Distributed Clock Synchronization over Wireless Networks: Algorithms and Analysis

Arvind Giridhar and P. R. Kumar

***Abstract*— We analyze the spatial smoothing algorithm of Solis, Borkar and Kumar [1] for clock synchronization over multi-hop wireless networks. In particular, for a model of a random wireless network we show that with high probability the error variance is $O(1)$ as the number of nodes in the network increases. This provides support for the feasibility of time-based computing n large wireless networks. We also provide bounds on the settling time of a distributed algorithm.**

I. Introduction

In a wireless network, nodes must often act in coordinated or synchronized fashion. This is true especially for applications such as mobile target localization and event detection. It is also true for the underlying network tasks such as packet scheduling for medium access control, and sleep scheduling in the case of low duty cycle sensor networks. Some of these applications or tasks require *global* clock synchronization, wherein all nodes in the network are synchronized to a common clock. Such global synchronization is likely to be essential wherever the network as a whole is required to act in coordinated fashion.

For concreteness, we consider two examples. The first is a problem of tracking the trajectory of a moving object using directional sensors [2]. Each sensor has a fixed line-of-sight beam, and can detect the time at which the line is crossed by the moving object. The trajectory must be estimated based on the sensor crossing times noted by all the different sensors. Clearly, the performance of any tracking algorithm is limited by the accuracy of clock synchronization. Another example is in the domain of wireless sensor networks. Consider the case of a habitat monitoring network [3], which is deployed in an infrastructure-less environment for an extended period of time. To conserve power, nodes would need to be operated on very low duty cycles. In such a scenario, it may be necessary to carefully design a timed schedule in which nodes turn on their radios at designated times, transmit to other nodes which are also simultaneously awake, and then go back into sleep mode. Tighter synchronization of wake periods would typically allow greater power savings.

Please address all correspondence to the second author at University of Illinois, CSL, 1308 West Main St, Urbana, IL 61801. Email: prkumar@uiuc.edu, University of Illinois at Urbana-Champaign

This material is based upon work partially supported by DARPA Contract N66001-06-C-2021, AFOSR under Contract No. F49620-02-1-0217, NSF under Contract Nos. NSF CNS 05-19535, ANI 02-21357 and CCR-0325716, USARO under Contract No. DAAD19-01010-465, DARPA/AFOSR under Contract No. F49620-02-1-0325, DARPA under Contact Nos. N00014-0-1-1-0576.

The purpose of a clock synchronization algorithm is to provide correction factors to each node in the network, thus enabling the node to convert its own clock value to that of a unique common global clock. These correction factors could be specified in different ways. The simplest choice would be a single correction factor specifying clock *offset* from the reference clock. Alternatively, offset and *skew*, which measures the relative speed of the node's clock with respect to the reference clock, could be calculated.

There have been a number of methods proposed to obtain estimates of such correction factors or parameters. A well known method of obtaining estimates of clock differences between pair of nodes which can directly communicate is based on exchange of time-stamped packets. Viewing the network as a graph, this corresponds to finding estimates of clock offsets across the *edges* of the graph. These estimates must then be processed by the network to obtain estimates of the clock offsets of all *nodes* from a common reference clock.

One way of carrying out the latter task has been proposed in [4]. The approach consists of designating a certain node's clock as the reference clock, building a tree rooted at that node and spanning the entire graph, and simply adding up the estimated edge differences along the unique path to each node to obtain nodal offset estimates.

Karp, Elson, Estrin and Shenker [5] have considered the problem of minimum variance estimation, particularly for the reference broadcast synchronization scheme of [6], and have shown that it satisfies the transitive property of offsets, i.e., the sum of the optimal estimate of the offsets between the node pairs (i, j) and (j, k) is the optimal estimate of the offset between the node pair (i, k). They have also anlayzed the optimal error variance and related it to the resistance distance in a corresponding graph.

Solis, Borkar and Kumar [1] have proposed an approach, based on the concept of least-squares smoothing, to smooth the set of estimates obtained by a packet exchange procedure. The problem that results can be formulated, under a simple noise model on the edge estimates, as a distributed parameter estimation problem on graphs.

The contribution of this paper is an analysis of this least-squares approach. We provide an alternate proof of the connection between the least-squares optimal set of estimates and electrical resistances in an equivalent resistive network. This allows us to compare the performance of the least-squares optimal estimate with the previously proposed tree based solution. This analysis indicates that the least-squares solution provides a substantially better performance, measured in error variance, for a range of typical network graphs. In particular for a model of a random wireless network that is connected, we show that the error variance is $O(1)$ as the number of nodes increases to infinity, with high probability. This provides support for the feasibility of time-based computing in large wireless networks.

We also analyze the convergence properties of the distributed synchronization algorithm proposed in [1], which seeks to converge to the least-squares optimal vector of estimates. We prove convergence and establish bounds on convergence time of these algorithms in terms of standard graph parameters.

The rest of the paper is organized as follows. Section 2 describes the model and assumptions. Section 3 describes formulation of the least-squares estimation problem. Section 4 presents and analyzes a distributed algorithm to compute the least-squares estimate. Finally, Section 5 describes the conclusions and proposed extensions.

II. Model and assumptions

The network is modeled as a directed graph of $n+1$ nodes $\{0, 1, 2, \ldots, n\}$, where each edge represents the ability to transmit and receive packets between the corresponding pair of nodes. Given an edge e connecting nodes i and j and oriented from i to j, i is referred to as the head and j as the tail of edge e.

The *incidence matrix* A of a directed graph of $n+1$ nodes and m edges is the $m \times n+1$ matrix with entries given by the following:

$$A_{ij} = \begin{cases} 1 & \text{if node } i \text{ is the head of edge } j \\ -1 & \text{if node } i \text{ is the tail of edge } j \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

For a connected graph, the rank of the incidence matrix is n, or one less than the number of nodes. Thus, deleting any row from the incidence matrix yields a full row rank matrix, which is called the *reduced incidence matrix*. Here, we will work with the $n \times m$ matrix obtained by deleting the row corresponding to the root node 0. For notational convenience, we use A to henceforth signify the reduced incidence matrix.

Each node i has a clock $c_i(t)$, where t represents the reference time variable of node 0, which is the designated reference node, and $c_i(\cdot)$ is some unknown function. A simple first order model of such a function would be

$$c_i(t) := t + o_i. \tag{2}$$

The clock synchronization problem would then consist of finding the *offset* o_i of each node i from the reference node 0.

Alternately, one could employ a two parameter model for the node clock:

$$c_i(t) = d_i t + o_i, \tag{3}$$

where the skew parameter d_i represent the clock drift relative to node 0. In reality, these parameters will be time-varying. Thus, it is necessary to re-compute their values periodically in order to maintain a bounded estimation error. There is an inherent tradeoff here in that computing parameters of a higher order model is more complex, but will ensure that the synchronization remains accurate over a longer timescale.

In this paper, we will focus only on the one parameter offset model. In fact, our approach can be extended to the two parameter drift+offset model as well (see [1] for details). However, the essential technique in estimating each parameter is the same.

III. Least-Squares clock synchronization

We now describe the least-squares synchronization approach for estimating the clock offsets of all nodes from the reference clock. For a detailed description, see [1].

The first step consists of estimating the clock differences across every edge of the network. Recall that edges signify the bidirectional ability to exchange packets. Therefore, each pair of nodes connected by an edge send packets to each other, while noting the time of transmission and the time of reception. The differences between the time of transmission and the time of reception is the sum of the propagation delay, the clock offsets of the respective clocks, and random effects due to uncertainty in noting the exact time of transmission/reception. The effect of propagation delay can be eliminated by subtracting the one-way difference from the reverse-way difference. Further, this process can be repeated to average out the random effects. At the end, we are able to obtain for each edge ij in the network, an estimate

$$\hat{o}_{ij} = o_{ij} + N_{ij}, \tag{4}$$

of the true clock offset o_{ij}. N_{ij} is the sum of random effects, and therefore can be modeled as a normal random variable with mean 0 and variance σ^2. We make the simplifying assumption that this noise is i.i.d. across all the edges of the network.

The second step consists of obtaining nodal offset estimates from these edge offset estimates. Observe that the network graph may in general have several cycles. For each of these cycles, the true edge clock differences must satisfy the natural Kirchhoff-like constraints:

$$\sum_{ij \in L} o_{ij} = 0, \tag{5}$$

for every loop L. That is, the differences must sum to zero around loops.

These constraints may be expressed in matrix form as follows:

$$o = A^t v, \tag{6}$$

where v is some n dimensional vector, and the orientation of the graph is chosen according to the sign of the corresponding edge offsets. Thus, the offset vector o must belong to the range space of the (reduced) incidence matrix A. However, this will in general not be true for the vector of offset estimates $\hat{o}$, because each of its entries are corrupted with random noise. Also note that what we want to compute is the vector v, which is the vector of nodal clock offsets. Thus, we can compute a least-squares fit of the offset estimate vector to the range space of A^t, i.e., the space of offset vectors satisfying the network Kirchhoff constraints. This is given by the following quadratic optimization problem:

$$v^* = arg\min_{v} \parallel A^t v - \hat{o} \parallel \tag{7}$$

Thus, the clock-synchronization problem is effectively reduced to an unconstrained quadratic optimization problem. Under the assumption that the edge errors are i.i.d. normal random variables, this optimization problem is equivalent to determining the maximum-likelihood vector estimate of the true edge offsets.

The solution to (7) has some interesting structural properties, outlined in the following result, observed also in [5].

Theorem 1: Given a connected graph with reduced incidence matrix A,

1) The unique optimal solution of (7) is given by
$$v^* = (AA^t)^{-1} A\hat{o}. \tag{8}$$
2) The error variance $E[v_i^* - v_i]^2$ of node i is equal to σ^2 times the *resistance distance* between node i and the root node in the electrical network defined by replacing each edge in the connectivity graph by a unit resistor.

a) Proof:: The proof of part (1) is fairly trivial, since (7) represents a simple quadratic optimization problem of with objective function $x^t Qx + b$, where $Q = AA^t$. The matrix AA^t is an $n \times n$ matrix, which is the principal sub-matrix of the *Laplacian* matrix of the graph, which is known to be positive semi-definite. AA^t is in fact full rank for connected graphs, and so is a positive definite matrix. Thus, setting the gradient of the objective function to zero yields the unique solution, which is given by (8).

We proceed to part (2). Let v be the true vector of node offsets, and N the noise vector. The error vector is thus given by

$$\begin{aligned} e &= v^* - v &(9)\\ &= (AA^t)^{-1} A(A^t v + N) - v &(10)\\ &= (AA^t)^{-1} AN, &(11) \end{aligned}$$

with covariance $E[ee^t] = \sigma^2 (AA^t)^{-1}$.

The diagonal entries of $E[ee^t]$ are the variances of the errors of the estimated offsets. Thus, part (2) is the statement that the diagonal elements of $(AA^t)^{-1}$ are the resistance distances between the corresponding nodes and the root node in the equivalent electrical network obtained by replacing edges of the connectivity graph by unit resistors. This fact is known in electrical network theory, and can be deduced by computing the voltage values in the network resulting from the addition of a set of current sources between every node in the network and the root. For details see [8], Chapter 2. ■

The benefit of relating the performance of least-squares synchronization to electrical resistance, as in [5], is that we can evaluate this approach for any network graph in which the resistance between two nodes can be computed or bounded. Furthermore, we have a more intuitive basis for understanding how the least-squares synchronization error will depend on the structure of the graph. In particular:

- More edges, i.e., more bilateral estimates, means better estimates. Adding more resistors to a network can only reduce the resistance between any pair of points. In particular, this proves that the least-squares based estimates will be better than the tree based estimates.
- Resistance is additive along a path, whereas the inverse of resistance (conductance) is additive over parallel paths. In terms of our estimation problem, having using estimates from parallel paths improves the resulting estimate due to noise averaging, while over a single path, the noise variances add up along the edges of the path.
- When the graph is a tree, resistance between any two nodes is the distance between them. Thus, the error variance will grow linearly with the distance of the node from the root. Note that the same is true for the tree-based synchronization approach of [4], in which the offsets, and consequently the noise variances as well, are added up along the unique path from the root to every node. This similarity is not coincidental, because for a tree graph the least squares solution in fact specializes to the tree-based solution. To see this, note that if the graph is a tree, the reduced incidence matrix is a full rank square matrix (because the number of edges is one less than the total number of nodes, which is $n + 1$). In this case, the offset estimate vector will in fact belong to the range space of A^t, which in effect means that no smoothing can be achieved, and that the least-squares optimal vector can be obtained by adding the edge offsets along paths.

We now evaluate the performance of the least-squares solution on some canonical graphs, using the metric of maximum error variance over all nodes of the graph.

b) Tree:: As mentioned above, the resistance distance from any node to the root is the distance of that node to the root. Thus, the least squares solution specializes to the standard tree based solution of adding up estimates along edges of a tree. The error variance is proportional to the diameter of the graph.

c) Clique:: The resistance distance between any two nodes in a clique of n nodes is $2/n$ (this can by easily derived using Kirchhoff voltage-current calculations). Thus, the error of the least squares solution decays inversely with the size of the network.

d) Lattice:: The resistance between two adjacent points in an infinite lattice is a well known problem in classical electrical circuit theory. Finding the resistance between points in a finite lattice has been addressed in [10]. In particular, it can be shown that the resistance between opposite corners of a square lattice of n nodes is $\Theta(\log n)$. Note that for the tree-based algorithm, the maximum error would be proportional to $\sqrt{n}$.

e) Random Planar Network:: This is a network in which nodes are scattered uniformly at random over some fixed area. It can be shown that if all nodes have a common transmission range, the network graph will be connected if and only if the range is large enough so that the average number of neighbors of each node is $\Omega(\log n)$ [11]. Thus, this network has a higher density of edges than the lattice network. For such a network, it can be shown that the maximum resistance distance is in fact $O(1)$ [12].

These examples indicate that the performance of the least-squares algorithm, measured in terms of maximum error variance, is substantially better than that of the tree-based approach.

IV. A Distributed algorithm based on coordinate descent

As shown in the previous section, the solution to the optimization problem (7) is easily computed analytically. However, it is not clear how such a solution would be computed in practice. In a randomly deployed self-organized network, the knowledge of the graph structure which is reflected in the incidence matrix A, is not likely to be known. Thus, we need a distributed algorithm which relies only on locally available information.

There are many iterative techniques for solving a quadratic optimization problem. One of these is *coordinate descent.* This involves choosing a coordinate i (i.e., a node) at each step and minimizing the objective function $F(v)$ with respect to v_i.

Let $F(v) := \parallel A^t v - \hat{o} \parallel^2$. Then,

$$\frac{dF(v)}{dv_i} = (AA^t)_i v - A_i \hat{o}. \qquad (12)$$

To simplify this, we can exploit the structure of the matrix AA^t. From the definition of the reduced incidence matrix A, it is easy to deduce that AA^t has the following structure:

$$(AA^t)_{ij} = \begin{cases} D_i & \text{if } i = j \\ -1 & \text{if } i \text{ and } j \text{ are neighbors} \\ 0 & \text{otherwise,} \end{cases} \qquad (13)$$

where D_i is the degree of node i. Therefore, (12) can be written as

$$\frac{dF(v)}{dv_i} = D_i v_i - \sum_{j \leftrightarrow i} (v_j + \hat{o}_{ji}). \qquad (14)$$

Setting $\frac{dF(v)}{dv_i} = 0$ leads to the following iteration for node i

$$v_i = \frac{1}{D_i} \sum_{j \leftrightarrow i} (v_j + \hat{o}_{ji}). \qquad (15)$$

This admits a very simple interpretation. Each node computes its updated estimate as the average of all its neighbors estimates plus the corresponding offsets, i.e., the average of its neighbors estimates of *its own estimate.* The advantage of implementing such an iteration is clear: Nodes need only communicate with their neighbors. Thus, no knowledge of global topology is needed, and the iteration is completely decentralized as well as asynchronous.

The following description summarizes the resulting algorithm of [1].

f) Distributed Asynchronous Time Averaging: : This consists of two stages, which run in parallel.

- Each node periodically exchanges timestamped packets with each of it's neighbors and estimates the bilateral offsets.
- Each node periodically transmits its current estimate of its own offset v_i to each of its neighbors. When it receives all its neighbors' current estimates, it updates its own estimate using the update (15).

We also consider the following synchronous version of the above algorithm.

g) Distributed Synchronous Time Averaging: : The bilateral estimates are computed as above. However, each estimate is associated with a sequence number n. Nodes transmit the latest estimates along with the sequence numbers to each of their neighbors. Each node i obtains its $(n+1)st$ estimate $v_i^{(n+1)}$ using the equation

$$v_i^{(n+1)} = \frac{1}{D_i} \sum_{j \leftrightarrow i} (v_j^{(n)} + \hat{o}_{ji}). \qquad (16)$$

We now analyze the convergence properties of the above algorithms. The first result establishes that convergence does in fact occur.

Theorem 2: The synchronous as well as asynchronous versions of the averaging algorithm converge to the optimal least-squares solution v^*.

h) Proof:: We give the proof only for the synchronous version. The asynchronous version is the standard coordinate descent approach, and is well known to converge for quadratic optimization problems.

The synchronous iteration can be written in vector form as the following.

$$v^{(n+1)} = v^{(n)} - D^{-1}(AA^t v^{(n)} - A\hat{o}) \qquad (17)$$

Subtracting v^* from both sides, we get

$$\begin{aligned} \bar{v}^{(n)} &:= v^{(n)} - (AA^t)^{-1} A\hat{o} \\ &= v^{(n+1)} - (AA^t)^{-1} A\hat{o} \\ &= v^{(n)} - D^{-1}(AA^t v^{(n)} - A\hat{o}) - (AA^t)^{-1} A\hat{o} \\ &= (I - D^{-1}AA^t)(v^{(n)} - (AA^t)^{-1} A\hat{o}) \end{aligned}$$

Defining $M := I - D^{-1}AA^t$, we have the equivalent iteration

$$\bar{v}^{(n+1)} = M\bar{v}^{(n)}. \tag{18}$$

Thus, the convergence of the sequence $v^{(n)}$ to v^* is equivalent to the convergence of $\bar{v}^{(n)}$ to 0, which is determined by the matrix M. The necessary and sufficient condition for this to happen is that the spectral radius of M is strictly less than 1.

The structure of the matrix M can be determined by inspection as the following:

$$M_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1/D_i & \text{if } i \neq j \end{cases} \tag{19}$$

The row sums of M are all equal to 1, except for the nodes which are adjacent to the root node. For each such root neighbor i, the corresponding row sum is $\frac{D_i - 1}{D_i}$. Thus, the matrix M is both non-negative and sub-stochastic. It follows that $\rho(M) < 1$. ■

The following theorem establishes a bound on the convergence time of the synchronous averaging algorithm.

Theorem 3: Given a graph with reduced incidence matrix A, let $T_\epsilon(A, v^{(0)})$ be the number of iterations of the synchronous averaging algorithm to converge to within a distance ϵ of the optimal solution v^*. Then,

$$\frac{n}{D_0}\log\frac{1}{\epsilon \parallel v^{(0)} \parallel} < T_\epsilon(A, v^{(0)}) < \frac{n^2}{\kappa^2}\log\frac{1}{\epsilon \parallel v^{(0)} \parallel},$$

where D_0 is the degree of the root node and κ is the edge connectivity parameter of the graph.

i) Proof:: We establish bounds on the spectral radius of the matrix Q. The result follows since the number of iterations required for convergence is of the order $\frac{1}{\log 1/\rho(Q)}$.

We map the problem into the Markov chain domain by augmenting the matrix Q to obtain a row-stochastic matrix. Define a matrix P as follows

$$P_{ij} = \begin{cases} M_{ij} & \text{if } 1 \leq i, j \leq n \\ 1/D_i & \text{if } j = n+1, i \leq n, \text{ and } i \leftrightarrow z \\ 1 & \text{if } i = j = N+1 \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

The matrix P can be viewed as accounting for the update equation for the root as well. Thus, the "augmented" iteration which also includes the root is given by

$$\bar{w}^{(n+1)} = P\bar{w}^{(n)}, \tag{21}$$

where

$$w^{(n)} = \begin{pmatrix} v^{(n)} \\ 0 \end{pmatrix}. \tag{22}$$

The states of the resulting Markov chain are the nodes of the network, including the root. From each node i other than the root, the self-transition probability P_{ii} is 0 and the probability of transitioning to each neighbor is $1/D_i$. For the root node 0, $P_{n+1,n+1}$ is 1. Thus, the Markov chain has n states which communicate with each other, and 1 absorbing state. The unique stationary distribution of this chain is $[00\ldots01]$.

We wish to bound the largest eigenvalue of the matrix Q. Now, every eigenvalue of the matrix P is also an eigenvalue of Q. To see this, let λ be an eigenvalue of Q, with eigenvector v. Then, we have

$$P\begin{pmatrix} v \\ 0 \end{pmatrix} = \begin{pmatrix} Mv \\ 0 \end{pmatrix} = \lambda\begin{pmatrix} v \\ 0 \end{pmatrix}. \tag{23}$$

Also, we know from Theorem 2 that the spectral radius of the matrix M is strictly less than 1. It follows that $\rho(Q) \leq |\lambda_2|$, where λ_2 is the second largest magnitude eigenvalue of P. We will bound this quantity.

The second largest eigenvalue of an irreducible aperiodic Markov chain is the parameter that determines its rate of convergence to stationarity. Geometric bounds on the second largest eigenvalue have been derived in [14]. However, the Markov chain in our problem is not irreducible but absorbing.

To make use of the results of [14], we define a new Markov chain with probability transition matrix P^ϵ, where $0 < \epsilon < 1/D_0$. The transition probabilities from all states excepting the root node are the same in P^ϵ and in P. From the root, the probability of transitioning to any neighbor of the root is ϵ, and the probability of self-transition is $1 - D_0\epsilon$. It is clear that the Markov chain defined by P^ϵ is both irreducible and aperiodic. The (unique) stationary distribution is determined by solving the equations

$$\epsilon\pi_0 = \pi_i/D_i, \text{ for every neighbor } i \text{ of the root, and}$$
$$\pi_i/D_i = \pi_j/D_j, \text{for non-root neighboring nodes } i, j.$$

The solution to the above equations is

$$\pi_0 = \frac{1}{1 + \epsilon(\sum_{1\leq j\leq N} D_j)}, \pi_i = \frac{\epsilon D_i}{\sum_{1\leq j\leq N} D_j}, 1 \leq i \leq N.$$

The above calculation also proves that the Markov chain is reversible (i.e., $\pi_i P^\epsilon_{ij} = \pi_j P^\epsilon_{ji}$).

We use Cheeger's inequality [14] to bound λ_2. Define $Q(i,j) = P^\epsilon_{ij}\pi_i = P^\epsilon_{ij}\pi_j$ for any pair of states i and j. Define the following geometric quantity

$$h := \min_{\pi(S)\leq 1/2} \frac{Q(S \times S^c)}{\pi(S)}, \tag{24}$$

where S denotes a set of states, $\pi(S)$ denotes the sum of π_i over all $i \in S$ and $Q(S \times S^c)$ denotes the sum of all Q_{ij} with $i \in S, j \in S^c$. Cheeger's inequality states that

$$1 - 2h \leq \lambda_2 \leq 1 - h^2 \tag{25}$$

To apply this inequality, we need to bound the quantity h. Consider $S = 1, 2, \ldots, n$. Then, the right hand side of (25) is $\frac{\pi_0 D_0 \epsilon}{n\epsilon} = \frac{D_0}{n(1+\epsilon(\sum_{1\leq i\leq n} D_i))}$. This is an upper bound for h. Therefore, we have

$$\lambda_2 \geq 1 - 2\frac{D_0}{n(1 + \epsilon(\sum_{1\leq i\leq n} D_i))} \tag{26}$$

We now derive the upper bound. Observe that the set S cannot contain the root node, as its probability would then exceed $1/2$. Thus, the denominator of the RHS is maximized by $S = 1, 2, \dots, n$. Also, note that for each state, the transition probability for each of its neighbors is equal. Therefore, $Q(i,j) = q$ for every neighbor j of i. Together with the fact that the Markov chain is reversible and irreducible, this implies that $Q(i,j)$ is equal to the constant q for every pair of nodes i and j. Therefore, we have

$$Q(S \times S^c) = q\sigma(Q \times Q^c) \geq q\kappa, \tag{27}$$

where $\sigma(Q \times Q^c)$ is the size of the cut and κ is the minimum value of the cut, i.e., the edge connectivity of the network graph. Thus, we have the upper bound

$$h \geq \frac{\kappa}{n(1 + \epsilon(\sum_{1 \leq i \leq n} D_i))}. \tag{28}$$

Thus, by the upper bound in Cheeger's inequality,

$$\lambda_2 \leq 1 - \left(\frac{\kappa}{n(1 + \epsilon(\sum_{1 \leq i \leq n} D_i))} \right)^2. \tag{29}$$

We now need to map the bounds derived for λ_2^ϵ to bounds on λ_2, the second largest eigenvalue of P^a. But since eigenvalues are continuous functions of the entries of a matrix (see [15],Appendix 1), and the entries of the matrix P^ϵ converge to the entries of P^a as ϵ goes to 0, we have

$$\lim_{\epsilon \to 0} \lambda_2^\epsilon = \lambda_2 \tag{30}$$

Therefore, we have the bounds

$$1 - 2\frac{D_0}{n} \leq \lambda_2 \leq 1 - \left(\frac{\kappa}{n}\right)^2 \tag{31}$$

The result follows by taking logarithms of the above terms. ■

V. Conclusions and Future Work

The contribution of this paper is an analysis of the least-squares based clock-synchronization approach, and distributed synchronization algorithm, which was introduced in [1].

Some planned extensions to this work include designing a sub-optimal algorithm which improves upon the performance of the tree-based approach but is faster than the distributed iterative algorithm described in this paper. Also of interest is extending these results to scenarios involving time varying network topologies.

References

[1] R. Solis, V. Borkar, and P. Kumar, "A new distributed time synchronization protocol for multihop wireless networks," University of Illinois, Tech. Rep., April 2005.

[2] K. Plarre and P. Kumar, "Object tracking by directional sensors," in *Proceedings of IEEE Conference on Decision and Control (CDC)*, Seville, Spain, 2006, pp. 3123–3128.

[3] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications.* ACM Press, 2002, pp. 88–97.

[4] S. Ganeriwal, R. Kumar, and M. Srivastava, "Timingsync protocol for sensor networks," 2003.

[5] R. Karp, J. Elson, D. Estrin and S. Shenker, "Optimal and global time synchronization in sensornets." Tech. Rep., Center for Embedded Networked Sensing, University of California, Los Angeles, April 2003.

[6] J. Elson, L. Girod and D. Estrin, "Fine grained network time synchronization using reference broadcasts." *Proc. Fifth Symposium on Operating Systems Design and Implementation*, Boston, Dec. 2002, pp. 147–163.

[7] M. Sichitiu and C. Veerarittiphan, "Simple, accurate time synchronization for wireless sensor networks," in *IEEE Wireless Communications and Networking Conference(WCNC*, 2003.

[8] L. Chua, C. Desoer, and E. Kuh, *Linear and Nonlinear Circuits.* New York: McGraw Hill Book Company, 1987.

[9] A. Jadbabaie, "On geographical routing without location information," in *Proceedings of IEEE Conference on Decision and Control (CDC)*, Nassau, Bahamas, 2004.

[10] F. Y. Wu, "Theory of resistor networks: The two-point resistance," *JOURNAL OF PHYSICS A*, vol. 37, p. 6653, 2004.

[11] P. Gupta and P. Kumar", "Critical power for asymptotic connectivity in wireless networks," in *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming*, W. McEneaney, G. Yin, and Q. Zhang, Eds., 1998.

[12] A. Giridhar and P. Kumar, "Clock synchronization in wireless networks: Algorithms and analysis," in preparation.

[13] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra.* SIAM, 2000.

[14] P. Diaconis and D. Stroock, "Geometric bounds for eigenvalues of markov chains," *Annals of App. Prob.*, vol. 1, no. 1, pp. 36–61, 1991.

[15] R. Horn and C. R. Johnson, *Matrix Analysis.* Cambridge, UK: Cambridge University Press, 1990.