Accelerated Corrective Consensus: Converge to the Exact Average at a Faster Rate

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*— Averaging consensus algorithms provide an elegant, fully distributed, iterative way to compute the average of a set of measurements in a wireless sensor network. Unfortunately, they typically require a large number of iterations to reach convergence. Therefore, a great deal of effort has been devoted into accelerating consensus with improved *accelerated consensus* algorithms. Nevertheless, these techniques assume the communication graph is undirected with fixed or switching topologies, whereas actual low-power wireless networks present random and asymmetric packet losses. As a consequence, these methods might fail to converge to the correct average value when deployed in wireless sensor networks. In this paper we integrate accelerated consensus with *corrective consensus*, a technique that can compute the correct average of the measurements under random packet losses. Our simulation results show that the proposed *accelerated corrective consensus* converges to the correct average, presents a faster convergence rate than *corrective consensus*, and, for a similar number of iterations, it achieves convergence errors about 5,000 times smaller than *accelerated consensus*.**

I. Introduction

Computing the average of a group of quantities measured separately by a network of nodes is an essential building block for numerous applications in various domains such as Distributed Maximum Likelihood Estimation [2], [17], Distributed Hypothesis Testing [13] and Distributed Kalman Filtering [16]. The average consensus algorithm [14] is a popular way to solve this problem iteratively in a fully distributed fashion.

One well documented critique to the consensus algorithm is that it typically requires a large number of iterations to reach convergence. Therefore, significant attention has been recently given to accelerating consensus algorithms [1], [3], [8], [9], [18], [19]. These solutions rely on the results from past iterations to predict a more accurate estimate of the consensus value at each node. Doing so can significantly reduce the number of iterations needed to converge, and in some cases convergence is guaranteed within a finite number of iterations ([8], [18]). Unfortunately, in all these methods the communication links between pairs of nodes are always assumed to be symmetric. Under this assumption, either nodes can communicate in both directions or, when a link fails, neither one can send packets to the other. However, in low-power wireless networks, packets could be independently dropped on each direction of a communication link, resulting in asymmetric network topologies [21]. As a consequence, the consensus algorithm typically converges to a value different from the average of the initial states [6]. As our experiments will show, accelerated consensus is also affected by the random packet drops in a similar way.

In the case of standard consensus, our prior work introduced *corrective consensus* [4], which converges to the correct average even in the presence of asymmetric packet losses. In this paper, we adopt similar techniques to the case of accelerated consensus and show that the resultant *accelerated corrective consensus* converges to the correct value. We will show that it reduces the convergence error by $\sim$5,000 times when compared to *accelerated consensus* in a 10-node ring topology (with both algorithms terminating when the variation across state variables falls below a threshold κ). In addition, accelerated corrective consensus will asymptotically reduce its convergence error down to machine precision.

The rest of this paper is structured as follows. Section II briefly reviews related work, while Section III gives a more detailed description of standard and corrective consensus, as well as the accelerated consensus. In Section IV we integrate corrective iterations to accelerated consensus and demonstrate its convergence to the correct value. Section V presents experiments that evaluate the performance of our *accelerated corrective consensus* algorithm.

II. Related Work

Initial work on accelerated consensus [8], [18], [19] proposed to predict the consensus value at each node from a sufficient number of past values of the local state. Under certain conditions, these consensus algorithms converge in a finite number of iterations. The main drawback is that all these approaches assume the communication graph is undirected with fixed topology. Hence they might fail in the presence of a time-varying topology [9].

To cope with time-varying topologies (i.e., link failures), a consensus algorithm based on adaptive filtering was developed in [3]. This method is proven to converge with varying topologies, and it requires fewer states to be stored at each node. Unfortunately, it provides significant convergence speed gains only in the case of fixed topologies.

In [1], each node performs linear extrapolation from the previous and current state values to predict the next step state value, which in turn will be used in the consensus iteration that follows. The work of [9] uses a similar approach, but it proposes different methods for computing the linear

combination of the past state values. In particular, when the method based on Newton's polynomials is used, this consensus algorithm is able to offer acceleration even under link failures.

As we mentioned earlier, the common assumption shared by all these techniques is that the links between pairs of nodes are always symmetric and thus asymmetric packet losses are not accounted for. Therefore, these algorithms can easily fail when deployed in wireless sensor networks which typically present random and asymmetric packet losses [21].

Our prior work in [4] introduced *corrective consensus*, a consensus algorithm that guarantees convergence to the correct consensus value even in presence of asymmetric packet losses. In this paper we build upon this method by incorporating ideas from accelerated consensus.

III. Background

In this work we consider a group of N nodes (for instance, wireless sensors) organized in a multi-hop wireless network, where each node may communicate only with its one-hop neighbors. We assume that each node i senses a physical quantity, obtaining a value $z_i \in \mathbf{R}$. The goal of the network is then to compute the *consensus value* $\overline{z} = \frac{1}{N}\sum_{i=1}^{N} z_i$.

We model the multi-hop wireless network as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the vertices $\mathcal{V} = \{1, 2, \ldots, N\}$ corresponding to the set of nodes. An ordered pair $(i, j) \in \mathcal{E}$ denotes a directed edge from node j to node i, indicating that node j can directly transmit packets to node i. Packets can occasionally get lost in the wireless channel. For the sake of simplicity, we assume that packet losses are random and independent from each other, and denote as $p_{ij} \in (0, 1]$ the probability of each packet getting through link $(i \leftarrow j)$. Here p_{ij} characterizes the Packet Reception Ratio (PRR) on the wireless link $i \leftarrow j$, and we further assume that $p_{ij} = p_{ji}$, $\forall (i, j) \in \mathcal{E}$. In other words, we are assuming that the PRRs in directions $(i \leftarrow j)$ and $(j \leftarrow i)$ are equal.

The probability of receiving a packet broadcasted from node i to node j is determined by the value of p_{ji}. Nodes can repeatedly broadcast their packets to increase the probability of delivery. For example, if node i broadcasts n times, the probability of delivering at least one copy of the state variable to node j equals to $\hat{p}_{ji} = 1 - (1 - p_{ji})^n$. We define $\hat{p}_{ji}$ as the *effective PRR* on the $j \leftarrow i$ link, whose value can be controlled by the number of repeated broadcasts: n.

The adjacency matrix $A(t)$ of graph $\mathcal{G}$ is defined as $A_{ij}(t) = 1$ if node i received the state variable $x_j(t)$ broadcast by node j during the t-th iteration, and zero otherwise. Note that by definition $A_{ii} = 0$, $\forall i$. We assume that the packet losses are random and independent, therefore $A(t)$ is a random matrix with the entries being stationary i.i.d. $\{0, 1\}$ Bernoulli variables. We define the degree matrix as the diagonal matrix $D(t)$ with $D_{ii}(t) = d_i(t)$, where $d_i(t) = \sum_{j=1}^{N} A_{ij}(t)$ is the in-degree of node i during iteration t. The Laplacian matrix is defined as $L(t) = D(t) - A(t)$. Due to Gershgorin's theorem [7], the eigenvalues of the Laplacian are contained in a disk centered at $\max_{i,t}(d_i(t)) + 0j$ with radius $\max_{i,t}(d_i(t))$ on the complex plane.

A. Standard Consensus

Average consensus [14] computes the average $\overline{z}$ through distributed linear iterations. Specifically, each node i keeps a local *state variable* $x_i(t)$ as its estimate of the consensus value $\overline{z}$, and exchanges the state variable with its one-hop neighbors during each consensus iteration. At the end of an iteration, each node updates its state variable $x_i(t)$ using a weighted average of the state variables from the neighbors. Under certain conditions, $x_i(t)$ asymptotically converges to the consensus value $\overline{z}$ [10], [12], [14], [15]. In more detail, during each consensus iteration, node i will broadcast its state variable $x_i(t)$ to its one-hop neighbors. Then, each node i updates its state variable x_i according to

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

Stacked in vector form, the state variables iteration becomes

$$x(t) = W(t-1)x(t-1), \quad x(0) = z, \tag{2}$$

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 for the t-th iteration. The definition of the weight matrix $W(t)$ is

$$W(t) = I - \epsilon L(t), \tag{3}$$

where I denotes the identity matrix with compatible dimensions and ϵ is a small positive constant. By construction we have $W(t)\mathbf{1} = \mathbf{1}$, where $\mathbf{1} \in \mathbf{R}^N$ is the column vector of all ones.

Denote $\mathbf{E}(W)$ as the expectation of $W(t)$. Note that $\mathbf{E}(W)$ is time-invariant and symmetric due to the assumption that $p_{ij} = p_{ji}, \forall (i, j) \in \mathcal{E}$. As a result, we have both $\mathbf{1}^T\mathbf{E}(W) = \mathbf{1}^T$ and $\mathbf{E}(W)\mathbf{1} = \mathbf{1}$. Provided that $\mathcal{G}$ is connected, it can be shown that if we select the positive constant ϵ such that $\epsilon < 1/\max_{i,t}(d_i(t))$, then the magnitude of the second largest eigenvalue of $\mathbf{E}(W)$ is less than 1, and the consensus iterations shown in (2) asymptotically converges to a common value, i.e., $\lim_{t\to\infty} x(t) = \alpha\mathbf{1}$ ([20]). Unfortunately, due to the random packet losses, $W(t)$ is not always *balanced*, i.e., $\exists\ t$ s.t. $\mathbf{1}^T W(t) \neq \mathbf{1}^T$, hence the converged value α does not equal to the consensus value $\overline{z}$ [6].

B. Corrective Consensus

Corrective consensus is capable of converging to the consensus value $\overline{z}$ even when the weight matrix $W(t)$ is not balanced [4]. To do so, each node i keeps a set of local auxiliary variables $\phi_{ij}(t)$, and updates them as

$$\phi_{ij}(t+1) = \phi_{ij}(t) + W_{ij}(t)\big(x_j(t) - x_i(t)\big), \ \ \phi_{ij}(0) = 0. \tag{4}$$

Notice that (1) can be reformulated into

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

from which it is obvious that $\phi_{ij}(t)$ stands for the amount of change that has been made to the state variable $x_i(t)$, due to neighbor node j.

It follows from (4) that $\phi_{ij} + \phi_{ji} = 0$ if the link between node i and j is undirected, i.e., $W_{ij}(t) = W_{ji}(t)$, for all t. In this case, the two packets transmitted in the two directions are either both received or both lost. With directed links, however, it may happen that $x_i(t)$ was delivered to node j but $x_j(t)$ got lost. In this situation we have $W_{ji}(t) \neq 0$ but $W_{ij}(t) = 0$, and as a result $\phi_{ij} + \phi_{ji} \neq 0$, which indicates that some error in state variables has accumulated due to packet exchanges between node i and node j.

For convenience, we denote $\Delta_{ij}(t) = \phi_{ij}(t) + \phi_{ji}(t)$. In corrective consensus, each node periodically checks $\Delta_{ij}(t)$ to make corrections on its state variable to ensure the convergence towards $\overline{z}$.

Specifically, each node starts with the standard consensus iterations according to (1); after every k such iterations, the nodes will execute a corrective iteration as follows

$$x_i(k+1) = x_i(k) - \sum_{j=1}^{N} \Delta_{ij}(k)/2 \tag{6}$$

$$\phi_{ij}(k+1) = \phi_{ij}(k) - \Delta_{ij}(k)/2 \tag{7}$$

Note that in order to compute Δ_{ij}, node i needs to retrieve ϕ_{ji} from neighbor j. However, packets containing ϕ_{ji} are also subject to link losses and therefore occasionally node i might be unable to compute Δ_{ij} because it does not have access to the value of ϕ_{ji}. In this case, the nodes do not include the affected Δ_{ij}'s in (6) and (7).

To increase the probability of packet delivery, a node will try to send each auxiliary variable repeatedly up to m times. It is shown in [4] that corrective consensus will converge to the consensus value $\overline{z}$ with appropriately selected m and n, where n is the maximum number of repeated broadcasts to deliver each state variable.

C. Accelerated Consensus

As we reviewed in Section II, significant efforts have been devoted into speeding up the convergence of (1). As our baseline *accelerated consensus* algorithm we consider the method proposed in [9], which uses a linear combination of past state values to predict the state value for the next step, and can increase convergence speed under dynamic (undirected) topologies. The accelerated consensus iterations in [9] are defined as

$$x_i(t+1) = \begin{cases} \sum_{l=0}^{K} \beta_{K-l} x_i(t-l), \text{ if } \mathrm{mod}(t+1, K+1) = 0 \\ x_i(t) + \sum_{j=1}^{N} W_{ij}(t)\big(x_j(t) - x_i(t)\big) \text{ otherwise,} \end{cases} \tag{8}$$

where the coefficients $\beta_0 \cdots \beta_K$ satisfy the normalization constraint $\sum_{l=0}^{K} \beta_l = 1$. It can be seen from (8) that accelerated consensus alternates between two types of iterations: *Standard* and *Predictive* iterations. During standard iterations, nodes behave identically as in standard consensus. Every K standard iterations, the nodes execute a predictive iteration during which each node will predict the next step state value based on a linear combination of the past $K+1$ iterations (including the previous predictive iteration).

In [9], the coefficients $\beta_0 \cdots \beta_K$ are set to to minimize the second largest eigenvalue of the weight matrix W. As a result, the accelerated consensus as shown in (8) inherently converges in the case of fixed topology, provided that the weight matrix W satisfies the convergence conditions for standard consensus. However, Kokiopoulou and Frossard showed that (8) might diverge when the topology becomes dynamic [9]. We note that the dynamic topologies in their work are assumed to be undirected at any time instance, i.e., $W(t)$ is symmetric for all t. Therefore, it is unclear how the accelerated consensus algorithm would perform in wireless networks with prevalent random and asymmetric packet losses. Intuitively, accelerated consensus would be more vulnerable to packet losses than standard consensus, due to the predictive iterations (the results in Section V verify this intuition). Hence, in what follows we will integrate the techniques developed for corrective consensus into the accelerated consensus algorithm to enable its application in real-world wireless networks.

IV. Accelerated Corrective Consensus

Recall that in corrective consensus we introduced a set of auxiliary variables ϕ_{ij} to keep track of the changes that are made to the state variables, and the neighbor responsible for each such change. Likewise, we can define auxiliary variables ϕ_{ij} for accelerated consensus and update ϕ_{ij} in the same manner as in (4). Nevertheless, during the predictive iterations, nodes do not exchange state variables with neighbors and therefore it is not immediately clear as to how to appropriately update the auxiliary variables ϕ_{ij}.

However, notice that node i updates its state variable in the predictive iteration based on its own past state values, hence it appears that node i itself should be solely responsible for changing its state variable. Accordingly, we can define ϕ_{ii} to represent the change that node i has made to its own state variable during the predictive iterations. The problem with this approach is that there is no immediate way to examine and rectify the errors. In the case of $\phi_{ij}(t)$, we can check the sum $\Delta_{ij}(t) = \phi_{ij}(t) + \phi_{ji}(t)$, and a value other than zero indicates error. Instead, it is easy to see that if we require the mean of the states to be preserved, we obtain the following network-wide (global) constraint on the ϕ_{ii}:

$$\sum_{i=1}^{N} \phi_{ii} = 0, \tag{9}$$

which can not be readily utilized in a distributed fashion to check and correct the errors. Consequently, we decide not to use ϕ_{ii} but instead to search for the appropriate way of updating the ϕ_{ij} in the predictive iterations. To do so, we first need to define φ_{ij} as

$$\varphi_{ij}(t) = W_{ij}(t)\big(x_j(t) - x_i(t)\big) \tag{10}$$

and decompose each of the past K state variables during the

standard iterations prior to a predictive iteration as follows

$$x_i(t-l) = x_i(t-K) + \sum_{j=1}^{N} \sum_{s=t-K}^{t-l-1} \varphi_{ij}(s), \qquad (11)$$

for $u = 1, 2, 3, \cdots, \forall t = u(K+1) - 1, \quad 0 \leq l < K, i$.

We can see that each state value $x_i(t-l)$ can be written as a linear combination of the previous φ_{ij}'s. Therefore, the predicted value $x_i(t+1)$ in (8) can also be decomposed into linear terms of φ_{ij}'s as

$$\begin{aligned} x_i(t+1) &= \sum_{l=0}^{K} \beta_{K-l} x_i(t-l) = \beta_0 x_i(t-K) \\ &\quad + \sum_{l=0}^{K-1} \beta_{K-l} \left(x_i(t-K) + \sum_{j=1}^{N} \sum_{s=t-K}^{t-l-1} \varphi_{ij}(s) \right) \\ &= x_i(t-K) \sum_{l=0}^{K} \beta_{K-l} + \sum_{j=1}^{N} \sum_{l=0}^{K-1} \sum_{s=t-K}^{t-l-1} \beta_{K-l} \varphi_{ij}(s) \\ &= x_i(t-K) + \sum_{j=1}^{N} \sum_{l=0}^{K-1} \sum_{s=t-K}^{t-l-1} \beta_{K-l} \varphi_{ij}(s) \end{aligned} \qquad (12)$$

where the last line follows from the constraint that $\sum_{l=0}^{K} \beta_l = 1$. Based on (12), we can derive the relationship between $\phi_{ij}(t+1)$ and $\phi_{ij}(t-K)$ as follows

$$\phi_{ij}(t+1) = \phi_{ij}(t-K) + \sum_{l=0}^{K-1} \sum_{s=t-K}^{t-l-1} \beta_{K-l} \varphi_{ij}(s) \qquad (13)$$

(13) provides the rule to update the ϕ_{ij} in the predictive iteration. Moreover, observe that $\phi_{ij}(t+1)$'s value is directly determined by $\phi_{ij}(t-K)$ and $\varphi_{ij}(s)$, hence we do not need to update ϕ_{ij} in the standard iterations. Instead, each node can simply store $\phi_{ij}(t-K)$ and $\varphi_{ij}(s)$ in memory, and update ϕ_{ij} only during each predictive iteration. Note that $\phi_{ij}(t-K)$ is given by the previous predictive iteration.

One problem with this update rule is that holding $\varphi_{ij}(s)$ in memory requires $O(Kd)$ memory space per node where d represents the number of neighbors and K stands for the K standard iterations between two predictive iterations. Memory space is an extremely constrained resource in today's low power wireless sensor network nodes (10kB RAM on one of the most popular platforms [11]). To reduce the memory footprint, observe that (12) can be rewritten as

$$\begin{aligned} x_i(t+1) &= x_i(t-K) + \sum_{j=1}^{N} \sum_{l=0}^{K-1} \sum_{s=t-K}^{t-l-1} \beta_{K-l} \varphi_{ij}(s) \\ &= x_i(t-K) + \sum_{j=1}^{N} \sum_{s=0}^{K-1} \varphi_{ij}(t-K+s) \sum_{l=s+1}^{K} \beta_l \end{aligned} \qquad (14)$$

and hence we can define

$$\beta(t) = \sum_{l=\text{mod}(t,K+1)}^{K} \beta_l \qquad (15)$$

and then write the update rules for the auxiliary variables as

$$\phi_{ij}(t+1) = \begin{cases} \phi_{ij}(t), & \text{if mod}(t+1, K+1) = 0 \\ \phi_{ij}(t) + \beta(t+1)\varphi_{ij}(t), & \text{otherwise,} \end{cases} \qquad (16)$$

Note that coefficients $\beta(t)$ depend only on the $K+1$ parameters $\beta_0 \cdots \beta_K$ and therefore can be precomputed. As a result, updating ϕ_{ij}'s according to (16) does not require additional memory space compared to the standard corrective consensus.

(16) does not update ϕ_{ij} in the predictive iteration, but instead it distributes the due updates among every standard iteration in advance. Hence, during each standard iteration, ϕ_{ij} is updated in a way similar (but not equal) to the actual change to the corresponding state variable. As a result, the property $x_i(t) = x_i(0) + \sum_{j \in N_i} \phi_{ij}(t)$ holds only immediately after a predictive iteration, i.e., when $\text{mod}(t, K+1) = 0$ (note that this property always holds in corrective consensus [4]). Since corrective iterations require this property to check for errors (i.e., whether Δ_{ij} is equal to 0), they can only be executed immediately after a predictive iteration. In practice, this implies that, if we run corrective and predictive iterations, respectively, every k and K standard iterations, it must hold $\text{mod}(k, K) = 0$, $k \geq K$.

For ease of presentation, we group the predictive iteration and the corrective iteration together. Specifically, we denote $x_i^P(t+1)$ as the predicted state value and $x_i(t+1)$ as the corrected state value based on $x_i^P(t+1)$. Then, we write the predictive iteration and the subsequent corrective iteration as

$$\begin{cases} x_i^P(t+1) = \sum_{l=0}^{K} \beta_{K-l} x_i(t-l) \\ x_i(t+1) = x_i^P(t+1) - \frac{1}{2} \sum_{j=1}^{N} \Delta_{ij}(t) v_{ij}(u) \end{cases} \qquad (17)$$

where v_{ij} represents the reception status of $\phi_{ij}(t)$ with u being the counter of the corrective iterations ($v_{ij} = 1$ if ϕ_{ij} was received and 0 otherwise [4]).

Algorithm 1 summarizes our *accelerated corrective consensus* algorithm. We can see that the parameter K controls the frequency of predictive iterations, while the parameter k determines the frequency of corrective iterations, i.e., a corrective iteration takes place after $\frac{k}{K}$ predictive iterations.

Regarding the convergence of our algorithm, we have the following Theorem.

Theorem 1. *Algorithm 1 will converge to the correct average if the original accelerated consensus converges.*

Proof. See the appendix. □

V. Evaluation

We compared four consensus algorithms (standard and accelerated consensus with or without corrective iterations) in a 10-node ring topology with PRR=80% on each one of the 20 directed links. Each node broadcasts the state variable only once during the standard iterations ($n = 1$), but will try up to 10 transmissions to deliver the auxiliary variables ϕ_{ij} to each of the intended neighbor nodes ($m = 10$, see [4] for details).

Algorithm 1 Accelerated Corrective Consensus

Input: constants k, K, function $\beta(\cdot)$
Output: the average value $\overline{z}$ on each node
Initialization: $\phi_{ij}(0) = 0$, $t = 0$, $u = 0$
loop
  if $\text{mod}(t+1, K+1) == 0$ **then** ▷ *run predictive iteration*
    $x_i^P(t+1) = \sum_{l=0}^{K} \beta_{K-l} x_i(t-l)$
    if $\text{mod}(t+1, (K+1)k/K) == 0$ **then** ▷ *run corrective iteration*
      $u \leftarrow u + 1$
      $\Delta_{ij}(t) := \phi_{ij}(t) + \phi_{ji}(t)$
      $x_i(t+1) = x_i^P(t+1) - \frac{1}{2}\sum_{j=1}^{N} \Delta_{ij}(t) v_{ij}(u)$
      $\phi_{ij}(t+1) = \phi_{ij}(t) - \frac{1}{2}\Delta_{ij}(t) v_{ij}(u)$
    else
      $x_i(t+1) = x_i^P(t+1)$
      $\phi_{ij}(t+1) = \phi_{ij}(t)$
  else ▷ *run standard iteration*
    $\varphi_{ij}(t) = W_{ij}(t)\big(x_j(t) - x_i(t)\big)$
    $x_i(t+1) = x_i(t) + \sum_{j=1}^{N} \varphi_{ij}(t)$
    $\phi_{ij}(t+1) = \phi_{ij}(t) + \beta(t+1)\varphi_{ij}(t)$
  $t \leftarrow t + 1$

	A=0,C=0	A=0,C=1	A=1,C=0	A=1,C=1
Convergence Error	0.198	2.607e-5	0.248	4.562e-5
Number of Iterations	77	87	62	69

TABLE I

PERFORMANCE OF FOUR CONSENSUS ALGORITHMS. (A,C) = { (0,0), (0,1), (1,0), (1,1) } REPRESENTS *standard*, *corrective*, *accelerated*, AND *accelerated corrective* CONSENSUS ALGORITHMS RESPECTIVELY.

Figure 1 presents the results, averaged over 100 independent trials. The y-axis plots the convergence error $\|x(t) - \overline{z}\mathbf{1}\|^2$, whose value should decrease to zero as the state variables converge to the consensus value $\overline{z}$. However, both *standard consensus* and *accelerated consensus* fail to converge to the consensus value. Among these two algorithms, accelerated consensus also exhibits larger errors. Our interpretation is that speeding up convergence amplifies the accumulated errors due to asymmetric message exchanges. Nevertheless, after integrating the corrective iterations, both algorithms can converge to the correct consensus value. As we expected, the errors for both corrective algorithms converge to zero (up to machine precision), with *accelerated corrective consensus* presenting a faster convergence rate than *corrective consensus*.

We have also evaluated the algorithms by stopping the iterations when a convergence criterion is met. Here we decide that a consensus algorithm reaches convergence at iteration t if $\|x(t) - \frac{1}{N}\mathbf{1}\mathbf{1}^T x(t)\| < \kappa$, where we set the threshold $\kappa = 0.001$. Table I lists the final convergence error and number of iterations taken. Even in this setting, the error for *accelerated corrective consensus* (A=1,C=1) is about 5,000 times smaller than for *accelerated consensus* (A=1,C=0) for a similar number of iterations (69 versus 62).

VI. CONCLUSION

Accelerated consensus algorithms constitute a valuable tool for quickly and efficiently computing the arithmetic mean across a multi-hop wireless network. However, previous solutions fail to converge to the correct mean in real-world wireless networks due to unavoidable random and asymmetric packet losses.

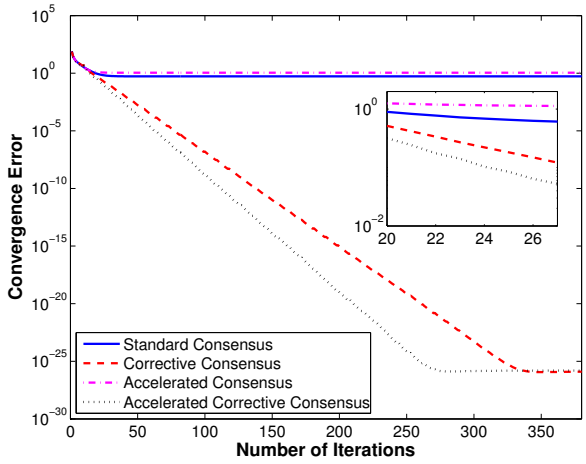

Fig. 1. Convergence Error and Speed of Four Consensus Algorithms.

In this paper we adopt the techniques originally developed in corrective consensus [4] to accelerated consensus, and show that the resultant *accelerated corrective consensus* converges faster to the correct consensus value, even with asymmetric packet drops.

In our future work, we will study methods to optimally select the intervals for predictive iterations (K) and corrective iterations (k).

REFERENCES

[1] T. C. Aysal, B. N. Oreshkin, and M. J. Coates. Accelerated distributed average consensus via localized node state prediction. *Trans. Sig. Proc.*, 57(4):1563–1576, 2009.

[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] R. Cavalcante and B. Mulgrew. Adaptive filter algorithms for accelerated discrete-time consensus. *Signal Processing, IEEE Transactions on*, 58(3):1049 –1058, march 2010.

[4] Y. Chen, R. Tron, A. Terzis, and R. Vidal. Corrective consensus: Converging to the exact average. In *IEEE Conference on Decision and Control*, 2010.

[5] 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.

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

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

[8] E. Kokiopoulou and P. Frossard. Accelerating Distributed Consensus Using Extrapolation. *IEEE Signal Processing Letters*, 14(10):665–668, 2007.

[9] E. Kokiopoulou and P. Frossard. Polynomial filtering for fast convergence in distributed consensus. *Trans. Sig. Proc.*, 57(1):342–354, 2009.

[10] 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.

[11] Moteiv Corporation. Tmote Sky Datasheet. `http://www.moteiv.com/products/docs/tmote-sky-datasheet.pdf`.

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

[16] 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.

[17] 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.

[18] S. Sundaram and C. Hadjicostis. Finite-time distributed consensus in graphs with time-invariant topologies. In *American Control Conference, 2007. ACC '07*, pages 711 –716, 9-13 2007.

[19] S. Sundaram and C. N. Hadjicostis. Distributed consensus and linear functional calculation in networks: an observability perspective. In *IPSN '07*. ACM, 2007.

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

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

APPENDIX

PROOF OF THEOREM 1

Proof. The convergence of the original accelerated consensus indicates that

$$\mathbf{E}\big\|\tilde{x}^P\big((K+1)l\big)\big\| \le \lambda \mathbf{E}\big\|\tilde{x}\big((K+1)(l-1)\big)\big\| \tag{18}$$

where $0 \le \lambda < 1$ determines the speed of convergence, and we define $\tilde{x}(t) = x(t) - \frac{1}{N}\mathbf{1}\mathbf{1}^T x(t)$ as the projection of the vector $x(t)$ onto the hyperplane that is vertical to the vector $\mathbf{1} \in \mathbf{R}^N$.

In what follows we will first consider the case where ϕ_{ij}'s are reliably exchanged. In this case we can follow the proof of Theorem 3 in [5] almost identically except that there would be a weight $\beta(s)$ multiplied to each $\varphi_{ij}(s)$.

Foremost, let us write the update of the state variables in the corrective iteration similar to (26) in [5] as

$$x(k+1) = x^P(k+1) - \frac{1}{2}\sum_{\substack{s=0\\T(s)\neq 0}}^{k-1}\begin{bmatrix}\sum_{i=1}^N \beta(s+1)\delta_{i1}(s)\\ \vdots \\ \sum_{i=1}^N \beta(s+1)\delta_{iN}(s)\end{bmatrix} \tag{19}$$

in which we define $T(s) = \mathrm{mod}(s+1, K+1)$, and $\delta_{ij}(s) = \varphi_{ij}(s) + \varphi_{ji}(s)$. Based on the derived results from (27), (28) and (29) in [5], we have

$$\begin{aligned}&\mathbf{E}\|\tilde{x}(k+1)\|\\ &\le \mathbf{E}\|\tilde{x}^P(k+1)\| + \hat{\beta}\frac{1}{2}\sum_{\substack{s=0\\T(s)\neq 0}}^{k-1}\mathbf{E}\sqrt{\sum_{j=1}^N\Big(\sum_{i=1}^N \delta_{ij}(s)\Big)^2}\end{aligned} \tag{20}$$

where we denote $\hat{\beta} = \max_s(|\beta(s)|)$. Substituting the results from (30), (31) and (32) in [5], we arrive at

$$\mathbf{E}\|\tilde{x}(k+1)\| \le \mathbf{E}\|\tilde{x}^P(k+1)\| + \hat{\beta}\frac{1}{2}\epsilon\sqrt{2\tilde{p}N}\sum_{\substack{s=0\\T(s)\neq 0}}^{k-1}\mathbf{E}\|\tilde{x}(s)\| \tag{21}$$

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

Note that for two successive regular iterations, we have $\mathbf{E}\|\tilde{x}(s)\| \le \overline{\lambda_2}\mathbf{E}\|\tilde{x}(s-1)\|$ where $\overline{\lambda_2} = \mathbf{E}\big(|\lambda_2(W(t))|\big)$ which is the expected second largest eigenvalue of the weight matrix $W(t)$. Combining this (18), we have

$$\begin{aligned}&\sum_{\substack{s=0\\T(s)\neq 0}}^{k-1}\mathbf{E}\|\tilde{x}(s)\|\\ &\le \|\tilde{x}(0)\|\Big\{(1+\overline{\lambda_2}+\overline{\lambda_2}^2+\cdots+\overline{\lambda_2}^{K-1})\\ &\qquad +\lambda(1+\overline{\lambda_2}+\overline{\lambda_2}^2+\cdots+\overline{\lambda_2}^{K-1})\\ &\qquad +\lambda^2(1+\overline{\lambda_2}+\overline{\lambda_2}^2+\cdots+\overline{\lambda_2}^{K-1})\\ &\qquad \cdots\\ &\qquad +\lambda^{k/K-1}(1+\overline{\lambda_2}+\overline{\lambda_2}^2+\cdots+\overline{\lambda_2}^{K-1})\Big\}\\ &= \|\tilde{x}(0)\|\frac{1-\overline{\lambda_2}^K}{1-\overline{\lambda_2}}\frac{1-\lambda^{k/K}}{1-\lambda}\end{aligned} \tag{22}$$

Plugging (22) into (21) and using the fact that $\mathbf{E}\|\tilde{x}^P(k+1)\| \le \lambda^{k/K}\|\tilde{x}(0)\|$ (from (18)), we have

$$\begin{aligned}&\mathbf{E}\|\tilde{x}(k+1)\|\\ &\le \left(\lambda^{k/K} + \frac{1}{2}\hat{\beta}\epsilon\sqrt{2\tilde{p}N}\frac{1-\overline{\lambda_2}^K}{1-\overline{\lambda_2}}\frac{1-\lambda^{k/K}}{1-\lambda}\right)\|\tilde{x}(0)\|\end{aligned} \tag{23}$$

Therefore, the critical value regarding the convergence of accelerated corrective consensus is

$$c = \lambda^{k/K} + \frac{1}{2}\hat{\beta}\epsilon\sqrt{2\tilde{p}N}\frac{1-\overline{\lambda_2}^K}{1-\overline{\lambda_2}}\frac{1-\lambda^{k/K}}{1-\lambda} \tag{24}$$

and it is obvious that c can be made smaller than 1 by controlling the value of $\tilde{p}$, which can be set arbitrarily close to 0 by adjusting the parameter n (the maximum number of repeated broadcasts to deliver a state variable), as explained in the proof of Theorem 5 in [5].

So far, we have proved Theorem 1 in the case where ϕ_{ij}'s are reliably exchanged. A similar proof can be given for the case that ϕ_{ij}'s could be lost in wireless channel, by following the similar procedures as does Theorem 6 in [5].

□