

LOCATION-AIDED FAST DISTRIBUTED AVERAGING

Wenjun Li and Huaiyu Dai

Department of Electrical and Computer Engineering
North Carolina State University, Raleigh, NC
{wli5, huaiyu_dai}@ncsu.edu

ABSTRACT

Existing works on distributed averaging explore linear iterations based on reversible Markov chains, and hence the convergence is bounded to be slow due to the diffusive behavior of the reversible random walk. In this paper, we study the possibility of utilizing nonreversible chains to achieve faster averaging in wireless networks. We first show that it is possible to achieve an ϵ -averaging time of $\Theta(r^{-1} \log(1/\epsilon))$ in a wireless network with a transmission radius r , with a centralized grid-based algorithm. We then proceed to propose a purely distributed algorithm, the Location-Aided Distributed Averaging-Uniform (LADA-U) algorithm, where the direction information of neighbors is used to construct nonreversible chains with uniform stationary distributions. It is shown that LADA-U can achieve the same scaling law in averaging time as the centralized scheme, but needs a substantially larger transmission range than minimum connectivity requirement, mainly due to the induced diffusive behavior.

1. INTRODUCTION

As basic building blocks for distributed and cooperative information processing in wireless networks, distributed consensus has been vigorously investigated recently [1–6]. In a distributed consensus problem, n nodes with node i having observation $x_i(0)$ try to reach a consensus on the average value $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i(0)$ through iterative local information exchange. Distributed averaging through deterministic linear iteration is studied in [1]. The class of randomized gossip algorithms recently studied by Boyd *et al* [2] realizes consensus through iterative pairwise averaging, and allows for asynchronous operation. In both fixed and random algorithms studied in [1,2], mainly a symmetric, doubly stochastic weight matrix is used, hence the convergence time of such algorithms is closely related to the mixing time of a reversible random walk, which is usually slow due to its diffusive behavior. Moreover, it has been shown in [2] that in a wireless network with a common transmission radius r , the optimal

gossip algorithm requires $\Theta(r^{-2} \log(1/\epsilon))$ time for the relative error to be bounded by ϵ .

It has been observed that certain nonreversible chains lifted from reversible ones mix substantially faster than the original chains [7, 8]. Motivated by this finding, we explore the idea of using nonreversible chains to accelerate distributed consensus. Each node maintains multiple copies of the estimated average value at any time, one corresponding to each direction the value will be transmitted in the next slot. By preventing the same information being “bounced” forth and back, information disseminates rapidly in the network and the averaging time is improved. Without loss of generality, we consider the set of initial values $\mathbf{x}(0) \in \mathbb{R}^{+n}$, and define the ϵ -averaging time as

$$T_{\text{ave}}(\epsilon) = \sup_{\mathbf{x}(0) \in \mathbb{R}^{+n}} \inf \{t : \|\mathbf{x}(t) - \bar{x}\mathbf{1}\|_1 \leq \epsilon \|\mathbf{x}(0)\|_1\}.$$

A wireless network is often modeled as a geometric random graph $G(n, r(n))$ [9], where n nodes are uniformly and independently distributed in a unit square $[0, 1]^2$, and $r(n)$ is the common transmission range of all nodes. It is known that $r(n) = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$ is required to ensure that the graph is connected with high probability [9]. We first show that it is possible to achieve an ϵ -averaging time of $O(r^{-1} \log(1/\epsilon))$ using a centralized grid-based algorithm, motivated by the fast-mixing nonreversible chain on a 2-dimensional grid [8].

In practice, purely distributed algorithms requiring no central coordination are typically preferred. This motivates a Location-Aided Distributed Averaging algorithm, LADA-U. As the name implies, this algorithm utilizes location information to construct nonreversible chains with fast mixing properties. Due to random node locations in wireless networks, a nonreversible chain constructed on the network graph typically does not possess a uniform stationary distribution. To facilitate distributed averaging, we carefully design the nonreversible chain to ensure such a property (which accounts for the suffix “U”), by allowing some controlled diffusive behavior. The chain is formed with locally computed weights based only on the direction information of neighbors. We show that a sufficient condition for LADA-U to achieve an ϵ -averaging

This research was supported in part by the National Science Foundation under Grant CCF-0515164.

time of $O(r^{-1} \log(1/\epsilon))$ is $r = \Omega\left(\left(\frac{\log n}{n}\right)^{\frac{1}{3}}\right)$.

Our paper is organized as follows. In Section II, we discuss the centralized grid-based algorithm, and analyze its performance. In Section III, we introduce the LADA-U algorithm and compare it with known schemes in literature. Finally, concluding remarks are contained in Section IV.

2. A CENTRALIZED GRID-BASED ALGORITHM

In this section, we present a centralized algorithm which has an ϵ -averaging time of $O(r^{-1} \log(\epsilon^{-1}))$ on $G(n, r)$. Consider a tessellation of the unit area into $k^2 \triangleq \lceil \frac{\sqrt{2}}{r} \rceil^2$ squares (clusters). We assume that a central controller can cluster nodes based on squares they fall in (this dictates some global information on nodes' coordinations). By this tessellation, a node in a given cluster is adjacent to all nodes in the four neighboring clusters. Denote the number of nodes in a given cluster m by n_m . Then for a geometric random graph $n_m \geq 1$ for all m w.h.p. as $n \rightarrow \infty$ [9]. One node in each cluster is selected as a cluster-head, which is responsible for exchanging values with other clusters and broadcasting the updated value inside the cluster. Denote the index of the cluster a node i lies in by C_i . For each cluster m , denote its east, north, west and south neighboring cluster (if exists) respectively by N_m^0, N_m^1, N_m^2 and N_m^3 .

The centralized algorithm works as follows. Every cluster-head maintains four values corresponding to the four directions, namely, the east, north, west and south value, denoted respectively by y_m^0, y_m^1, y_m^2 and y_m^3 for cluster m . In the initialization stage, every node transmits its value to the cluster-head. The cluster-head computes the sum of the values within the cluster and initializes all four values to

$$y_m^l(0) = \sum_{C_i=m} x_i(0), \quad l = 0, \dots, 3. \quad (1)$$

At each time instant t , the cluster-heads of neighboring clusters communicate and update their values. Specifically, the east value of cluster m is updated with

$$y_m^0(t+1) = \left(1 - \frac{1}{k}\right) y_{N_m^2}^0(t) + \frac{1}{2k} (y_m^1(t) + y_m^3(t)). \quad (2)$$

That is, the east value of cluster m is a weighted sum of the previous values of the west neighbor of m and itself, with the majority $(1 - \frac{1}{k})$ coming from the east value of the west neighbor, and a fraction of $\frac{1}{2k}$ coming from the north value as well as the south value of itself. If m is a west border cluster (with no west neighbor), then $y_{N_m^2}^0(t)$ is replaced with the west value of itself $y_m^2(t)$ (i.e., the west value is ‘‘bounced back’’ when it reaches the boundary). This is a natural procedure on the grid structure to ensure that the iteration evolves according to a doubly stochastic matrix which is desirable for averaging. Moreover, the fact that the information continues

to propagate when it reaches the boundary is essential for the associated chain to mix rapidly. Similarly, the north value of cluster m is a weighted sum of the previous values of its south neighbor and itself, with the majority coming from the north value of the south neighbor, and so on. Each cluster-head then calculates the average of its four values as an estimate for the global average, and broadcasts this estimate to its members, so that every node i in this cluster obtains

$$x_i(t+1) = \frac{k^2}{4n} \sum_{l=0}^3 y_{C_i}^l(t+1). \quad (3)$$

In the following, we give some analysis on the performance of this algorithm. Denote $\hat{\mathbf{y}} = [\mathbf{y}_0^T, \mathbf{y}_1^T, \mathbf{y}_2^T, \mathbf{y}_3^T]^T$, with $\mathbf{y}_l = [y_1^l, y_2^l, \dots, y_{k^2}^l]^T$. The update of $\hat{\mathbf{y}}$ can be expressed with $\hat{\mathbf{y}}(t+1) = \mathbf{P}^T \hat{\mathbf{y}}(t)$, where \mathbf{P} corresponds to a doubly stochastic, irreducible and aperiodic nonreversible Markov chain. Hence, the chain has a uniform stationary distribution, and it is known from Markov chain theory that for any initial values, each y_i^l approaches $\frac{nx_{\text{ave}}}{k^2}$, i.e., the sum of values averaged by the number of clusters. This accounts for the normalization factor $\frac{k^2}{4n}$ in (3). It can be expected that the convergence rate of the centralized scheme is closely related to the mixing time of the Markov chain \mathbf{P} . For $\epsilon > 0$, the ϵ -mixing time of an irreducible and aperiodic Markov chain \mathbf{P} with stationary distribution π is defined as [10]

$$T_{\text{mix}}(\mathbf{P}, \epsilon) = \max_i \inf \{t : 1/2 \|\mathbf{P}^t(i, \cdot) - \pi\|_1 \leq \epsilon\}, \quad (4)$$

where $\mathbf{P}^t(i, \cdot)$ is the t -step transition probabilities given that the start state is i . Since the nonreversible random walk \mathbf{P} most likely keeps its direction, occasionally makes a turn, and never turns back, it mixes substantially faster than a simple random walk. In [11], we provide a rigorous proof for an upper bound for the mixing time of \mathbf{P} , as given in Part a) of the following lemma. Part b) and c) of the lemma establish lower bounds for the mixing time of \mathbf{P} .

Lemma 2.1. *The ϵ -mixing time of \mathbf{P} is*

- a) $T_{\text{mix}}(\mathbf{P}, \epsilon) = O(k \log(\epsilon^{-1}))$, for any $\epsilon > 0$;
- b) $T_{\text{mix}}(\mathbf{P}, \epsilon) = \Omega(k)$, for a constant $0 < \epsilon < \frac{3}{32}$;
- c) $T_{\text{mix}}(\mathbf{P}, \epsilon) = \Omega(k \log k)$, for $\epsilon = O(\frac{1}{k})$;

Proof. a) See Appendix I of [11].

b) For the random walk starting from s , denote by \hat{s}_t the state it visits at time t if it never makes a turn. Then for $t \leq k$, we have $P^t(s, \hat{s}_t) = (1 - \frac{1}{k})^t \geq (1 - \frac{1}{k})^k > \frac{1}{4k^2}$, and hence for $k \geq 2$, $\|P^t(s, \cdot) - \pi\|_1 \geq (1 - \frac{1}{k})^k - \frac{1}{4k^2} \geq \frac{1}{4} - \frac{1}{16} = \frac{3}{16}$.

c) For any initial state s , denote \hat{s}_t as in b). Then for $t \leq k \log k$, $P^t(s, \hat{s}_t) = (1 - \frac{1}{k})^t \geq (1 - \frac{1}{k})^{k \log k} > \frac{1}{2k}$, and hence for $k \geq 2$, $\|P^t(s, \cdot) - \pi\|_1 \geq (1 - \frac{1}{k})^{k \log k} - \frac{1}{4k^2} > \frac{1}{2k} - \frac{1}{4k^2} > \frac{1}{4k}$. \square

Remark: In order to establish a lower bound for the averaging time of the centralized algorithm, the result in b) and c)

needs to be extended as follows. For a node v in the $k \times k$ grid, denote the four lifted states $v_i, i = 0, 1, 2, 3$. We have for any initial state s , and $k \geq 3$,

1. for $t \leq k, \sum_{v=1}^{k^2} |\sum_{i=0}^3 P^t(s, v_i) - \sum_{i=0}^3 \pi(v_i)| \geq (1 - \frac{1}{k})^k - \frac{1}{k^2} \geq \frac{8}{27} - \frac{1}{9} = \frac{5}{27}$.
2. for $t \leq k \log k, \sum_{v=1}^{k^2} |\sum_{i=0}^3 P^t(s, v_i) - \sum_{i=0}^3 \pi(v_i)| \geq (1 - \frac{1}{k})^{k \log k} - \frac{1}{k^2} \geq \frac{1}{2k} - \frac{1}{k^2} > \frac{1}{6k}$.

Theorem 2.1. *The centralized algorithm has an ϵ -averaging time $T_{\text{ave}}(\epsilon) = O(r^{-1} \log(\epsilon^{-1}))$ in a wireless network with a common transmission radius $r = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$. Moreover, for a small enough constant $\epsilon, T_{\text{ave}}(\epsilon) = \Theta(r^{-1})$; for $\epsilon = O(r), T_{\text{ave}}(\epsilon) = \Omega(r^{-1} \log(r^{-1}))$.*

Proof. We can appeal to uniform convergence in the law of large numbers using Vapnik-Chervonenkis theory as in [9] to show that the number of nodes in each cluster satisfies

$$\Pr\left(\max_{1 \leq m \leq k^2} \left| \frac{n_m}{n} - \frac{1}{k^2} \right| \leq \frac{4 \log n}{n}\right) > 1 - \frac{4 \log n}{n}. \quad (5)$$

Thus we have for all $m, n_m \geq \frac{n}{k^2} - 4 \log n = \frac{nr^2}{5} - 4 \log n$, which is at least 1 for sufficiently large n if $r > \sqrt{\frac{20 \log n}{n}}$. In this case, we have that $\frac{c_2 n}{k^2} \leq n_m \leq \frac{c_1 n}{k^2}$ for all m for some constants $c_1, c_2 > 0$ w.h.p. as $n \rightarrow \infty$. By Lemma 2.1 a), for any $\epsilon > 0$, there exists some $\tau \triangleq T_{\text{mix}}(\mathbf{P}, \frac{\epsilon}{2c_1}) = O(r^{-1} \log(\epsilon^{-1}))$ such that for all $t \geq \tau$,

$$\begin{aligned} \|\mathbf{x}(t) - x_{\text{ave}} \mathbf{1}\|_1 &= \sum_{m=1}^{k^2} n_m \left| \frac{k^2}{4n} \sum_{l=0}^3 y_m^l(t) - x_{\text{ave}} \right| \\ &\leq \sum_{m=1}^{k^2} \frac{n_m k^2}{4n} \sum_{l=0}^3 |y_m^l(t) - \frac{n x_{\text{ave}}}{k^2}| \\ &\leq \frac{c_1}{4} \sum_{j=1}^{4k^2} |\hat{y}_j(t) - \frac{n x_{\text{ave}}}{k^2}| \\ &\leq \frac{c_1}{4} \cdot \frac{\epsilon}{c_1} \cdot \sum_{m=1}^{4k^2} |\hat{y}_j(0)| = \epsilon \|\mathbf{x}(0)\|_1, \end{aligned}$$

where $\hat{y}_j(t)$ as defined above is the value for the j th state without noting the specific node it belongs to, and the last inequality follows from the definition of the mixing time and the convexity of the l_1 norm (i.e., the l_1 norm is maximized when the initial distribution is a point mass).

To prove the latter part of the theorem, note that $\|\mathbf{x}(t) - x_{\text{ave}} \mathbf{1}\|_1 \geq \frac{c_2}{4} \sum_{m=1}^{k^2} |\sum_{l=0}^3 y_m^l(t) - \frac{4n x_{\text{ave}}}{k^2}|$. The rest follows from the remark of Lemma 2.1. \square

3. LOCATION-AIDED DISTRIBUTED AVERAGING

In large dynamic wireless networks, it is often impossible to have a central controller that maintains a global coordinate system and clusters the nodes accordingly. In this section, we introduce the LADA-U algorithm, which utilizes only local information to construct fast mixing nonreversible chains.

3.1. Neighbor Classification

A neighbor classification procedure is needed prior to the distributed computation. We assume that each node i has knowledge of directions and can classify its neighbors as follows: a neighbor j of node i is said to be a Type- l neighbor of i , denoted as $j \in \mathcal{N}_i^l$, if

$$\angle(X_j - X_i) \in \left(\frac{l\pi}{2} - \frac{\pi}{4}, \frac{l\pi}{2} + \frac{\pi}{4} \right] \quad l = 0, \dots, 3, \quad (6)$$

where $X_i = \text{Re}(X_i) + i\text{Im}(X_i)$ denotes the location of node i in complex form.

In literature, a unit torus or sphere is often assumed in performance analysis to avoid the edge effects [2, 9]. In our study, we explicitly deal with the edge effects by considering the following modification, as illustrated in Fig. 1. A boundary node is a node within distance r from one of the boundaries, e.g., node i in Fig. 1. For a boundary node i , we create mirror images of its neighbors with respect to the boundary. If a neighbor j has an image located within the transmission range of i , node j (besides its original role) is considered as a virtual neighbor of i , whose direction is determined by the image's location with respect to the location of i . Denote the set of virtual east neighbors of an east boundary node i resulted from reflecting at the east boundary by $\tilde{\mathcal{N}}_i^0$, and the set of virtual east neighbors of a north or south boundary node i resulted from reflecting at the north or south boundary by $\tilde{\mathcal{N}}_i^0$. Similarly, $\tilde{\mathcal{N}}_i^l$ denotes the set of virtual north neighbors of node i resulted from reflecting at the north boundary, and $\tilde{\mathcal{N}}_i^l$ denotes those reflected at the east or west boundary, and so on for virtual west and south neighbors. We denote the number of type l (physical and virtual) neighbors in direction l as $d_i^l \triangleq |\mathcal{N}_i^l| + |\tilde{\mathcal{N}}_i^l| + |\hat{\mathcal{N}}_i^l|$. With this modification, every type- l neighborhood has an effective area $\frac{\pi r^2}{4}$, hence d_i^l is roughly the same for all i and l . We also expect that as r increases, the fluctuation in d_i^l diminishes, as given by the following lemma, whose proof follows a similar argument as the proof of Lemma 2.1, and is omitted due to space concerns.

Lemma 3.1. *When $r = \Omega\left(\left(\frac{\log n}{n}\right)^{\frac{1}{3}}\right)$, we have for all i and $l, d_i^l = \frac{n\pi r^2}{4} (1 \pm O(r))$ w.h.p.*

3.2. LADA-U Algorithm

The LADA-U algorithm works as follows. Each node i holds four values $y_i^l, l = 0, \dots, 3$ corresponding to the four direc-

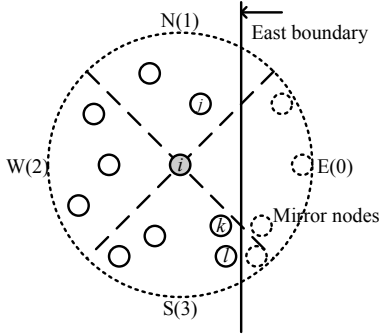


Fig. 1. Illustration of neighbor classification and virtual neighbors for boundary nodes. Note that for an east boundary node i , there can only be virtual east neighbors of the first category ($i, j, k \in \tilde{\mathcal{N}}_i^0$), and virtual north and south neighbors of the second category ($l \in \tilde{\mathcal{N}}_i^3$).

tions, all initialized to $x_i(0)$. During each iteration, the east value of node i is updated with

$$y_i^0(t+1) = (1-p) \left[\sum_{j \in \mathcal{N}_i^2} \frac{y_j^0(t)}{d_{\max}} + \left(1 - \frac{d_i^2}{d_{\max}}\right) y_i^2(t) \right] + \frac{1}{2} p (y_i^1(t) + y_i^3(t)) \quad (7)$$

where $d_{\max} = \max_{i,l} d_{i,l}^l$, and $p = \Theta(r)$ is the probability of making a 90 degree turn. As in the centralized algorithm, boundary nodes must be treated specially. If i is a west boundary node, then we must include an additional term $\sum_{j \in \tilde{\mathcal{N}}_i^2} \frac{y_j^2(t)}{d_{\max}}$ in the bracket in (7), i.e. west values of virtual west neighbors $\tilde{\mathcal{N}}_i^2$ resulted from the west boundary are also used to form the east value of i . This ensures that information continues to propagate when reaching the boundary as in the centralized case. If i is a north or south boundary node, however, the sum in (7) is replaced with $\sum_{j \in \mathcal{N}_i^2 \cup \tilde{\mathcal{N}}_i^2} \frac{y_j^2(t)}{d_{\max}}$, i.e., east values of both physical and virtual west neighbors resulted from the north or south boundary are used. Note that $\tilde{\mathcal{N}}_i^2$ are meant only for compensating the loss of neighbors for north or boundary nodes, so unlike the previous case, their east or west values continue to propagate in the usual direction. The north, west and south values are updated in the same fashion. Node i computes its estimate of \bar{x} with $x_i(t) = \frac{1}{4} \sum_{l=0}^3 y_i^l$.

We then give some performance analysis for LADA-U. Denote $\hat{\mathbf{y}} = [\mathbf{y}_0^T, \mathbf{y}_1^T, \mathbf{y}_2^T, \mathbf{y}_3^T]^T$, with $\mathbf{y}_l = [y_1^l, y_2^l, \dots, y_n^l]^T$. The iteration can be written as $\hat{\mathbf{y}}(t+1) = \mathbf{P}_1^T \hat{\mathbf{y}}(t)$, where \mathbf{P}_1 is a doubly stochastic matrix through our design. The exchange weights for an east value of some node i are illustrated in Fig. 2: a fraction $\frac{p}{2}$ of the east value goes to the north and south value of the same node respectively, a total fraction of

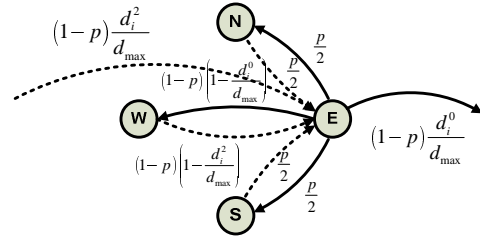


Fig. 2. The Markov chain used in LADA-U: outgoing probabilities (solid lines) and incoming probabilities (dotted lines) for the east state are depicted

$\frac{d_i^0}{d_{\max}}(1-p)$ goes uniformly to the east values of d_i^0 east neighbors, and the remaining $\left(1 - \frac{d_i^0}{d_{\max}}\right)(1-p)$ goes to the west value of node i . The transitions between the east and west state make up for the difference in d_i^0 and d_i^2 , and ensures that the incoming probabilities for each state also sum to 1. While such a design guarantees that the associated chain has a uniform stationary distribution, it also introduces some diffusive behavior, hence the centralized performance can only be achieved with a sufficiently large r .

Theorem 2: For LADA-U, $T_{\text{ave}}(\epsilon) = \Theta(r^{-1} \log(1/\epsilon))$ when the transmission radius $r = \Omega\left(\left(\frac{\log n}{n}\right)^{\frac{1}{3}}\right)$.

Sketch of Proof. Consider a movement of the random walk \mathbf{P}_1 . From uniform node deployment, it can be calculated that the average distance traveled in the direction of movement is $\frac{4\sqrt{2}}{3\pi}r \triangleq \mu_\alpha$, and that orthogonal to the direction of movement is 0. Assume $k = \frac{1}{\mu_\alpha} + 1$ and the turning probability $p = \frac{1}{k} = \Theta(r)$. It can be shown that the expected location of the random walk \mathbf{P}_1 evolves according to a random walk $\tilde{\mathbf{P}}$ on the $k \times k$ grid. $\tilde{\mathbf{P}}$ differs from \mathbf{P} in Section 2 only in the additional transitions between states of opposite directions corresponding to the same node. From Lemma 3.1, when $r = \Omega\left(\left(\frac{\log n}{n}\right)^{\frac{1}{3}}\right)$, we have for each move, the probability that the random walk $\tilde{\mathbf{P}}$ keeps the direction is at least

$(1-p) \frac{d_{\min}}{d_{\max}} = (1-1/k)(1-O(r)) > 1 - \frac{c_1}{k}$ for some constant $c_1 > 1$. During the first $6k$ moves, the probability that the random walk $\tilde{\mathbf{P}}$ makes exactly two 90 degree turns towards given directions at given times T_1 and T_2 , and keeps direction for the remaining moves is at least $\frac{1}{4k^2} \left(1 - \frac{c_1}{k}\right)^{6k-2} \geq \frac{2^{-12c_1}}{4k^2}$.

Then, if the random walk $\tilde{\mathbf{P}}$ starts from an east or west state, any east or west state can be reached with probability at least $\frac{2^{-12c_1}}{4k^2}$ in $6k$ steps. The case for north and south states can be similarly argued, and we conclude that the state distribution of the random walk $\tilde{\mathbf{P}}$ is approximately uniform at $t = 6k$. Now, taking the random node location into account, it can be shown that when $n \rightarrow \infty$, the exact location of the random walk \mathbf{P}_1 is also approximately uniform in $6k$ steps, due to the approximate regularity of the underlying graph through

the introduction of virtual neighbors. This implies that the ϵ -mixing time of \mathbf{P}_1 , as well as the ϵ -averaging time of LADA-U is $O(r^{-1} \log(\epsilon^{-1}))$.

3.3. Comparison with Existing Works

In this section, we compare the LADA-U algorithm with some known distributed consensus algorithms proposed in literature, in addition to those mentioned in the introduction. The consensus propagation algorithm proposed by Moalleimi and Roy [3] is a special form of Gaussian belief propagation, which avoids passing information back to where it is received. The convergence time is significantly reduced compared with gossip algorithms on a singly connected graph and on an n -cycle. While exact performance scaling laws are not yet available, it can be expected that the gain of consensus propagation over gossip algorithms diminishes on geometric random graphs where node degrees are much larger than 1, as the diffusive behavior is not effectively reduced with this algorithm.

The work by Savas *et al.* [4] explored distributed computation of decomposable functions through sequential algorithms, namely, SIMPLE-WALK and COALESCENT. It is shown that both the time and message complexity of SIMPLE-WALK is equal to the cover time of the network graph, which is $\Theta(n \log n)$ w.h.p. for $G(n, r)$ with r guaranteeing connectivity [12]. Note that in LADA-U, each node needs to broadcast its value to its neighbors in each iteration. Thus, for $\epsilon = \frac{1}{n^\alpha}$, $\alpha > 0$, our LADA-U algorithm suffers from a loss of $O\left(\sqrt{\frac{n}{\log n}}\right)$ in message complexity, but gains $\Omega\left(\sqrt{n \log n}\right)$ in time complexity compared with SIMPLE-WALK.

The geographic gossip algorithm of Dimakis *et al.* [5] endeavored to mitigate the slow convergence of standard gossip algorithms by constructing long-haul links to facilitate information propagation. It is shown that geographic gossip computes the average to accuracy ϵ using $O\left(\sqrt{\frac{n^{1.5}}{\log n}} \log(\epsilon^{-1})\right)$ radio transmissions, which is the same as the LADA-U algorithm. However, their algorithm requires a global coordinate system as in our centralized scheme, and the geographic routing and resampling procedures make it considerably more complex than our LADA-U algorithm.

A similar idea of fast averaging through nonreversible lifting has been independently studied in [6]. Their work adopts a lifting proposed in [8], which requires global information of the network to construct. When a node joins or leaves the network, the entire chain needs to be re-calculated, and the state space of the new chain is of a size up to n^3 . On the other hand, LADA-U is robust to topology changes, and the size of the chain remains on the same order ($4n$). Finally, in our recent work [11], a LADA algorithm based on Markov chains with nonuniform stationary distributions is proposed. A weight variable must be maintained for each state which estimates its stationary probability and asymptotically yields the accurate average estimate. The LADA-U algorithm proposed

in this paper avoids the weight estimation procedure, and provides some theoretical insight in constructing non-reversible chains with uniform stationary distributions.

4. CONCLUSION

The idea of using nonreversible chains to accelerate distributed consensus in wireless networks is investigated in this paper. We show that in a wireless network with transmission radius r , an ϵ -averaging time of $O(r^{-1} \log(\epsilon^{-1}))$ can be achieved with a centralized algorithm, where nodes are clustered using a central controller based on a global coordinate system. We then present a purely distributed algorithm, the LADA-U(Uniform) algorithm, which utilizes only direction information of neighbors to construct nonreversible chains. The non-reversible chain constructed in LADA-U is carefully designed to ensure a uniform stationary distribution, by allowing some controlled diffusive behavior. It is shown that LADA-U can achieve the same scaling law in averaging time as the centralized scheme and LADA, but needs a substantially larger transmission range than minimum connectivity requirement, mainly due to the induced diffusive behavior.

5. REFERENCES

- [1] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," in *IEEE Conf. on Decision and Control*, Maui, Hawaii, Dec. 2003.
- [2] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2506–2530, 2006.
- [3] C. C. Moalleimi and B. Van Roy, "Consensus propagation," *IEEE Trans. Inform. Theory*, vol. 52, no. 11, pp. 4753–4766, Nov. 2006.
- [4] O. Savas, M. Alanyali, and V. Saligrama, "Randomized sequential algorithms for data aggregation in sensor networks," in *Conference on Information Sciences and Systems (CISS)*, Princeton University, NJ, Mar. 2006.
- [5] A. G. Dimakis, A. D. Sarwate, and M. J. Wainwright, "Geographic gossip: efficient aggregation for sensor networks," in *Information Processing in Sensor Networks (IPSN) 2006*, Nashville, TN, Apr. 2006.
- [6] K. Jung and D. Shah, "Fast gossip via nonreversible random walk," in *IEEE Information Theory Workshop (ITW'06)*, Punta del Este, Uruguay, Mar. 2006.
- [7] P. Diaconis, S. Holmes, and R. M. Neal, "Analysis of a non-reversible markov chain sampler," Tech. Rep. BU-1385-M, Biometrics Unit, Cornell University, 1997.
- [8] Fang Chen, László Lovász, and Igor Pak, "Lifting markov chains to speed up mixing," in *31st Annual ACM Symposium on Theory of Computing (STOC'99)*, Atlanta, Georgia, May 1999.
- [9] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.
- [10] D. Aldous and J. Fill, *Reversible Markov Chains and Random Walks on Graphs*, Online book available at <http://www.stat.berkeley.edu/users/aldous/RWG/book.html>.
- [11] W. Li and H. Dai, "Accelerating distributed consensus via lifting Markov chains," in *Proc. IEEE Int. Symp. Information Theory (ISIT) 2007*, Nice, France, June 2007.
- [12] C. Avin and G. Ercal, "On the cover time and mixing time of random geometric graphs," To appear in *Journal of Theoretical Computer Science*.