). This means that \( \hat{\theta}_{ML} \) is UMVU.

Again from [10], the invariance of the ML estimate tells us that the ML estimate for \( \phi = g(\theta) = \theta_1 + m\theta_2 \) is \( \hat{\phi}_{ML} = \hat{\theta}_{ML} + m\theta_{2ML} \). First, it is clear that \( \hat{\phi}_{ML} = C\hat{\theta}_{ML} \), where \( C = [1 \ m] \).

As a result, we first see that \( E(\hat{\phi}_{ML}) = CE(\hat{\theta}_{ML}) = \theta_1 + m\theta_2 \) so \( \hat{\phi}_{ML} \) is unbiased. Next, to see that \( \hat{\phi}_{ML} \) is also minimum variance we compare its variance to the lower bound.

\[
\text{Var}(\hat{\phi}_{ML}) = C\sigma^2 (H^T H)^{-1} C^T = \frac{2\sigma^2(2m+1)}{m(m-1)}.
\]

The extension of the Cramer Rao lower bound in [10] to a function of parameters tells us that

\[
E(||\hat{g} - g(\theta)||^2) \geq G(\theta)I^{-1}(\theta)G^T(\theta)
\]

with \( G(\theta) = \{\nabla g(\theta)\}^T \). In this case, \( G(\theta) = [1 \ m] \) so the lower bound to the mean squared error is

\[
G(\theta)I^{-1}(\theta)G^T(\theta) = \frac{2\sigma^2(2m+1)}{m(m-1)}.
\]

As a result, we see that \( \hat{\phi}_{ML} \) is UMVU.

We recall that \( y_{m+1} = \theta_1 + m\theta_2 + w_{m+1} \), so \( \hat{y}_{m+1} = \hat{\phi}_{ML} + w_{m+1} \). We know that \( w_{m+1} \sim \mathcal{N}(0, \sigma^2) \) and from above,

\[
\hat{\phi}_{ML} \sim \mathcal{N}(\phi, \frac{2\sigma^2(2m+1)}{m(m-1)}).
\]

As a result,

\[
\hat{y}_{m+1} \sim \mathcal{N}(\phi, \frac{2\sigma^2(2m+1)}{m(m-1)})
\]

We note that as \( m \to \infty \), \( \text{Var}(\hat{y}_{m+1}) \to \sigma^2 \).

For any given \( m \), \( \hat{y}_{m+1} \) is the optimal estimate for \( y_{m+1} \). It is clear that the minimum mean squared error achieved with \( \hat{y}_{m+1} \) is the minimum possible for node \( i \) over the entire network area since the nodes in the broadcast domain of node 1 have the most accurate reading of \( c_1 \). In fact, the only noise in the pulse receive time reading comes from the unmodelled random clock jitter \( \Psi_i(t) \).

Please also note that the above optimal estimation is carried out by node \( i \) according to \( c_i \). It is seeking to estimate the next pulse receive time in terms of \( c_i \). Thus,

\[
E(\hat{y}_{m+1}^i) = t_{n+1}^i = \alpha_i(s_{n+1}^i - \Delta_i) + m\alpha_i.
\]

However, in terms of \( c_i \),

\[
E(\hat{y}_{m+1}^i) = t_{n+1}^i = s_{n+1}^i + m.
\]

Also, since the variance \( \sigma^2 \) is defined in terms of \( c_i \), \( \text{Var}(\hat{y}_{m+1}^i) \) will also have a different value since we do a linear transformation to get from \( \hat{y}_{m+1} \) to \( y_{m+1} \). However, we will not go through the calculations since knowing the mean of \( y_{m+1} \) and that it is symmetric about the mean are the important properties utilized in section 4.3.

### 4.3 Multi-Hop Synchronization

#### Assumptions and Definitions

We make the following assumptions regarding our network of \( N \) nodes, uniformly distributed over the \([0,1] \times [0,1] \) plane:

- The transmission from any one node can be heard within a range of

\[
d_N = \sqrt{\frac{\log(N) + C_N}{\pi N}}
\]

from the node. \( C_N \) is any function going to infinity as \( N \to \infty \). The signal magnitude will be assumed to be constant in this range. From [4], we choose this communication range to keep the network connected with high probability as \( N \) gets large.

- Nodes can transmit pulses and receive signals at the same time. This assumption is made to simplify the synchronization mechanism used to develop the mathematics for an optimally synchronized network. Future work will involve developing synchronization mechanisms without this assumption.

- We assume that the signals transmitted by the nodes are not corrupted by channel noise. We make this assumption to study the characteristics of the basic aggregate waveform. The impact of channel noise will definitely be considered in future work.

- \( K \) nodes transmitting within a given circular area centered at node \( i \) can be heard within a range of \( K^{1/\alpha} d_N \) where \( 0 < \alpha < 2 \).

For the last assumption, we use this simple model because we do not know of a power decay model that behaves well for small distances. The common power loss model has power decaying with distance \( d \) as \( 1/d^\beta \), where \( \beta > 2 \), but this model is only accurate outside a small area around the transmitter—for small \( d \), the received power can be arbitrarily large, clearly a bad assumption. Thus, considering an arrangement of \( K \) nodes, we realized that a transmission with \( K \) nodes arranged in a linear topology starting with node \( i \) could reach a distance of \( Kd_N \) from node \( i \), by means of pure relaying. Thus, by choosing \( \alpha \) appropriately, we can achieve both faster than linear and slower than linear transmission range growth with \( K \), depending on the different transmission media. Please note that the development of a model for radio wave propagation over very small distances is a topic that lies beyond the intended scope of this paper (and well beyond our expertise). Here we only give a condition that looks plausible to us, and guarantees our synchronization techniques work. The final word on this issue will have to come from carefully designed radio wave propagation experiments.

Furthermore, we make the following definitions:

- The synchronization phase is the period of time during which the pulses are transmitted. Pulses are only transmitted periodically and in the time outside the synchronization phase, nodes can be dedicated to other tasks. Thus, the smaller the synchronization phase, the better. The duration of the synchronization phase is \( 2\tau \), where \( \tau \) is defined below.

- Zero-crossing is defined for signals that have a positive amplitude and then transition to a negative amplitude. It is the time that the signal first reaches zero.

- The pulse receive time for a node is defined as the time when the observed waveform first makes a zero-crossing.

- The basic waveform transmitted by nodes in the network, \( p'(t) \), is defined as

\[
p'(t) = \begin{cases} 
1 & -\tau < t < 0 \\
0 & t = 0 \\
-1 & 0 < t < \tau 
\end{cases}
\]
where $\tau$ is a constant and defined in terms of $c_i$ for node $i$. $\tau^{-c_1}$ should be chosen large compared to $\max_j \sigma_j^2 e^{c_1}$, where $\sigma_j^2 e^{c_1}$ is the value of $\sigma^2$ translated from the time scale of $c_j$ to $c_1$, i.e. $\sigma_j^2 e^{c_1} \ll \tau^{-c_1}$. This way, over the synchronization phase, with high probability a zero-crossing will occur. The actual value of $\tau$ will determine how much time is spent in the synchronization phase. Thus, we also define $p_j(t)$ as the pulse transmitted by node $j$ in terms of $c_1$.

- $T_j$ is the random variable offset of a transmitted pulse waveform by the $j$th node, i.e. $p_j(t - T_j)$. For example, if node $j$ was in the broadcast domain of node 1 then $T_j = \tilde{y}_{m+1}^j$, where $\tilde{y}_{m+1}^j$ is from section 4.2. We use $T_j$ to simplify notation and stress that this is the estimate of one particular $j$th node. The $T_j$'s are independent since the random jitter, $\tilde{y}_j(t)$, at each node $j$ is independent. We say $T_j$ has the properties

$$E(T_j) = \mu$$
$$Pr(T_j < \mu) = Pr(T_j > \mu) = 0.5$$

for all $j$. We will use these properties later in our proof of Lemma (4.2). Note that if node $j$ were in the broadcast domain of node 1, then $T_j = \tilde{y}_{m+1}^j$ and from section 4.2 we know that $T_j$ satisfies these properties. Using this fact, in section 4.3 we justify that, with our synchronization mechanism, $T_j$ satisfies these properties even when node $j$ is not in the broadcast domain of node 1. In our problem, $\mu$ is the ideal offset and can be interpreted as the time, with respect to $c_1$, that $s_i(t)$ increments.

**Main Result**

An optimally synchronized network would be possible if every node in the network, no matter its distance from node 1, could somehow hear the synchronization pulses emitted by node 1. In this section, we show that this is possible by proving that in the limit as $N \to \infty$, the aggregate waveform seen by any node in the network, will have its zero-crossing at the time of node 1’s synchronization pulse. This means that the aggregate waveform observed by any node $i$ in the network effectively allows node $i$ to observe the synchronization pulses of node 1. We develop the proof focusing on one single synchronization phase. The results, however, apply to any arbitrary synchronization phase.

For simplicity, for the proof we use $p(t)$ defined as

$$p(t) = \begin{cases} 1 & t < 0 \\ 0 & t = 0 \\ -1 & t > 0 \end{cases}$$

instead of $p(t)$. We make this approximation since $\tau$ is chosen such that $\sigma^2 \ll \tau$. This means that with very low probability will $|T_j - \mu| > \tau$. Note that $p_j(t) = p(t)$ since there is no $\tau$ parameter.

The waveform seen by node $i$ is the sum of the waveforms transmitted by all nodes around $i$. Since the shape of the waveform is of particular interest, we keep the magnitude bounded away from zero and infinity by scaling the transmitted amplitude by the number of nodes that contribute to the aggregate waveform at node $i$. This is important since if the magnitude becomes zero, then we can not derive any useful information from the zero crossing. If the magnitude grows unbounded, then that means the network is emitting infinite power. Also, by keeping the magnitude of the aggregate waveform constant, the power per node will go to zero, making sense for an infinitely dense network.

Thus, we first determine the number of nodes that transmit at each step of the synchronization processes. We define the number of successive broadcasts, $k$, that have occurred at a point in time during the initial synchronization period as the value of $q$ received by the set of nodes currently farthest away from node 1 and hearing pulses, with $q$ as defined in section 3. For example, if node 1 and the $R_2$ nodes are transmitting and the $R_3$ nodes can hear, then there have been $k = 2$ successive broadcasts. Note that after $k$ successive broadcasts, the $R_{k+1}$ nodes can here synchronization pulses. We define $\eta_k$ as the set of all nodes that hear transmissions after $k$ successive broadcasts. That is, we call node 1 the starting node and call the set of nodes in its broadcast domain, including itself, $\eta_1$. $\eta_2$ will be the set of nodes that can hear the transmissions of $\eta_1$. This implies that $\eta_1 \subset \eta_2$. In general, $\eta_m$ is the set of nodes that will hear the transmissions from $\eta_{m-1}$. We can also write $\eta_k$ as $\eta_k = (\bigcup_{j=1}^{k+1} R_j)$, with $R_1$ defined as the set with only node 1.

**Lemma 4.1.** Under the above assumptions, the number of nodes that can hear a signal after $k$ successive broadcasts is

$$\eta_k = (\log(N))^\sum_{j=0}^{k-1} (\frac{1}{2})^j = (\log(N))\frac{1-(\frac{1}{2})^k}{1 - \frac{1}{2}}$$

**Proof Outline for Lemma 4.1** By assumption, $K$ nodes transmitting in a circular area centered at node 1 can be heard at a distance of $K^{1/\alpha} d_N$. We define $\eta_k$ as the number of nodes that can hear a signal after $k$ successive transmissions. We define $\eta_0 = 1$. Recalling that we are considering $N$ nodes uniformly distributed over a unit area, the average number of nodes in a circle of radius $R$ is $\pi R^2 N$. Using this, we see that $\eta_1 = \log(N)$. We prove the desired result by induction. This completes the proof outline for Lemma (4.1).

Using Lemma (4.1), we will be able to prove that the aggregate waveform seen by a node $i$ will have its zero-crossing at $\delta_{n+1}^i$, the time that node 1 increments its operational clock with respect to the clock of node $i$. We call the aggregate waveform seen by a node $i$ in terms of $c_1$, $r_i(t)$.

**Lemma 4.2.** Under the assumptions presented above, as $N \to \infty$, $r_i^c(t)$ is continuous with $r_i^c(t = \mu) \to 0$ and $r_i^c(t \neq \mu) \to G$, where $G > 0$ for $t < \mu$ and $G < 0$ for $t > \mu$.

**Proof Outline for Lemma 4.2** For any $\eta_k$, from the Lemma we know that a node in $\eta_{k+1}$ will hear on the order of $\eta_k = (\log(N))^\sum_{j=0}^{k-1} (\frac{1}{2})^j$ nodes. Since, as mentioned earlier in the section, we want to keep the magnitude bounded away from zero and infinity, nodes in $\eta_k$ will transmit a signal magnitude of $\frac{1}{\eta_k}$, where $A$ is the amplitude transmitted by any node $i$ when only it is transmitting. When $\eta_K \geq N$, then all nodes will hear on the order of $N$ nodes. In this case, we define $\eta_{K'} = N$ and $\eta_{K'+1} = N$.

The signal seen at a node $j$ in $\eta_{k+1}$ will be a sum of the signals transmitted from all nodes in $\eta_k$. This will take on the form $r_i^c(t) = \sum_{j=1}^{\eta_k} \frac{A}{\eta_k} p(t - T_j)$. We first realize that the value of $r_i^c(t_o)$ at any time $t_o$ is

$$r_i^c(t_o) = \frac{A}{\eta_k} \text{(number of nodes that transition after } t_o \text{)}$$

where we see that $r_i^c(t_o)$ is going to be random variable since the number of nodes that transition before and after $t_o$ is random. If we define the random variable

$$I_{J,t_o} = \begin{cases} 1 & \text{if } T_J > t_o \\ -1 & \text{if } T_J \leq t_o \end{cases}$$
then \( r_i^\mu(t_o) \) can be written as \( r_i^\mu(t_o) = \sum_{j=1}^{\eta_k} \frac{A_{ij,\mu}}{\eta_k} \) where \( I_{j,t_o} \in \{-1,1\} \) since those are the possible amplitudes of \( p(t) \).

For \( t = \mu \), we know that all \( I_{j,\mu} \) are i.i.d. with \( \Pr(I_{j,\mu} = 1) = \Pr(I_{j,\mu} = -1) = 0.5 \) since \( \Pr(T_j < \mu) = \Pr(T_j > \mu) = 0.5 \) by assumption. As a result, by the strong law of large numbers,

\[
\lim_{N \to \infty} r_i^\mu(\mu) = \lim_{N \to \infty} \sum_{j=1}^{\eta_k} \frac{A_{ij,\mu}}{\eta_k} = AE(I_{j,\mu}) = 0.
\]

This is true for all \( k \) since for all \( k \), \( \eta_k \to \infty \) as \( N \to \infty \).

For \( t = t_o \neq \mu \), we see that all the \( I_{j,t_o} \) are independent but not identically distributed. However, we can show from Resnick [13] that as \( N \to \infty \), \( \sum_{j=1}^{\eta_k} \frac{A_{ij,\mu}}{\eta_k} \) converges to the average value of the means of \( I_{j,t_o} \). For any \( t_o \neq \mu \), we know that \( E(I_{j,t_o}) \leq 0 \) for all \( j \) since \( \Pr(I_{j,t_o} = 1) \leq \Pr(I_{j,t_o} = -1) \). Since \( E(I_{j,t_o}) \) is bounded away from zero for all \( j \) given \( t_o \neq \mu \), the average value of the means will be non-zero.

For continuity, we show that for any \( m_1 \geq 0 \) and \( t_o \) there exists an \( n_1 \) such that \(|t - t_o| < 1/n_1\) implies that \(|r_i^\mu(t) - r_i^\mu(t_o)| < 1/m_1\).

This concludes the proof outline for Lemma (4.2). \( \triangle \)

It is very important to note that the result of theorem (4.2) does not only hold for the general assumptions made in this section, but holds for specific path loss models. For example, if we model the power at a distance \( d \) from a transmitting with power \( P \) as \( \min\{P, \frac{P}{d^2}\} \). We can show that the result still holds. The proof is not included due to a lack of space, but is available from the authors.

**Globally Optimal Synchronization**

The result of Lemma (4.2) has very significant implications for synchronization. First note that the fact that as \( N \to \infty \), \( r_i^\mu(\mu) \to 0 \) implies that \( r_i^\mu(t) \to 0 \) as \( \mu \to 0 \). Now following our synchronization mechanism outlined in section 3, we know that after node 1 starts transmitting synchronization pulses, \( T_j \) for any node \( j \) in \( R_2 \) will satisfy the properties of (10) since node \( j \) is in the broadcast domain of node 1. Thus, we can apply Lemma (4.2) and any node \( k \) in \( R_2 \cup R_3 \) will see a received signal \( r_i^\mu(\mu) = 0 \) for \( N \to \infty \). Since a node \( i \) in \( R_3 \) can effectively see the exact time the pulse from node 1 makes a zero-crossing, its estimate \( c_{t_i,i} \) will have a minimum mean squared error equal to if it was in the broadcast domain of node 1, its minimum achievable mean squared error over the area of the network. This is why the only optimal estimator needed was the estimator outlined in section 4.2 for nodes in the broadcast domain of node 1. Now because the estimate made by node 1 in \( R_3 \) is the same as if it would have made in the broadcast domain of node 1, \( T_i \) for node \( i \) again satisfies the properties of (10). Hence, again Lemma (4.2) can be applied. This cycle will then repeat until all nodes in the network are synchronized and then the cycle will continue to keep the nodes synchronized. It is important to note that with this dense network, the network will be **optimally synchronized** at each step of the synchronization process since every node has access to the transition times of node 1.

### 4.4 Read-World Issues

**Complexity**

If we consider the computational complexity of the optimal estimator described in section 4.2, we realize that each of the nodes would have to compute, in real-time, \( 2m \) multiplications and \( 2(m - 1) \) additions to calculate \( \hat{\theta}_{M_L} \). Depending on the actual synchronization protocol that is implemented, these computations may prove to be a burden for the tiny, low-power nodes for which these synchronization results most apply. This computational burden can be addressed through protocol design or the use of a suboptimal estimator. Suboptimal estimation is addressed later in this section. It is important to note that making the optimal estimate is the most computationally demanding task.

**Large but Finite Networks**

In real-world networks, we will only have a large, but finite, number of nodes in the network. This means that the result of Lemma (4.2) will only be closely approximated. The observed zero-crossings will not be at exactly \( \mu \), but somewhere close to it. This implies that the optimal estimates of the ideal clock will have a variance that increases with the number of successive broadcasts. This, however, can be compensated for.

First, note that we can find the value of \( k \) needed for all \( N \) nodes in the network to hear a signal transmission by setting \( \eta_k \geq N \). After simplifying we get

\[
k \geq \frac{1}{\log(\frac{2}{\alpha})} \log\left(\left(\frac{2}{\alpha} - 1\right) \log(\log(N)) + 1\right).
\]

the desired result.

We notice that \( k \) grows extremely slowly with \( N \). This means that even with a very large \( N \), the number of successive broadcasts that are required to synchronize the entire network is very small. As a result, at each successive broadcast, we can increase the number of observations taken by nodes. For example, if the \( R_3 \) nodes make \( m' \) observations before transmitting synchronization pulses, we can have the \( R_4 \) nodes make \( m'' \) observations, where \( m'' > m' \), before transmitting synchronization pulses. This way, in a practical manner, we can aim to keep the estimation variance the same by increasing the number of observations at each step.

**Suboptimal Estimation**

As mentioned in section 4.4, the optimal estimator is quite computationally expensive. Due to the computational complexity, we look for an unbiased suboptimal estimator of less computational complexity. The suboptimal estimator should be unbiased because of (3) and the fact that Lemma (4.2) requires the mean of the estimate to be the time \( s_1(t) \) increments. As well, the distribution of the estimate must have the property that it is symmetric about the mean. More precisely, if \( X \) is the estimate, then \( \Pr(X < E(X)) = \Pr(X > E(X)) = 0.5 \).

Using notation from (6), we consider the estimator

\[
\hat{g}(y_1, \ldots, y_m) = \frac{2}{m-1} (1 - \frac{m-1}{2} y_1 + \sum_{r=2}^{m} y_r).
\]

To see that the estimator is unbiased,

\[
E(\hat{g}(y_1, \ldots, y_m)) = \frac{2}{m-1} \left( \frac{m-1}{2} \theta_1 + (m-1) \theta_1 \right) + \frac{1}{2} \left( m(m-1) \theta_2 \right)
\]

\[= \theta_1 + m \theta_2.
\]

This estimator only requires two multiplications and \( m - 1 \) additions. However, this estimator is clearly suboptimal and this can be easily seen when we consider the variance. We can easily calculate
the variance of $\hat{g}$ by using the fact that $\hat{g}(Y) = GY$ where $G$ is a row vector of length $n$ and

$$
G = \frac{2}{m-1} \left[ \begin{array}{cccc} (m-1) & 2 & 1 & 1 & \ldots & 1 \end{array} \right].
$$

$\hat{g}$ is a linear transformation of $Y$, so we know that $\Sigma\hat{g} = \sigma^2 = G \Sigma G^T = \sigma^2(1 + \frac{4}{m-1})$. Since the optimal estimate $\hat{\phi}_{ML}$ has

$$
\text{Var}(\hat{\phi}_{ML}) = 2\sigma^2(1 + \frac{4}{m-1}) - 1\right),
$$

we see that as $m \to \infty$, $\sigma_{\hat{g}}^2 \to \sigma^2$ while

$$
\text{Var}(\hat{\phi}_{ML}) \to 0.
$$

This means that as $m \to \infty$, $\hat{\gamma}_{m+1}$ will have a variance of $\sigma^2$ with the optimal estimator, but a variance of $2\sigma^2$ with the suboptimal estimator.

5. RELATED WORK

Many of the challenges and issues facing the synchronization of wireless sensor networks were outlined in [3]. Following this, Reference-Broadcast Synchronization (RBS) was proposed by Elson, Girod, and Estrin [2] to address many of these issues. RBS achieves fine-grained synchronization by having nodes periodically broadcast reference messages to receivers in its broadcast domain. A reference message transmitted by a transmitting node will be received by all nodes in its broadcast domain and these receiving nodes will time stamp the arrival of the reference packet. Using a series of reference packet receive-time differences between two nodes, RBS can do a least-squares linear regression and estimate each node’s clock skew and offset relative to the other node. When two nodes are far apart and are not in a single broadcast domain of any one node, their relative clock differences can be resolved by making a series of time scale conversions.

Even though both our work and RBS uses signals broadcast by nodes to achieve synchronization, the similarities end there:

- Our synchronization protocol is targeted specifically for high-density networks, while RBS is designed for networks with fewer nodes. At high densities, our mechanism requires fewer packet exchanges than RBS to achieve and maintain synchronization of the network. In our protocol, packet exchanges are needed only to achieve initial synchronization. After all nodes are synchronized, synchronization pulses are all that is required to maintain synchronization. In RBS however, every time all nodes want to synchronize, packet exchanges are required to compare the reference pulse receive times. For large numbers of nodes, a large number of packet exchanges would be needed. As well, with RBS the synchronization error increases as the number of hops between two nodes increases. Thus given a finite area, the synchronization error will vary greatly over the area of the network. In our proposed synchronization method, synchronization accuracy actually increases as the network becomes very dense.

- The two synchronization protocols are best suited for different applications. For certain applications, the RBS architecture benefits from post-facto synchronization, where clocks run unsynchronized until events need to be compared. However, for other applications like beam-forming [20] or cooperative transmission [7], a large number of packets may need to be exchanged because the entire network must remain synchronized for extended periods of time. Our method is better suited for these latter applications, since at high densities, our network would remain synchronized using very few transmissions.

- Finally, the fundamental concept of how synchronization is achieved is different in the two methods. Our method synchronizes nodes to the synchronization pulse transmitters, while RBS has the nodes that receive a reference broadcast synchronize with each other.

In other related work, Karp, Elson, Estrin, and Shenker [9] derive an optimal pairwise and globally consistent estimator for pairwise synchronization procedures with independent errors. Their estimator is for pairwise synchronization procedures and thus has important differences compared to our optimal estimator:

- Even though both optimal estimators are minimum variance unbiased and maximum likelihood estimators, they seek to estimate different parameters. In [9], they develop an estimator that seeks to give a set of optimal and globally consistent estimates of the time difference between any two nodes in the network. With such an estimator, they can achieve optimal synchronization in the sense that any two nodes will know their relative clock offset with minimum mean squared error. Our estimator, instead gives an optimal estimate of the clock to which all nodes in the network are trying to synchronize. With such an estimator, we can achieve optimal synchronization in the sense that nodes know the clock to which they want to synchronize with minimum mean squared error.

- In the development of their optimal estimator, they only use one set of reference broadcast receive time information and do not consider using multiple sets of information to improve estimates. Our optimal estimator is presented as a function of the number of observations that the nodes can take. We also present a closed form expression for the variance as a function of the number of observations.

- Furthermore, in the development of our estimator, we treat clock skew and clock offsets simultaneously and present an optimal estimator that accounts for both. In their treatment, they first develop the optimal estimator assuming only clock offsets and no clock skew. Then they apply the estimator to optimally estimate clock skew using different information.

The germ idea of developing a “distributed beamformer”, based on having all nodes first agree on a common packet of bits to transmit and then coordinating their actions, stems naturally from previous work on the sensor broadcast problem [15, 16, 17, 18]. In that work it was established that, even for sensor networks in which the bad scaling laws of Gupta and Kumar [5] hold, there exist combinations of protocols and codes that allow each node to obtain accurate estimates of the measurements collected by every other node in the network. Therefore, with all nodes “sync-ed” to the same data, the distributed transmit antenna array was a natural candidate for taking advantage of this new functionality.

As well, in recent work that also stems from our prior work on the sensor broadcast problem discussed above, Opportunistic Large Arrays have been proposed by Scaglione and Hong as a way to communicate with a far receiver [14]. Both that work and our proposed synchronization method use the idea that nodes base their decisions of when to transmit a signal on signals that they hear from other nodes, but that is where similarities end. Besides differences of form (e.g., the fact that they consider communication with a far receiver, and we consider time synchronization), perhaps the most substantial one is given by the fact that in their work, no attempt is made at “controlling” the shape of the aggregate waveform; instead, nodes wait until they hear a message and as soon as that happens they fire a signal. In their setup, the aggregate waveform is whatever it is that this mechanism produces—engineering tools proposed in that work for obtaining this waveform involve training
and blind estimation. The focus of our work, and our main contributions, lie precisely in the study of structural properties of this limit waveform as a function of the mechanism used to generate it in the network. All of the properties are obtained by setting up an appropriate distributed estimation problem, where each node forms an estimate of when to fire pulses. None of these issues (distributed estimation, properties of the limit waveform, and their interactions) have been addressed in the context of [14].

6. CONCLUSIONS

In this work, we considered the synchronization of all nodes in a network to the clock of one particular node, which we considered to be the ideal clock. Our main results are:

- We developed the notion of an \textit{optimally synchronized network} and showed that, for $N$ nodes over a fixed finite area, as $N \to \infty$ optimality is possible at reasonable complexity.

- The way to achieve optimal synchronization is by solving a distributed estimation problem. After properly formulating this estimation problem, we showed a distributed algorithm for solving it that produces both uniformly minimum variance and maximum likelihood estimates of the ideal clock.

- A key element of our solution was a proof that the aggregate waveform that results from the transmission of a (properly scaled) pulse $p(t - \psi_i)$ by the $i$-th node, when $\psi_i$ is the result of our distributed estimation process, becomes deterministic and contains all the information needed about the state of the ideal clock.

- We also presented a lower complexity suboptimal estimator.

Our results show that, in the high-density regime, all nodes in the network, regardless of their location, can effectively observe the synchronization signals sent by the node with the ideal clock. This implies that the synchronization precision and accuracy achievable is the same as if all nodes in the network were within the broadcast domain of the node containing the ideal clock.

In terms of future work, there are two main lines we intend to pursue. On the more practical aspects of this problem, we intend to test the performance of the proposed synchronization techniques in a sensor network testbed currently under construction at Cornell, in the context of our design of a distributed beamformer [7]. On the more analytical aspects though, the solution obtained in this work for the synchronization problem raises some very interesting questions worth being explored. Specifically, if it is possible to obtain a limit waveform that contains all the information needed to achieve network-wide synchronization, by changing the formulation of the distributed estimation problem, would it be possible to obtain a strong limit waveform that can be used for communication with a far receiver? It turns out the answer is yes, and these results will be published elsewhere.

7. REFERENCES


