Corrective Consensus: Converging to the Exact Average

Yin Chen[†] Roberto Tron[*] Andreas Terzis[†] Rene Vidal[*]
yinchen@cs.jhu.edu tron@cis.jhu.edu terzis@cs.jhu.edu rvidal@jhu.edu
Computer Science Department[†] Center for Imaging Science[*]
Johns Hopkins University

***Abstract*— Consensus algorithms provide an elegant distributed way for computing the average of a set of measurements across a sensor network. However, the convergence of the node estimates to the global average depends on the timely and reliable exchange of the measurements to neighboring sensors. These assumptions are violated in practice due to random packet losses, causing the estimated average to be biased. In this paper we present and analyze a practical consensus protocol that overcomes these difficulties and assures convergence to the correct average. Simulation results show that the proposed *corrective consensus* has ten times less overhead to reach the same level of accuracy as the one achieved by a variant of standard consensus that uses retransmissions to (partially) overcome the negative effects of packet losses. In networks with more severe packet loss rates, corrective consensus is more than forty times more accurate than standard consensus that uses retransmissions. More importantly, by continuing to execute the corrective consensus algorithm the estimation error can become *arbitrarily* small.**

I. Introduction

Consider a collection of low-power sensing nodes that form an ad-hoc wireless network, each measuring a quantity of interest, and imagine we are interested in computing the mean of the network's measurements. Collecting the measurements to a central location and disseminating the result to the whole network is often inefficient or infeasible. Instead, averaging *consensus* algorithms [14] provide a fully distributed way to iteratively solve this task. While every node exchanges messages only with its directly connected neighbors, it can be shown that, under mild conditions, these algorithms converge to the correct global average ([24]).

Consensus algorithms form the basis for a large number of distributed algorithms, such as Distributed Hypothesis Testing [13], Distributed Maximum Likelihood Estimation [2], [19], and Distributed Kalman Filtering [11], [18], and have been studied under a wide variety of conditions including networks with undirected or directed links, time varying topology [7], and noisy channels [9], [20].

Nevertheless, standard consensus is not robust to packet losses [4], which are common in low-power wireless networks. The problem is that traditional solutions require symmetric packet exchanges, i.e., if node j receives a packet from node i, then it is assumed that node i will receive a packet from node j. If this does not happen (due to packet losses), consensus algorithms will misbehave and not converge to the correct global average. This adverse effect can be lessened, though not completely eliminated, through reliable transmission schemes that reduce the *effective* link loss. Doing so, however, significantly increases the communication and execution overhead of the algorithm. Moreover, the estimated average can still be far from the correct value, as we show in Section III.

In Section IV we introduce *corrective consensus*, a mechanism that eliminates the consensus error even in the presence of asymmetric link losses. The core idea is to maintain at each node i additional variables ϕ_{ij} that represent the amount of change node i has made to its local state due to the updates from its neighbor j. By periodically exchanging these new variables between neighbors during *corrective iterations*, the nodes have a chance to update their local states and correct the errors caused by lost packets. We prove that this scheme results in almost sure convergence to the correct average by selecting an appropriate number of retransmissions in the standard and corrective iterations.

In Section V we use simulations to compare the convergence properties of corrective and standard consensus algorithms. The results show that in a network with 10 nodes and 80% packet loss, standard consensus may fail to converge to the correct average even with 50 retransmissions. On the other hand, corrective consensus reaches the same order of error at a speed that is more than 10 times faster. Continuing the execution of the corrective consensus reduces the error to zero.

II. Related Work

Rajagopal and Wainwright investigated consensus averaging on graphs whose links are symmetric and polluted by quantization noise [15]. Specifically, they studied consensus algorithms based on damped updates and proved convergence to a Gaussian distribution whose variance depends on eigenvalues of the network's Laplacian graph. We consider networks that suffer from a more severe form of faults, where links can arbitrarily discard packets.

Kar and Moura studied average consensus with random topologies and noisy channels [7]. They proposed two algorithms: A-NC, which averages multiple runs of consensus with a fixed number of iterations, and A-ND, which modifies conventional consensus by forcing the weights to satisfy a persistence condition (slowly decaying to zero). When the channels are without noise, only A-ND assures almost-sure convergence to the correct average as our algorithm does. However, their assumption is that packet losses are symmetric and the modification on the weights increases the number of iterations needed to reach convergence.

Asymmetric packet losses have also been modeled with directed switching network topologies, for instance by Kingston and Beard [8] (which use standard consensus) and Li and Zhang [9] (which use an approach similar to A-ND). In all the cases, however, there is the critical assumption that the network is *balanced* (a precise definition will be given later).

Mehyar et al. derived asynchronous averaging algorithms in packet-switched networks [10]. They implemented and tested their algorithms on Planetlab, a global-scale distributed network overlaid on the public Internet. Their formulation also uses variables ϕ_{ij} to correct errors. Nevertheless, their algorithms assume a reliable bidirectional exchange of packets between any pair of nodes. Furthermore, Internet-connected PC class nodes do not have the resource constraints that wireless sensor nodes face.

In summary, the main problem with existing solutions is that they assume symmetric packet losses or balanced graphs, which is not a realistic assumption in wireless networks. Our work, on the other hand, guarantees almost-sure convergence without this restrictive assumption.

Control systems over wireless networks that drop packets have drawn much attention [6], [21]. Instead of achieving statistical optimality, our work aims to completely eliminate the error caused by packet drops in average consensus algorithms and enable their practical use in wireless networks.

III. Background

Consensus is a distributed iterative algorithm designed for networks of connected devices, such as networks of wireless sensor nodes deployed to measure physical quantities of interest (e.g., temperature). Each node i starts with its own measurement z_i and the consensus algorithm aims to compute $\overline{z} = \frac{1}{N}\sum_{i=1}^{N} z_i$, where N is the number of nodes in the network. In other words, consensus calculates the average of the initial measurements across the whole network. Rather than collecting all the z_i's at a central node to compute the average, each node i maintains a running local estimate of $\overline{z}$, denoted as $x_i(t)$ with t being the iteration counter. $x_i(t)$ is also known as the state variable of node i. During each consensus iteration, nodes exchange their state variables with their neighbors and each node updates its own state based on a weighted average of the state variables it receives. It can then be shown that under certain conditions all $x_i(t)$ eventually converge to $\overline{z}$ [9], [12], [14], [17].

We model the network as a directed graph (digraph) $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the vertex set $\mathcal{V} = \{1, 2, \ldots, N\}$ represents the nodes and the ordered pair $(i, j) \in \mathcal{E}$ is an edge if node j can transmit packets to node i directly (note that this definition is different from a common notation where (i, j) denotes the directed link $i \to j$). Reflecting the realities of packet-switched wireless communication, we assume that link (i, j) may experience random packet losses. For ease of analysis, we further assume that the packet losses are independent, and denote the probability of loss as $1 - p_{ij} \in [0, 1)$ for link (i, j). Said differently, $p_{ij} > 0$ is the Packet Reception Ratio (PRR) of the wireless link $j \to i$, and $p_{ij} = 0$ if $(i, j) \notin \mathcal{E}$. We assume that p_{ij} is time-invariant and that $p_{ij} = p_{ji}$ for all $(i, j) \in \mathcal{E}$ pair, i.e., the PRRs in both directions $(j \to i)$ and $(i \to j)$ are equal. The set of neighboring nodes of node i is denoted as $N_i = \{j : (i, j) \in \mathcal{E}\}$.

Nodes can leverage the broadcast nature of the wireless medium to broadcast their state variables to their neighbors in every iteration of the consensus algorithm. Then, the value of p_{ij} determines the probability of receiving the broadcast packet on the $j \to i$ link. Nodes can perform multiple broadcasts in one iteration to increase the probability of delivering their updates. Then if node j transmits n broadcasts, the probability of successfully delivering at least one copy of the update on the $j \to i$ link is $\hat{p}_{ij} = 1 - (1 - p_{ij})^n$. Here the $\hat{p}_{ij}$ is the effective PRR on the $j \to i$ link.

One can define the network adjacency matrix $A(t)$ of graph $\mathcal{G}$ as $A_{ij}(t) = 1$ if node j successfully transmitted its state value to node i during the t-th iteration and zero otherwise. Furthermore, by definition $A_{ii} = 0$ for all i's. We assume that the network adjacency matrix $A(t)$ is a random matrix whose entries are i.i.d. and stationary Bernoulli variables taking values $\{0, 1\}$. Also, we denote $d_i(t) = \sum_{j=1}^{N} A_{ij}(t)$ as the in-degree of node i in iteration t. The degree matrix $D(t)$ is a diagonal matrix with $D_{ii}(t) = d_i(t)$ and the Laplacian matrix of graph $\mathcal{G}$ is

$$L(t) = D(t) - A(t). \tag{1}$$

The eigenvalues of $L(t)$ are within a disk centered at $\max_{i,t}(d_i(t)) + 0j$ on the complex plane with a radius of $\max_{i,t}(d_i(t))$, due to Gershgorin's theorem [5].

A. Standard Consensus

In the standard Laplacian-based consensus each node i updates its state x_i as

$$x_i(t) = \sum_{j=1}^{N} W_{ij}(t-1)x_j(t-1), \quad x_i(0) = z_i. \tag{2}$$

In vector form, the whole network iterates as

$$x(t) = W(t-1)x(t-1), \quad x(0) = z, \tag{3}$$

where $x(t)$ and z are column vectors in $\mathbf{R}^N$, and $W(t-1) \in \mathbf{R}^{N \times N}$ is the weight matrix used during the t-th iteration. The weight matrix $W(t)$ is defined as

$$W(t) = I - \epsilon L(t), \tag{4}$$

where I is the identity matrix and $\epsilon > 0$ is a small constant. One important property of $W(t)$ is that $W(t)\mathbf{1} = \mathbf{1}$ for any t, where $\mathbf{1}$ is the column vector of all ones. If we select $\epsilon < 1/\max_{i,t}(d_i(t))$, then the eigenvalues of $W(t)$ will be distributed as $1 = \lambda_1 \geq |\lambda_2| \geq \cdots \geq |\lambda_N| \geq 0$.

Let $\mathbf{E}(W)$ be the expectation of $W(t)$, which does not depend on t. Note that when link qualities are symmetric (i.e., $p_{ij} = p_{ji}, \forall (i, j) \in \mathcal{E}$) $\mathbf{E}(W)$ is also symmetric and thus from $\mathbf{E}(W)\mathbf{1} = \mathbf{1}$, it follows that $\mathbf{1}^T\mathbf{E}(W) = \mathbf{1}^T$ also holds. It can be shown that if $\mathcal{G}$ is connected and $\epsilon < 1/\max_{i,t}(d_i(t))$, then $|\lambda_2(\mathbf{E}(W))| < 1$ ([14]). This is an important property because it follows that the update scheme shown in (3) reaches consensus, i.e., $\lim_{t\to\infty} x(t) = \alpha\mathbf{1}$, if $|\lambda_2(\mathbf{E}(W))| < 1$ ([22]).

While $\alpha = \overline{z}$ when the weight matrix $W(t)$ is symmetric (as is the case for undirected graphs [25]), $W(t)$ is a random matrix and is not always *balanced*, i.e., $\exists\ t$ s.t. $\mathbf{1}^T W(t) \neq \mathbf{1}^T$. In this case, consensus will converge to a *biased* value that is not equal to the average of the nodes' initial states [4].

B. Effect of Packet Loss

Link losses affect consensus since packets exchanged between a pair of nodes $\{i, j\}$ can be asymmetrically dropped such that the sum of the states is not preserved. Note that symmetric packet drops do not affect the sum of states.

Acknowledgments and retransmissions intuitively should allow node i to detect asymmetric packet losses and either resend the lost packet to j or discard the packet received from j. In this way nodes i and j would both either update or not update their local states and the sum of states would be preserved. Unfortunately, due to the well-known Two Generals' Problem [1], node i and node j can never be certain that they will take the same action, regardless of how many messages they are willing to exchange. If for once node i updates whereas node j does not, an error could be introduced and consensus will generally not converge to the exact average of the initial states.

Nonetheless, doing retransmissions should effectively reduce the probability that nodes take different actions. However, our simulation results show that the deviation between α and $\overline{z}$ can still be quite significant even with a high number of repeated message exchanges. Figure 1 presents the results of running consensus on a 10-node ring topology in which the PRR of all links is 20%. It is evident from Figure 1(a) that, without retransmissions, errors can easily accumulate. Every node in Figure 1(b) is configured to perform up to 20 retransmissions if necessary. Therefore the effective PRR is approximately 99% on every link. Nevertheless, the converged value still has a nontrivial bias.

The results above indicate that retransmissions can indeed reduce biases in the converged value. However, real-life wireless networks have long links with low PRR ([26]) and thus require a large number of retransmissions to ameliorate the effects of packet losses. Doing so however increases the time necessary to complete each consensus iteration. For example, each iteration in Figure 1(b) takes at least 20 times longer than one iteration in Figure 1(a). Therefore, the total convergence time ends up significantly prolonged in Figure 1(b), despite the fewer number of iterations. The speed of convergence is critical for some type of applications, such as control of unmanned vehicles and surveillance with a camera network. Section V elaborates on convergence speed.

IV. Corrective Consensus

Notice that (2) can be written as

$$x_i(t+1) = x_i(t) + \sum_{j=1}^{N} W_{ij}(t)\big(x_j(t) - x_i(t)\big), \quad x_i(0) = z_i, \tag{5}$$

therefore we can define a new set of variables $\phi_{ij}(t)$ as

$$\phi_{ij}(t+1) = \phi_{ij}(t) + W_{ij}(t)\big(x_j(t) - x_i(t)\big), \quad \phi_{ij}(0) = 0. \tag{6}$$

It can be seen that on each node i, the auxiliary variable ϕ_{ij} represents the amount of change that node i has made to its state variable $x_i(t)$ due to neighbor j. Also, note that keeping ϕ_{ij} and updating it according to (6) do not need any additional message exchange because the nodes already execute (5) at every iteration.

If node i and j always take the same action, as explained in Section III-B, then the changes they make should be symmetric, i.e., $\phi_{ij} = -\phi_{ji}$. Therefore, it is natural to define a new set of variables $\Delta_{ij}(t) = \phi_{ij}(t) + \phi_{ji}(t)$, whereas $\Delta_{ij}(t)$ represents the amount of bias (in the sum of the states) that has accumulated on both directions of the (i, j) link. Note that from (5) and (6) we have the interesting property that

$$x_i(t) = x_i(0) + \textstyle\sum_{j \in N_i} \phi_{ij}(t), \tag{7}$$

from which it follows that

$$\textstyle\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \Delta_{ij}(t) = \mathbf{1}^T x(t) - \mathbf{1}^T x(0). \tag{8}$$

Therefore, one straightforward approach for eliminating the consensus error is to first run consensus according to (5) until $x = \alpha\mathbf{1}$ and then remove the bias by

$$\alpha \leftarrow \alpha - \tfrac{1}{N} \textstyle\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \Delta_{ij}. \tag{9}$$

Unfortunately, (9) requires each node to know $\frac{1}{N} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \Delta_{ij}$, which is a consensus problem by itself. Note that $\phi_{ij}(t)$ is locally stored at node i, and calculating $\Delta_{ij}(t)$ requires nodes i and j to exchange $\phi_{ij}(t)$ and $\phi_{ji}(t)$.

Instead of removing the bias in one step, which is unrealistic, nodes should reduce the error in a distributed and iterative manner: each node i corrects its own state value. Specifically, node i collects ϕ_{ji} $(j \in N_i)$ from its neighbors to calculate the Δ_{ij}'s. Then node i adjusts its state variable $x_i(t)$ using the Δ_{ij}'s, thereby accounting for the errors accumulated on its direct (1-hop) links. After this correction, nodes resume the standard consensus shown in (5) while periodically performing the corrective step described above.

In summary, there are two types of iterations in corrective consensus: *Standard* and *Corrective* iterations.

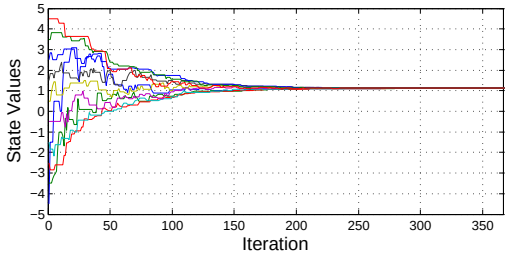
(1) During a *standard iteration*, nodes exchange state values in a best-effort manner and update the values in accordance with (5). Each node performs up to n transmissions to deliver its state variables (i.e., the node performs up to $n-1$ retransmissions). In addition, each node also updates the ϕ_{ij}'s according to (6).

(2) During a *corrective iteration*, nodes exchange ϕ_{ij}'s to calculate the Δ_{ij}'s and use them to adjust their state variables x_i's and auxiliary variables ϕ_{ij}'s. Each node attempts up to m transmissions to deliver the ϕ_{ij}'s.

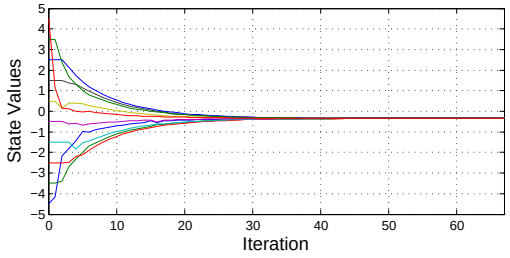
Corrective consensus starts with the standard iterations and after every k consecutive standard iterations, one corrective iteration takes place as follows

$$x_i(k+1) = x_i(k) - \sum_{j=1}^{N} \Delta_{ij}(k)/2 \tag{10}$$

$$\phi_{ij}(k+1) = \phi_{ij}(k) - \Delta_{ij}(k)/2 \tag{11}$$



(a) Standard consensus without any retransmissions.



(b) Each node performs up to 20 retransmissions when necessary.

Fig. 1. Running consensus on a 10-node ring topology in which all links have PRR of 20% and the sum of all initial state values is equal to zero. In both cases, consensus converges to biased estimates (1.12 and -0.34 respectively).

Overall, corrective consensus iterates as follows

$$x_i(t+1) = \begin{cases} x_i(t) - \frac{1}{2}\sum_{j=1}^{N} \Delta_{ij}(t) & \text{if } t = l(k+1)-1 \\ x_i(t) + \sum_{j=1}^{N} W_{ij}(t)\big(x_j(t) - x_i(t)\big) & \text{otherwise} \end{cases}$$

$$\phi_{ij}(t+1) = \begin{cases} \phi_{ij}(t) - \frac{1}{2}\Delta_{ij}(t) & \text{if } t = l(k+1)-1 \\ \phi_{ij}(t) + W_{ij}(t)(x_j(t) - x_i(t)) & \text{otherwise} \end{cases} \tag{12}$$

where $l = 1, 2, 3, \cdots$.

Notice that similar to the x_i's, the ϕ_{ij}'s can be delivered asymmetrically (despite the m transmissions) due to packet losses. Nevertheless, for ease of exposition we assume for now that the ϕ_{ij}'s are reliably delivered and remove this assumption in Section IV-C. It follows from this assumption that $\Delta_{ij}(l(k+1)) = 0$, $\forall i, j, l$.

Figure 2 shows two examples of the corrective consensus algorithm, in a 10-node ring topology for which all links have PRR of 20%. The average of the initial state values is zero. Figure 2(a) uses $k = 149$, while Figure 2(b) uses $k = 24$. In Figure 2(a) the states converge to a biased value during the k standard iterations and the biases are then eliminated during each corrective iteration. After two corrective iterations, the state values converge to the correct average of the initial states. In Figure 2(b) k is not large enough for the states to converge within each k standard iterations. Therefore, the state value curves appear to be much smoother. It is also evident from Figure 2(a) that selecting an overly large k can lead to unnecessary iterations, which waste resources and delay convergence.

A. Basic Properties

Based on the previous discussion, it is easy to see that

$$\begin{aligned} x(k+1) &= x(k) - \frac{1}{2}\begin{bmatrix} \sum_{j=1}^{N} \Delta_{1j}(k) \\ \vdots \\ \sum_{j=1}^{N} \Delta_{Nj}(k) \end{bmatrix} \\ &= x(0) + \begin{bmatrix} \sum_{j=1}^{N} \phi_{1j}(k) \\ \vdots \\ \sum_{j=1}^{N} \phi_{Nj}(k) \end{bmatrix} - \frac{1}{2}\begin{bmatrix} \sum_{j=1}^{N} \phi_{1j}(k) + \phi_{j1}(k) \\ \vdots \\ \sum_{j=1}^{N} \phi_{Nj}(k) + \phi_{jN}(k) \end{bmatrix} \\ &= x(0) + \frac{1}{2}\begin{bmatrix} \sum_{j=1}^{N} \phi_{1j}(k) - \phi_{j1}(k) \\ \vdots \\ \sum_{j=1}^{N} \phi_{Nj}(k) - \phi_{jN}(k) \end{bmatrix} \\ &= x(0) + \frac{1}{2}\sum_{\substack{1\le j\le N \\ 0\le s<k}} \begin{bmatrix} (W_{1j}(s) + W_{j1}(s))(x_j(s) - x_1(s)) \\ \vdots \\ (W_{Nj}(s) + W_{jN}(s))(x_j(s) - x_N(s)) \end{bmatrix} \\ &= x(0) + \frac{1}{2}\sum_{s=0}^{k-1} \Big[W(s) + W^T(s) \\ &\qquad - \mathbf{diag}\Big(\big(W(s) + W^T(s)\big)\mathbf{1}\Big)\Big] M(s)x(0) \end{aligned} \tag{13}$$

where

$$M(s) = \begin{cases} \prod_{u=0}^{s-1} W(u) & s \ge 1 \\ I & s = 0 \end{cases} \tag{14}$$

and note that $M(s)x(0) = x(s)$.

Let us define

$$\begin{aligned} P(u) = I + \frac{1}{2}\sum_{s=u(k+1)}^{(u+1)(k+1)-2} &\Big[W(s) + W^T(s) \\ &- \mathbf{diag}\Big(\big(W(s) + W^T(s)\big)\mathbf{1}\Big)\Big] M(s) \end{aligned} \tag{15}$$

and it follows from $\Delta_{ij}(u(k+1)) = 0$ that

$$x\big((k+1)(u+1)\big) = P(u)x\big((k+1)u\big) \tag{16}$$

Theorem 1. *$P(u)$ has the following properties*

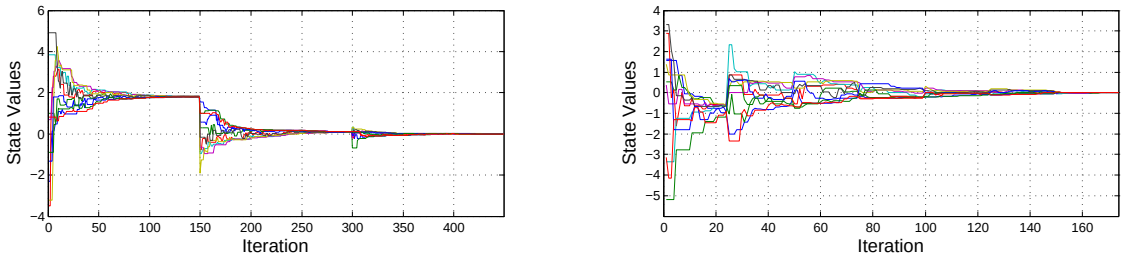
1) $P(u)\mathbf{1} = \mathbf{1}$.
2) $\mathbf{1}^T P(u) = \mathbf{1}^T$.
3) $\mathbf{E}(P(u)) = (\mathbf{E}(W))^k$.

Proof. 1) and 2) are easily verifiable. To show 3), note that $W(s)$ and $M(s)$ are independent, $\mathbf{E}(W(s)) = \mathbf{E}\big(W^T(s)\big) = \mathbf{E}(W)$ and $\mathbf{E}(W)\mathbf{1} = \mathbf{1}$. □

As a direct result, we have

$$\mathbf{E}\big(x\big((k+1)u\big)\big) = (\mathbf{E}(W))^{ku} x(0) \tag{17}$$

which indicates that the expectation of the state values after each corrective iteration will converge to $\mathbf{1}\bar{z}$ because $\mathbf{E}(W)$ is symmetric and $|\lambda_2(\mathbf{E}(W))| < 1$ (cf. §III-A).



(a) $k = 149$. (b) $k = 24$.

Fig. 2. Two examples of corrective consensus. (a) When k is large, state values almost converge before the corrective iterations. After two corrective iterations, state values reach the average of the initial measurements. (b) When k is small, variations across state values are still fairly large when corrective iterations take place. In both cases corrective consensus converges to the correct average value.

B. Convergence Analysis

We start with the observation that if (12) converges to a single point, then the converged point has to satisfy two conditions: $x = \overline{z}\mathbf{1}$ and $\Delta_{ij} = 0, \forall i, j$. Next, it suffices to show that (12) will converge to a single point, which is equivalent to proving $x(t) - \mathbf{1}\mathbf{1}^T x(t)/N \to \mathbf{0}$.

Define

$$\tilde{x}(t) = x(t) - \frac{1}{N}\mathbf{1}\mathbf{1}^T x(t) \tag{18}$$

and our goal is to show that the L2-norm $\|\tilde{x}(t)\| \to 0$.

Theorem 2. *After k consecutive standard iterations, we have*

$$\mathbf{E}\Big(\big\|\tilde{x}\big((k+1)u-1\big)\big\|\Big) \le \overline{\lambda_2}^k \mathbf{E}\Big(\big\|\tilde{x}\big((k+1)(u-1)\big)\big\|\Big)$$

where $\overline{\lambda_2} = \mathbf{E}\big(|\lambda_2(W(t))|\big)$ and $u = 1, 2, 3, \cdots$.

Proof. $\forall t \neq (k+1)u - 1$, $u = 1, 2, \cdots$, we have

$$x(t+1) - \frac{1}{N}\mathbf{1}\mathbf{1}^T x(t) = W(t)\big(x(t) - \frac{1}{N}\mathbf{1}\mathbf{1}^T x(t)\big)$$

because $W(t)\mathbf{1} = \mathbf{1}$, $\forall t \ge 0$. Therefore, we have

$$\|\tilde{x}(t+1)\| \le \|x(t+1) - \frac{1}{N}\mathbf{1}\mathbf{1}^T x(t)\| \le |\lambda_2(W(t))|\|\tilde{x}(t)\|$$

Taking expectations on both sides of the inequality leads to

$$\mathbf{E}(\|\tilde{x}(t+1)\|) \le \mathbf{E}\big(|\lambda_2(W(t))|\big)\mathbf{E}(\|\tilde{x}(t)\|) = \overline{\lambda_2}\mathbf{E}(\|\tilde{x}(t)\|)$$

Therefore, for k consecutive standard iterations

$$\mathbf{E}\Big(\big\|\tilde{x}\big((k+1)u-1\big)\big\|\Big) \le \overline{\lambda_2}^k \mathbf{E}\Big(\big\|\tilde{x}\big((k+1)(u-1)\big)\big\|\Big)$$

□

Theorem 2 shows that $\|\tilde{x}(t)\|$ decreases during the standard iterations. However, it is easy to verify that $\|\tilde{x}(t)\|$ might increase in a corrective iteration. Therefore, we will focus on the state values immediately after each corrective iteration. Denote $y(u) = x\big((k+1)u\big)$, then we are interested in the sequence of $\|\tilde{y}(u)\|$. Here we also denote $\tilde{y}(u) = y(u) - \frac{1}{N}\mathbf{1}\mathbf{1}^T y(u)$.

Theorem 3.

$$\mathbf{E}\big(\|\tilde{y}(u)\|\big) \le \left(\overline{\lambda_2}^k + \frac{1}{2}\epsilon\sqrt{2\tilde{p}N}\frac{1-\overline{\lambda_2}^k}{1-\overline{\lambda_2}}\right)^u \|\tilde{y}(0)\|$$

where $p = \arg\min_{r \in \{\hat{p}_{ij}\}} |r - 0.5|$. We denote $\tilde{p}$ as a shorthand for $2(p - p^2)$ and note that $0 \le \tilde{p} \le 0.5$.

Proof. Proof is omitted in the interest of space, and is available in [3]. □

Define $c = \left(\overline{\lambda_2}^k + \frac{1}{2}\epsilon\sqrt{2\tilde{p}N}\frac{1-\overline{\lambda_2}^k}{1-\overline{\lambda_2}}\right)$, and c is the critical value that determines the speed of convergence.

Theorem 4. *$\|\tilde{y}(u)\|$ almost surely converges to zero if $c < 1$.*

Proof. By Markov's inequality, we have $\forall \delta > 0$

$$\lim_{t\to\infty}\sum_{i=t}^{\infty} P(\|\tilde{y}(i)\| > \delta) \le \lim_{t\to\infty}\sum_{i=t}^{\infty}\frac{\mathbf{E}\big(\|\tilde{y}(i)\|\big)}{\delta}$$

$$\le \lim_{t\to\infty}\sum_{i=t}^{\infty}\frac{\|\tilde{y}(0)\|c^i}{\delta} = \frac{\|\tilde{y}(0)\|}{\delta(1-c)}\lim_{t\to\infty} c^t = 0$$

□

Now, the question is whether c will be smaller than 1. The value of c is collectively controlled by ϵ, $\overline{\lambda_2}$, $\tilde{p}$ and k. Here $\tilde{p}$ depends on the $\hat{p}_{ij}$'s only. $\overline{\lambda_2}$ depends on the expected topology $\mathbf{E}(W)$, the $\hat{p}_{ij}$'s, and ϵ.

Theorem 5. *Critical value c can always be made less than one by employing retransmissions during standard iterations.*

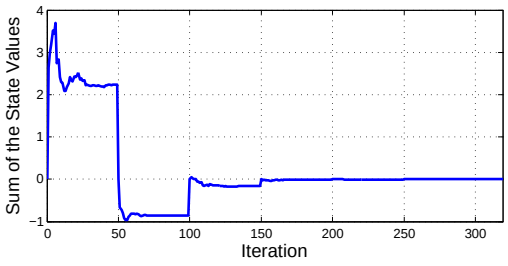
Proof. First, $\overline{\lambda_2} < 1$ if $\mathcal{G}$ is connected. As a result, $\overline{\lambda_2}^k$ goes to 0 as k increases, whereas $\frac{1-\overline{\lambda_2}^k}{1-\overline{\lambda_2}}$ in the second term is upper bounded by $\frac{1}{1-\overline{\lambda_2}}$. Second, $\tilde{p}$ can be made arbitrarily close to 0 by increasing n, which is the number of transmissions in each standard iteration. □

C. Unreliable Exchange of ϕ_{ij}'s

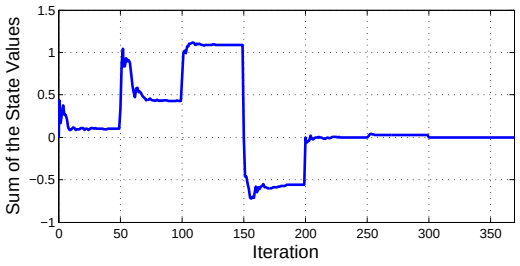
In this section we consider the case that ϕ_{ij}'s can be lost, i.e., $\Delta_{ij}\big((k+1)u\big)$ may be nonzero for $u = 1, 2, 3, \cdots$. As a consequence, the nonzero Δ_{ij} values will propagate to the standard iterations that follow. Therefore, the results obtained from Theorem 3 are not directly applicable.

First, we need to define a set of variables indicating the reception status of ϕ_{ij}:

$$v_{ij}(u) = \begin{cases} 1 & \text{if node } i \text{ receives } \phi_{ji} \text{ from node } j \\ & \text{at the } u\text{-th corrective iteration} \\ 0 & \text{otherwise} \end{cases} \tag{19}$$



(a) ϕ_{ij}'s are reliably delivered in corrective iterations.



(b) ϕ_{ij}'s can be lost in corrective iterations.

Fig. 3. Sum of the state values when running corrective consensus in a 10-node ring topology in which all links have PRR of 80%. $k = 49$ in both cases. (a) The sum of the state values is zero during each corrective iteration. When the ϕ_{ij}'s are not reliably exchanged in (b) the sum is nonzero even during the corrective iterations (e.g., the 50th, 100th and 150th iterations). However, both cases eventually converge to the correct value.

We note here that $v_{ij}(u)$ is a random variable whose distribution depends on p_{ij} and the number of transmissions. For example, $v_{ij}(u)$ is always 1 if $p_{ij} = 1$, because node i receives every packet from j. We denote $q_{ij} = P(v_{ij}(u) = 1)$, $\forall u$. $q_{ij} = 1 - (1 - p_{ij})^m$, where m is the maximum number of transmissions allowed in each corrective iteration.

Next, we can rewrite the corrective consensus scheme as

$$x_i(t+1) = \begin{cases} x_i(t) - \dfrac{1}{2}\displaystyle\sum_{j=1}^{N} \Delta_{ij}(t) v_{ij}(l) & t = l(k+1) - 1 \\ x_i(t) + \displaystyle\sum_{j=1}^{N} W_{ij}(t)\big(x_j(t) - x_i(t)\big) & \text{otherwise} \end{cases}$$

$$\phi_{ij}(t+1) = \begin{cases} \phi_{ij}(t) - \dfrac{1}{2}\Delta_{ij}(t) v_{ij}(l) & t = l(k+1) - 1 \\ \phi_{ij}(t) + W_{ij}(t)(x_j(t) - x_i(t)) & \text{otherwise} \end{cases} \tag{20}$$

where $l = 1, 2, 3, \cdots$.

Figure 3 plots the sum of the state values when running corrective consensus, with the sum of the initial measurements being zero. Figure 3(a) assumes the ϕ_{ij}'s can always be delivered and therefore the sum is always zero in every corrective iteration. On the other hand, ϕ_{ij}'s can be lost in Figure 3(b) and as a result each corrective iteration is not always able to reduce all the bias. In both cases the sums eventually converge to the correct value.

Theorem 6. *Define $q = \min\{q_{ij} : q_{ij} > 0\}$. If there exists b such that $c \le b < 1$ and $\sqrt{(1-q)(1-q/2)} \le \frac{b(b-c)}{2b-c}$, then we have $\mathbf{E}\big(\|\tilde{x}((k+1)u)\|\big) \le cb^{u-1}\|\tilde{x}(0)\|$ for $u \ge 1$.*

Proof. The proof can be done by mathematical induction. It is omitted in the interest of space and is available in [3]. □

Theorem 7. *If $\mathbf{E}\big(\|\tilde{x}((k+1)u)\|\big) \le cb^{u-1}\|\tilde{x}(0)\|$ with $c \le b < 1$, then $\|\tilde{x}((k+1)u)\|$ almost surely converges to zero.*

Proof. Proof is similar to the one used in Theorem 4. □

Theorem 8. *By tuning the number of retransmissions used during the corrective iterations, it is always possible to satisfy $\sqrt{(1-q)(1-q/2)} \le \frac{b(b-c)}{2b-c}$.*

Proof. Define $p = \min\{p_{ij} : p_{ij} > 0\}$, then $q = 1 - (1-p)^m$. By increasing m, one can make $\sqrt{(1-q)(1-q/2)}$ arbitrarily close to 0. Therefore, $\forall b$ such that $c < b < 1$, $\exists$ m such that $\sqrt{(1-q)(1-q/2)} \le \frac{b(b-c)}{2b-c}$. □

Notice that when $q = 1$ (i.e., the ϕ_{ij}'s are never lost), we can choose b such that $b = c$ and the results derived in this section naturally degenerate to the ones presented in Section IV-B.

We also note that while the convergence conditions derived in this section are sufficient, it is not clear whether they are also necessary. Nevertheless, the simulation results in the next section suggest that the corrective consensus algorithms always converge to the correct value when $\mathcal{G}$ is connected. Furthermore, the corrective consensus algorithms in the next section all use $n = 1$, meaning that the nodes simply broadcast their state values once during each standard iteration.

V. Evaluation

In what follows, we first define *convergence* and then evaluate the performance of the proposed corrective consensus algorithm using convergence error and speed as the evaluation metrics.

For the standard consensus, we declare that the algorithm converges during the t-th iteration if the non-increasing quantity $\|\tilde{x}(t)\|$ is less than a threshold κ_1. For the corrective consensus, however, this quantity could increase during a corrective iteration. Therefore, we add one more condition: $|\frac{1}{N}\mathbf{1}^T x(t) - \overline{z}| < \kappa_2$, where κ_2 is another predefined threshold. We declare that corrective consensus reaches convergence at the t-th iteration if both conditions are satisfied.

A. Convergence Error

Let the $\hat{t}$-th iteration be the first iteration that the convergence condition(s) are met. Then, our first metric is the convergence error, defined as $e = |\frac{1}{N}\mathbf{1}^T x(\hat{t}) - \overline{z}|$.

Figure 4 compares the convergence errors for different consensus algorithms that ran in a 10-node ring topology with PRR=20% for every link. The initial states are within $[-10, 10]$ and $\overline{z} = 0$, $\kappa_1 = \kappa_2 = 0.001$. As described in Section III-B, increasing the number of transmissions n can reduce the error of the standard consensus algorithm. When $n = 50$ the standard consensus has comparable error values

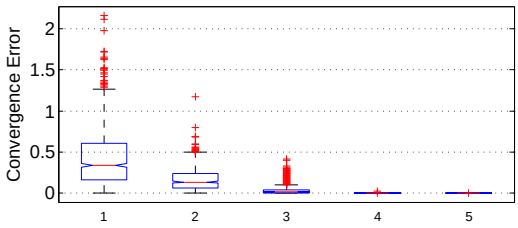


Fig. 4. Box-and-whisker diagram of the convergence errors for standard consensus and corrective consensus. Columns 1-4 are standard consensus with $n = 1, 10, 20, 50$ transmissions per iteration, respectively; column 5 is corrective consensus with $n = 1$, $m = 10$ and $k = 25$. Each column is generated by running 1000 repetitions of experiments.

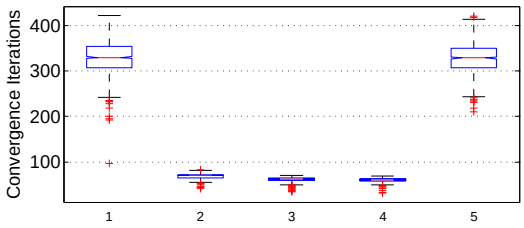


Fig. 5. Number of iterations until convergence. Columns 1-5 are the same as Figure 4.

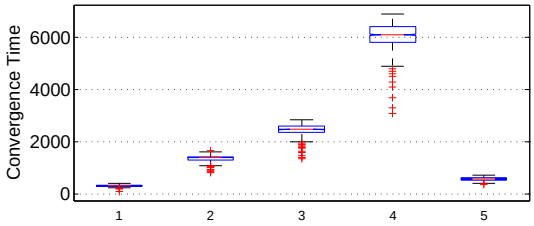


Fig. 6. Convergence time. y-axis is proportional to the actual time. Columns 1-5 are the same as Figure 4.

to the corrective consensus, with the average error being in the order of 10^{-5} for both algorithms. We note that unlike standard consensus which does not guarantee convergence to $\overline{z}$, the convergence error of corrective consensus will reach zero, if not stopped by the convergence conditions.

B. Convergence Speed

Convergence speed can be measured in terms of number of iterations. Generally, fewer iterations translate to shorter time. Nevertheless, as mentioned in Section III-B, the duration of each iteration varies based on the number of retransmissions performed.

In standard consensus, a node only needs to transmit its state value to its neighbors. Due to the nature of wireless communication, the most efficient approach is to broadcast the state value as a packet, allowing all neighbors a chance to receive the update. However, when nodes start to employ the retransmission scheme, broadcasting leads to the ack implosion problem (i.e., the neighbors' acknowledgments can collide at the origin). Not requiring neighbors to send acknowledgments removes this problem. In that case however a node has to blindly broadcast n times in every iteration, regardless of whether the neighbors have received the update. The problem becomes worse when n is large, which is likely the case in real life wireless networks due to the existence of long links with low PRR [26].

For this reason, we decide that when retransmissions are enabled, a node unicasts its state value to each of its neighbors. In turn, every neighbor replies with an acknowledgement packet upon the reception of the state value. The node attempts up to n times to deliver its state value to a neighbor. Also due to the broadcast nature of wireless communication, the other neighbors can overhear the unicast packet and take it as the state value. Therefore, each node actually has in average more than n opportunities of receiving the packet.

Without loss of generality, we assume that the duration of one iteration is $\tau = n\alpha\beta$ for $n > 1$. Here n is the number of transmissions per iteration, α reflects the network density (i.e., the average number of neighbors) and β represents the transmission latency.

In the corrective iteration of the corrective consensus, each node needs to send the ϕ_{ij}'s to its neighbors. Because generally ϕ_{ij}'s are not equal, the node needs to send different packets to its neighbors and it does so by unicast. Therefore, the time of one corrective iteration is $\tau = m\alpha\beta$ in this case, with m being the number of allowed transmissions to deliver each ϕ_{ij}. In practice, the node could potentially put all the ϕ_{ij}'s in one packet and broadcast, if the network density is low. Nevertheless, we do not consider this optimization in the above analysis as it belongs to the implementation details and also depends upon network conditions.

Doing retransmissions can dramatically increase the duration of each iteration. Corrective consensus amortizes the overhead related to retransmissions by performing them once every k standard iterations. Note that in each of the k standard iterations, the corrective consensus could simply broadcast the state value once.

Figure 5 presents the number of iterations necessary for the different consensus algorithms to converge, while Figure 6 shows the actual length of time necessary for the algorithms to converge. The network configurations and parameters are identical to those in Figure 4. The results from Figure 5 are somewhat misleading, suggesting that standard consensus with retransmissions requires fewer iterations and is 'faster' to converge. However, it is evident from Figure 6 that the actual convergence time for standard consensus with retransmissions is significantly longer than the time for corrective consensus, confirming the amortization effect of corrective consensus. In other words, each iteration in corrective consensus is much cheaper (i.e., shorter) than standard consensus with retransmissions.

Combining the results from Figure 4, we know that the standard consensus requires $n = 50$ transmissions to achieve the same error level with corrective consensus. Therefore, one can see from Figure 6 that achieving the same error level by retransmissions increases the execution time by more than tenfold. Last but not least, by comparing columns 1 and 5 in Figure 6, one can see that the overhead introduced by the

	$n=1$	$n=10$	$n=20$	$n=50$	$n=100$	corrective
error	0.1153	0.0407	0.0386	0.0222	0.0208	4.6e-4

TABLE I

CONVERGENCE ERROR IN A 10-NODE RANDOM TOPOLOGY IN WHICH LONG LINKS WITH LOW PRR EXIST.

corrective iterations is marginal compared to that of using even a modest number of retransmissions.

C. Random Topology

Table I lists the performance of various consensus algorithms in a topology formed by randomly placing 10 nodes within a rectangular area. The PRR between each pair of nodes is determined by the log-normal path loss model [16] with parameters experimentally derived from an environmental monitoring sensor network deployed in a forest and the PRR-SNR curve of the 802.15.4 compliant CC2420 radio [23]. In this topology there exist long links that have low PRR. Specifically, several links have PRR below 10%, and the lowest is 3.4%. As a consequence, doing retransmissions alone is hard to eliminate the convergence error. Nevertheless, corrective consensus is able to reduce the error to a very low level. Once again we note that the error is due to the use of the κ_1 and κ_2 thresholds and the error eventually reduces to zero as the corrective consensus algorithm continues to execute.

VI. CONCLUSION

Consensus algorithms constitute a valuable theoretical tool for computing scalar averages across networks of interconnected devices. Unfortunately, existing solutions are impractical when applied to wireless networks that naturally exhibit asymmetric packet losses [26].

In this paper we present a novel *corrective consensus* algorithm that enables the practical use of consensus in real-life sensor networks. Through the addition of new variables at each node and new corrective iterations, we prove that the proposed method converges almost surely to the correct average despite random and asymmetric link losses. Furthermore, we compare the performance of corrective and standard consensus algorithms both in terms of accuracy of the results and convergence speed.

Selecting the optimal interval k for corrective iterations is part of our future work. It is also interesting to investigate the behavior of corrective consensus when the $p_{ij} = p_{ji}$ assumption is removed.

REFERENCES

[1] E. A. Akkoyunlu, K. Ekanadham, and R. V. Huber. Some constraints and tradeoffs in the design of network communications. In *SOSP '75: Proceedings of the fifth ACM symposium on Operating systems principles*, pages 67–74, New York, NY, USA, 1975. ACM.

[2] S. Barbarossa and G. Scutari. Decentralized maximum-likelihood estimation for sensor networks composed of nonlinearly coupled dynamical systems. *Signal Processing, IEEE Transactions on*, 55(7):3456–3470, 2007.

[3] Y. Chen, R. Tron, A. Terzis, and R. Vidal. On corrective consensus: Converging to the exact average. Technical report, Computer Science Department, Johns Hopkins University, Mar 2010.

[4] F. Fagnani and S. Zampieri. Average consensus with packet drop communication. *SIAM J. Control Optim.*, 48(1):102–133, 2009.

[5] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge Univ. Press, 1987.

[6] Z. Jin, V. Gupta, and R. M. Murray. State estimation over packet dropping networks using multiple description coding. *Automatica*, 42(9):1441–1452, 2006.

[7] S. Kar and J. M. F. Moura. Distributed consensus algorithms in sensor networks with imperfect communication: link failures and channel noise. *Trans. Sig. Proc.*, 57(1):355–369, 2009.

[8] D. Kingston and R. Beard. Discrete-time average-consensus under switching network topologies. In *American Control Conference*, page 6 pp., june 2006.

[9] T. Li and J.-F. Zhang. Consensus conditions of multi-agent systems with time-varying topologies and stochastic communication noises. *Automatic Control, IEEE Transactions on*, 55(9), 2010.

[10] M. Mehyar, D. Spanos, J. Pongsajapan, S. Low, and R. Murray. Asynchronous distributed averaging on communication networks. *IEEE/ACM Transactions on Networking*, 15(3):512–520, 2007.

[11] R. Olfati-Saber. Distributed kalman filtering for sensor networks. In *Decision and Control, 2007 46th IEEE Conference on*, pages 5492–5498, Dec. 2007.

[12] R. Olfati-Saber, J. Fax, and R. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, Jan. 2007.

[13] R. Olfati-saber, E. Franco, E. Frazzoli, and J. S. Shamma. Belief consensus and distributed hypothesis testing in sensor networks. In *Network Embedded Sensing and Control. (Proceedings of NESC'05 Worskhop), volume 331 of Lecture Notes in Control and Information Sciences*, pages 169–182. Springer Verlag, 2006.

[14] R. Olfati-Saber and R. Murray. Consensus problems in networks of agents with switching topology and time-delays. *Automatic Control, IEEE Transactions on*, 49(9):1520–1533, Sept. 2004.

[15] R. Rajagopal and M. J. Wainwright. Network-based consensus averaging with general noisy channels. Technical report, Department of Statistics, University of California, Berkeley, May 2008.

[16] T. S. Rappaport. *Wireless Communications: Principles & Practices*. Prentice Hall, 1996.

[17] W. Ren, R. W. Beard, and E. M. Atkins. Information consensus in multivehicle cooperative control. In *IEEE Control Systems Magazine*, 2007.

[18] I. Schizas, G. Giannakis, S. Roumeliotis, and A. Ribeiro. Consensus in ad hoc wsns with noisy links—part ii: Distributed estimation and smoothing of random signals. *Signal Processing, IEEE Transactions on*, 56(4):1650–1666, April 2008.

[19] I. Schizas, A. Ribeiro, and G. Giannakis. Consensus in ad hoc wsns with noisy links—part i: Distributed estimation of deterministic signals. *Signal Processing, IEEE Transactions on*, 56(1):350–364, Jan. 2008.

[20] U. Schmid and C. Fetzer. Randomized asynchronous consensus with imperfect communications. *Reliable Distributed Systems, IEEE Symposium on*, 0:361, 2003.

[21] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry. Kalman filtering with intermittent observations. *IEEE Transactions on Automatic Control*, 49:1453–1464, 2004.

[22] A. Tahbaz-Salehi and A. Jadbabaie. On consensus over random networks. In *44th Annu. Allerton Conf. Commun., Contr. Comput.*, pages 1315–1321, 2006.

[23] Texas Instruments. CC2420: 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver. Available at Available at `http://www.ti.com/lit/gpn/cc2420`, 2006.

[24] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 5, pages 4997–5002 Vol.5, Dec. 2003.

[25] L. Xiao, S. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, page 9, Piscataway, NJ, USA, 2005. IEEE Press.

[26] J. Zhao and R. Govindan. Understanding Packet Delivery Performance In Dense Wireless Sensor Networks. In *Proceedings of the ACM Sensys*, Nov. 2003.