

# Accelerating Distributed Consensus Via Lifting Markov Chains

Wenjun Li and Huaiyu Dai

Department of Electrical and Computer Engineering,  
North Carolina State University, Box 7911, Raleigh, NC 27695  
{wli5, Huaiyu\_dai}@ncsu.edu

**Abstract**—Existing works on distributed averaging explore linear iterations based on reversible Markov chains. The convergence of such algorithms is bounded to be slow due to the diffusive behavior of the reversible chains. It has been observed that certain nonreversible chains lifted from reversible ones mix substantially faster than the original chains [1], [2]. We show that the idea of nonreversible lifting lends itself naturally to a fast distributed averaging algorithm, where each node maintains multiple estimates, corresponding to multiple lifted states in the Markov chain. We give a rigorous proof that it is possible to achieve an  $\epsilon$ -averaging time of  $\Theta(k \log(1/\epsilon))$  on a  $k \times k$  grid. For a general wireless network, we propose a Location-Aided Distributed Averaging (LADA) algorithm, which utilizes local information to construct a fast-mixing nonreversible chain in a distributed manner. We show that using LADA, an  $\epsilon$ -averaging time of  $\Theta(r^{-1} \log(1/\epsilon))$  is achievable in a wireless network with transmission radius  $r$ .

## I. INTRODUCTION

The average of node values is desired in many network applications, such as computing the sufficient statistic for distributed detection and estimation, as well as network optimization. The distributed averaging problem where nodes try to reach consensus on the average value through iterative local information exchange has been vigorously investigated recently [3]–[5]. Compared with centralized counterparts, such distributed algorithms scale well as the network grows, and exhibit robustness to node and link failures. Distributed averaging through deterministic linear iteration is studied in [3]. The randomized gossip algorithm studied by Boyd *et al* [4] realizes averaging through iterative pairwise relaxations, and allows for asynchronous operation. In both fixed and random algorithms studied in these works, a symmetric 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  $\Theta(n^2)$  on an  $n$ -path, and  $\Theta(k^2)$  on a  $k \times k$  2-dimensional (2-d) grid. Moreover, it has been shown in [4] that in a wireless network with common transmission radius  $r$ , the optimal gossip algorithm requires  $\Theta(r^{-2} \log(1/\epsilon))$  time for the relative error to be bounded by  $\epsilon$ . This means that for a small radius of transmission, even the fastest gossip algorithm converges slowly.

In this work, we explore distributed averaging algorithms based on nonreversible random walks, thereby considerably

improving the averaging time. In the literature on Markov chain sampling, the constructed Markov chain is usually reversible, as they are more mathematically tractable—see [6] and references therein. However, it is observed by Diaconis *et al.* [1] and later by Chen *et al.* [2] that certain nonreversible chains mix substantially faster than closely related reversible chains, by overcoming the diffusive behavior of the reversible random walk. Our work is directly motivated by this finding. Firstly, we show that indeed it is possible to design a distributed averaging algorithm that has averaging time  $\Theta(k \log(1/\epsilon))$  on a 2-d  $k \times k$  grid. This is achieved by allowing each node to have multiple copies of the estimated average value at any time, one corresponding to each direction the value will be transmitted in the next slot. The underlying random walk is more likely to keep its direction than making a turn, thus the diffusive behavior of a simple random walk is suppressed. Subsequently, for a general wireless network, we propose a Location-Aided Distributed Averaging (LADA) algorithm, which requires for each node, a loose knowledge of the relative location of its neighbors. A fast-mixing nonreversible chain is constructed in a distributed manner based on the local information. The constructed chain does not naturally possess a uniform stationary distribution, thus every node needs to keep a weight estimate in order to produce an average estimate. We show that an  $\epsilon$ -averaging time of  $\Theta(r^{-1} \log(1/\epsilon))$  is achievable with the LADA algorithm in a wireless network with common transmission range  $r$ .

Our paper is organized as follows. In Section II, we formulate the problem and discuss several important related works. In Section III, we propose distributed averaging algorithms based on nonreversible chains on a 2-d grid, and study its averaging time. In Section IV, we introduce the LADA algorithm for general wireless networks, and analyze its performance. We conclude the paper in Section V.

## II. PROBLEM FORMULATION AND RELATED WORKS

### A. Problem Formulation

Consider a network represented by a connected graph  $G = (V, E)$ , where the vertex set  $V$  contains  $n$  nodes and  $E$  is the edge set. Let vector  $\mathbf{x}(0) = [x_1(0), \dots, x_n(0)]^T$  contain the initial values observed by the nodes, and  $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$  denote the average value of  $\mathbf{x}(0)$ . The goal is to compute  $\bar{x}$  in a distributed and robust fashion. Let  $\mathbf{x}(t)$  be the vector containing node values at the  $t$ th iteration. In this paper, we

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

focus on synchronous algorithms without gossip constraints, i.e., at each time slot, every sensor updates its value based on the neighbors' values in the previous iteration. However, the proposed algorithms can also be realized in a deterministic gossip fashion, by simulating at most  $d_{\max}$  matchings for each iteration, where  $d_{\max}$  is the maximum node degree [7]. Without loss of generality, we consider the set of initial values  $\mathbf{x}(0) \in \mathbb{R}^{+n}$ , and define the  $\epsilon$ -averaging time as

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

where  $\|\cdot\|_1$  denotes the  $l_1$  norm.

### B. Related Works

Xiao and Boyd [3] derived necessary and sufficient conditions for the weight matrix  $\mathbf{W}$  such that the linear iteration  $\mathbf{x}(t+1) = \mathbf{W}\mathbf{x}(t)$  asymptotically computes  $\bar{x}\mathbf{1}$  as  $t \rightarrow \infty$ , where  $\mathbf{W}$  is graph comformant, i.e.,  $W_{ij} \neq 0$  only if  $(i, j) \in G$ . They formulated the fastest linear averaging problem as a semi-definite program, which is convex when  $\mathbf{W}$  is restricted to be symmetric. Finding the optimal symmetric  $\mathbf{W}$  with non-negative weights is closely tied to the problem of finding the fastest mixing reversible Markov chain on the graph.

The gossip algorithm proposed by Boyd *et al.* [4] realizes distributed averaging through asynchronous pairwise relaxation. At each time instant, a random node  $i$  chooses one of its neighbor  $j$  randomly with probability  $P_{ij}$ , where  $\mathbf{P}$  is a graph comformant stochastic matrix. Node  $i$  and  $j$  both update their values by averaging their values. For symmetric  $\mathbf{P}$ , the authors show that the absolute  $1/n^k$ -averaging time ( $k > 0$ ) of the gossip algorithm is related to the mixing time of the reversible chain  $\mathbf{P}$  as  $\Theta(\log n + T_{\text{mix}}(\mathbf{P}, \epsilon))$ .

Recently, Moalleimi and Roy [5] proposed consensus propagation, a special form of Gaussian belief propagation, as another alternative for distributed averaging. By avoiding passing information back to the neighbor from which the message is received, consensus propagation suppresses to some extent the diffusive nature of a reversible random walk. However, the gain of consensus propagation over gossip algorithms is expected to diminish as the graph gets better connected, i.e., the node degrees become much larger than 1, in which case the diffusive behavior is not effectively reduced.

Upon submission, we became aware of the independent work by Jung and Shah [7], which also explored the idea of using nonreversible chains for fast averaging. However, our work differs from theirs in several aspects. Firstly, their algorithm is based on the lifting proposed in [2], which is difficult to construct for large-scale networks and requires global knowledge of the network. One the other hand, the chain used in our algorithm is formed in a distributed fashion using only local information. As a result, our algorithm is scalable while theirs is not: when a node joins or leaves the network, only its neighbors need to update their local processing rules using our algorithm; while using their algorithm, the entire chain needs to be re-calculated. Secondly, our algorithm enjoys significant computational advantage: every node needs to maintain 4 values, in contrast to as many as  $n^2$  values in

their algorithm. In conclusion, the LADA algorithm is suited for large-scale networks with frequent topological changes.

### III. FAST DISTRIBUTED CONSENSUS ON A GRID

It has been observed by Diaconis *et al.* [1] and Chen *et al.* [2] that non-reversible chains constructed on a "lifted" graph mixes substantially faster than related reversible chains. In this section, we show a nonreversible chain on a 2-d grid lends itself naturally to a fast distributed averaging algorithm on the grid. The study of the grid case is motivated by its close connection with a 2-d geometric random graph often used to model wireless networks [8], which will be discussed in Section IV.

Consider a  $k \times k$  grid. For each node  $i$ , denote its east, north, west and south neighbor (if exists) respectively by  $N_i^0, N_i^1, N_i^2$  and  $N_i^3$ . Each node maintains four copies of estimate corresponding to the four directions. The east, north, west and south values are denoted respectively by  $y_i^0, y_i^1, y_i^2$  and  $y_i^3$ , and all are initialized to  $x_i(0)$ . At each time instant  $t$ , the east value of node  $i$  is updated with

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

That is, the east value of  $i$  consists of a fraction of  $1 - \frac{1}{k}$  of the east value of the west neighbor, and a fraction of  $\frac{1}{2k}$  of the north value as well as the south value of itself. If  $i$  is a west border node (with no west neighbor),  $y_{N_i^2}^0(t)$  is replaced with  $y_i^2(t)$  (i.e., the west value is "bounced back" when it reaches the boundary). Similarly, the north value of  $i$  is a weighted sum of its south neighbor and itself in the previous iteration, with majority coming from the north value of the south neighbor, and so on. Node  $i$  estimates  $\bar{x}$  with

$$x_i(t) = \frac{1}{4} \sum_{l=0}^3 y_i^l. \quad (3)$$

Denote  $\hat{\mathbf{y}} = [y_0^T, y_1^T, y_2^T, y_3^T]^T$ , 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  $\mathbf{P}$  defines a doubly stochastic, irreducible and aperiodic Markov chain, as illustrated in Fig. 1. For  $\epsilon > 0$ , the mixing time of a Markov chain  $\mathbf{P}$  with stationary distribution  $\pi$  is defined as

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

Since the random walk of interest most likely keeps its direction, occasionally makes a turn, and never turns back, it mixes substantially faster than a simple random walk. A slightly different nonreversible chain on a 2-d torus has been studied in [2]. It is shown that an optimal stopping rule takes  $\Theta(k)$  time in order that the state distribution of the random walk is approximately stationary. In this paper, we provide a result of the mixing time of  $\mathbf{P}$  in terms of the standard definition in (4) on a 2-d grid.

*Lemma 3.1* The mixing time of the chain  $\mathbf{P}$  is  $T_{\text{mix}}(\mathbf{P}, \epsilon) = \Theta(k \log(1/\epsilon))$ .

The proof is given in Appendix I. The key is to show that by time  $t = 6k$ , the random walk starting from any state visits every state  $i$  with probability at least  $C\pi_i$  with  $C = 2^{-15}$ . Then

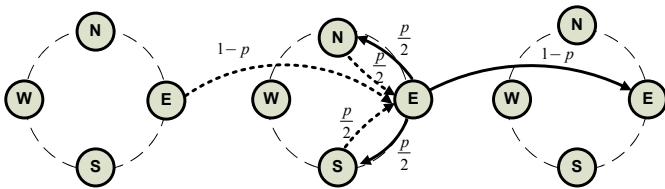


Fig. 1. Nonreversible chain used in distributed averaging on a  $k \times k$  grid: outgoing probabilities (solid lines) and incoming probabilities (dotted lines) for the east state of the central node are depicted.

by a well-known result in Markov chain theory [9], we have that for all  $t = 1, 2, \dots$ ,  $\|P(s_0, \cdot)^t - \pi\|_1 \leq (1-C)^{t/6k}$ . This implies that the mixing time is  $T_{\text{mix}}(\mathbf{P}_1, \epsilon) = \Theta(k \log(1/\epsilon))$ . The former argument is shown by enumerating the possibilities of all ending states if the random walk makes 2 turns or 3 turns towards given directions and their corresponding probabilities.

Theorem 3.1 is immediate from Lemma 3.1 and the convexity of the  $l_1$  norm (i.e., the  $l_1$  norm is maximized when the initial distribution is a point mass).

*Theorem 3.1* On a  $k \times k$  grid using the updates (2)-(3),  $T_{\text{ave}}(\epsilon) = \Theta(k \log(1/\epsilon))$ .

#### IV. LOCATION-AIDED DISTRIBUTED CONSENSUS IN WIRELESS NETWORKS

Inspired by the tremendous speedup of the convergence of distributed consensus on regular graphs, we propose a distributed averaging algorithm for wireless sensor networks, which utilizes location information of the neighborhood to “orient” the information flows to accelerate convergence. We will use the celebrated geometric random graph introduced by Gupta and Kumar [8] to model the wireless sensor network. In a geometric random graph  $G(n, r(n))$ ,  $n$  nodes are uniformly and independently distributed on a unit square  $[0, 1]^2$ , and  $r(n)$  is the common transmission range of all nodes. It is known that the choice of  $r(n) = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$  is required to ensure the graph is connected with high probability [8].

##### A. Neighbor Classification

We assume that each node  $i$  knows the locations of its neighbors with respect to itself. Let the coordinates of the  $i$ th node in the complex form be  $X_i = \text{Re}(X_i) + i\text{Im}(X_i)$ , node  $i$  classifies its neighbors based on their locations w.r.t. itself: a neighbor  $j$  of  $i$  is said to be a Type- $k$  neighbor of node  $i$ , denoted as  $j \in \mathcal{N}_i^k$ , if

$$\angle(X_j - X_i) \in \left(\frac{k\pi}{2} - \frac{\pi}{4}, \frac{k\pi}{2} + \frac{\pi}{4}\right] \quad k = 0, \dots, 3. \quad (5)$$

That is, each neighbor  $j$  of  $i$  belongs to one of the four regions each spanning 90 degrees, corresponding to east (0), north (1), west (2) and south (3). When dealing with the geometric random graph, we remove the edge effects by considering the following modification, as illustrated in Fig. 2. A boundary node is a node within distance  $r$  from one of the boundaries. For an east boundary node  $i$ , we create mirror images of the neighbors of  $i$  with respect to the east boundary. If a neighbor  $j$  has an image located within the transmission range of  $i$ ,

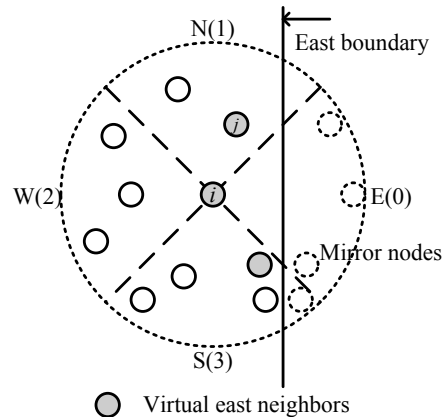


Fig. 2. Illustration of neighbor classification and virtual neighbors for boundary nodes.

node  $j$  (besides its original role) is considered as a virtual neighbor of  $i$ , with direction being determined by the image's location with respect to the location of  $i$ . Denote the set of virtual east, north, west and south neighbors of node  $i$  by  $\tilde{\mathcal{N}}_i^0, \dots, \tilde{\mathcal{N}}_i^3$ . Let  $d_i^l$  be the number of (physical and virtual) neighbors in direction  $l$ , i.e.,  $d_i^l \triangleq |\mathcal{N}_i^l| + |\tilde{\mathcal{N}}_i^l|$ . Thus, every type- $l$  neighborhood has an effective area  $\frac{\pi r^2}{4}$ .

##### B. Algorithm

The LADA algorithm works as follows. Each node  $i$  holds four pairs of values  $(y_i^l, w_i^l)$ ,  $l = 0, \dots, 3$  corresponding to the four directions. The values are initialized with

$$y_i^l(0) = x_i(0), \quad w_i^l(0) = 1, \quad l = 0, \dots, 3. \quad (6)$$

At time  $t$ , each node  $i$  broadcasts its four values. It updates its east value  $y_i^0$  with

$$y_i^0(t+1) = (1-p) \sum_{j \in \mathcal{N}_i^2} \frac{y_j^0(t)}{d_j^0} + \frac{1}{2}p (y_i^1(t) + y_i^3(t)) \quad (7)$$

where  $p = \Theta(r)$  is the probability of turning left or right, which will be justified later. If  $d_i^2 = 0$ , then  $\sum_{j \in \mathcal{N}_i^2} \frac{y_j^0(t)}{d_j^0}$  is replaced with  $y_i^2(t)$ . The north, west and south values, as well as the corresponding  $w$  values are updated in the same fashion. Node  $i$  computes its estimate of  $\bar{x}$  with

$$x_i(t) = \frac{1}{4} \sum_{l=0}^3 (y_i^l / w_i^l). \quad (8)$$

Denote  $\hat{\mathbf{y}} = [\mathbf{y}_0^T, \mathbf{y}_1^T, \mathbf{y}_2^T, \mathbf{y}_3^T]^T$ , and similarly denote  $\hat{\mathbf{w}}$ . The above iteration can be written as  $\hat{\mathbf{y}}(t+1) = \mathbf{P}_1^T \hat{\mathbf{y}}(t)$  and  $\hat{\mathbf{w}}(t+1) = \mathbf{P}_1^T \hat{\mathbf{w}}(t)$ , where  $\mathbf{P}_1$  is a stochastic matrix.

*Theorem 4.1:* The LADA algorithm converges to the true average of node values on a finite connected 2-d graph.

*Proof:* The Markov chain  $\mathbf{P}_1$  is irreducible since the underlying graph is connected. On a finite 2-d graph, we can always find some boundary node  $i$  and direction  $l$  such that  $d_i^l = 0$ . Suppose  $l = 0$ , then there exists transition from the east to the west state of  $i$ , which implies that  $\mathbf{P}_1$  is aperiodic. Therefore,  $\mathbf{P}_1$  has a unique stationary distribution  $\pi$ , which is

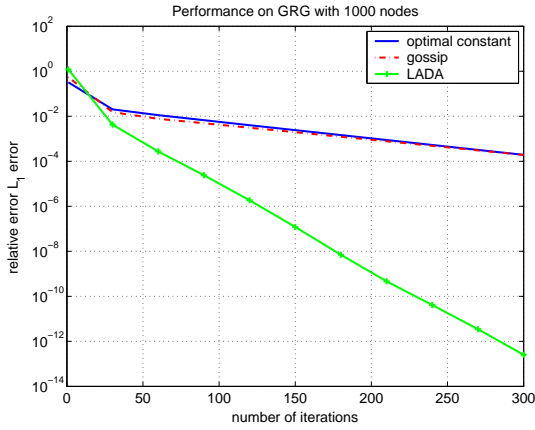


Fig. 3. Performance of distributed averaging algorithms on  $G(1000, r(1000))$

in general not uniform since  $\mathbf{P}_1$  is not doubly stochastic. It is obvious that  $\lim_{t \rightarrow \infty} \hat{\mathbf{y}}(t) = 4n\bar{x}\boldsymbol{\pi}$  and  $\lim_{t \rightarrow \infty} \hat{\mathbf{w}}(t) = 4n\boldsymbol{\pi}$ . It follows that  $\lim_{t \rightarrow \infty} \mathbf{x}(t) = \bar{x}\mathbf{1}$ . ■

It is clear that  $w_j$  serves to estimate the scaling factor  $4n\pi_j$  at each iteration. Alternatively, we may assume that sensor  $j$  has perfect knowledge of  $4n\pi_j$  (e.g. through a pre-computation stage). Then only the  $y$  values need to be communicated.

### C. Performance in Wireless Networks

To study the performance of the LADA algorithm in wireless networks modeled with a geometric random graph, we make the following modification of the updating rule for boundary nodes. If  $i$  is a west boundary node, then the term  $\sum_{j \in \mathcal{N}_i^2} \frac{y_j^0(t)}{d_j^0}$  in (7) is replaced with  $\sum_{j \in \mathcal{N}_i^2} \frac{y_j^0(t)}{d_j^0} + \sum_{j \in \tilde{\mathcal{N}}_i^2} \frac{y_j^2(t)}{d_j^2}$ , i.e. the west values of virtual west neighbors are also used to form the east value of  $i$ .

**Lemma 4.1.** On a geometric graph  $G(n, r)$ ,  $T_{\text{mix}}(\mathbf{P}_1, \epsilon) = \Theta(r^{-1} \log(1/\epsilon))$ .

*Proof:* See Appendix II. ■

**Theorem 4.2** On the geometric random graph  $G(n, r)$ , the LADA algorithm has  $\epsilon$ -averaging time  $T_{\text{ave}}(\epsilon) = \Theta(r^{-1} \log(1/\epsilon))$ .

*Proof:* See Appendix III. ■

We simulated the performance of LADA, fixed iteration with optimal constant edge weights [3], as well as randomized gossip [4] where a node chooses one of its neighbors with equal probability (for a fair comparison, the absolute averaging time of the asynchronous gossip is used). Fig. 3 illustrates the relative  $l_1$  error decay on a  $G(1000, r(1000))$ , where  $r(n) = \sqrt{\frac{2 \log n}{n}}$ , which shows that LADA significantly outperforms algorithms based on reversible chains.

## V. CONCLUSION

Motivated by the idea of accelerating mixing of random walks on a grid through lifting, we propose a Location-Aided Distributed Averaging (LADA) algorithm for general wireless networks, which constructs nonreversible random walks based on the location of the neighbors. We show that the LADA algorithm offers tremendous performance improvement over algorithms based on reversible chains.

## VI. APPENDICES

### A. Proof of Lemma 3.1

We will show that by time  $t = 6k$ , the random walk starting from any state visits every state with probability at least  $\frac{C}{4n}$  with  $C = 2^{-15}$ . Denote the state space of the chain  $\mathbf{P}$  by  $\mathcal{S}$ , and the set of east, west, north and south states respectively by  $\mathcal{S}_E, \mathcal{S}_W, \mathcal{S}_N$  and  $\mathcal{S}_S$ . For each state  $s \in \mathcal{S}$ , let  $x_s$  and  $y_s$  ( $0 \leq y_s, x_s \leq k-1$ ) respectively denote the horizontal and vertical index of the node the state corresponds to. Then a state  $s \in \mathcal{S}$  can be represented as a three tuple  $s = (s^0, s^1, s^2)$  with  $s^0 \in \{+1, -1\}$ ,  $s^1 \in \{0, 1, \dots, k-1\}$  and  $s^2 \in \{0, 1, \dots, 2k-1\}$  defined as follows: 1) if  $s \in \mathcal{S}_E \cup \mathcal{S}_W$ ,  $s^0 = 1$ ,  $s^1 = y_s$  and

$$s^2 \triangleq \begin{cases} x_s & s \in \mathcal{S}_E \\ 2k - x_s - 1 & s \in \mathcal{S}_W \end{cases}$$

2) if  $s \in \mathcal{S}_N \cup \mathcal{S}_S$ ,  $s^0 = -1$ ,  $s^1 = x_s$  and

$$s^2 \triangleq \begin{cases} y_s & s \in \mathcal{S}_N \\ 2k - y_s - 1 & s \in \mathcal{S}_S. \end{cases}$$

In other words,  $s^0$  indicates whether the state is a horizontal (east/west) or a vertical (north/south) state;  $s^1$  indicates the vertical index of a horizontal state, or the horizontal index of a vertical state;  $s^2$  indicates the particular state within the row or column specified by  $s^1$ , where states are numbered circularly. Define the function  $g(x) = \min(x, 2k - x - 1)$ . For a horizontal state,  $g$  maps the horizontal state index to the horizontal index of the corresponding node, and similarly for a vertical state. We use a similar argument as in [1], but the following analysis is a nontrivial extension of that for the path case. Without loss of generality, we assume that the chain starts from some horizontal state  $s_0 = (+1, a, b)$ . Let  $T_1, T_2, \dots$ , ( $1 \leq T_1 \leq T_2 \leq \dots$ ) be the times that the random walk makes a turn. Let  $s_t$  be the state the random walk visits at the  $t$ th step, and  $A_t$  be the number of turns made by the random walk up to time  $t$ . We must look two cases,  $A_t = 2$  and  $A_t = 3$ , with the former ensuring that any east or west state could be reached with probability at least  $\frac{C}{4n}$  at time  $6k$ , and the latter ensuring the same is true for any north or south state.

Given  $A_t = 2$ , we have  $s_t^0 = +1$ ,

$$s_t^1 = \begin{cases} g(a - T_1 + T_2 - 1 \pmod{2k}) \\ g(-a - T_1 + T_2 - 2 \pmod{2k}). \end{cases}$$

with probability 0.5 each, where the choice is determined only by the direction of the first turn, and

$$s_t^2 = \begin{cases} b + t + T_1 - T_2 - 1 \pmod{2k} \\ -b + t - T_1 - T_2 \pmod{2k} \end{cases}$$

with probability 0.5 each, where the choice is determined only by the direction of the second turn. Therefore, for any target state  $s$  with  $s^0 = +1$ , we have when  $t = 6k$ ,

$$\begin{aligned} & \Pr\{s_t = s\} \geq \Pr\{s_t = s, A_t = 2\} \\ & \geq \Pr\{a - T_1 + T_2 - 1 = s^1 \pmod{2k}, \\ & \quad -b + t - T_1 - T_2 = s^2 \pmod{2k}, A_t = 2\} \\ & \quad + \Pr\{-a - T_1 + T_2 - 2 = s^1 \pmod{2k}, \\ & \quad -b + t - T_1 - T_2 = s^2 \pmod{2k}, A_t = 2\} \\ & \geq \frac{1}{4k^2} \left(1 - \frac{1}{k}\right)^{t-2} \geq \frac{2^{-15}}{4k^2}. \end{aligned}$$

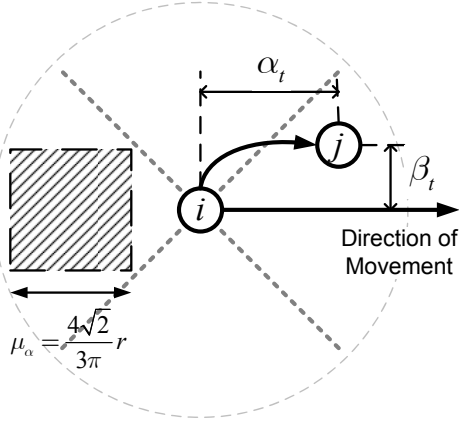


Fig. 4. Illustration of moving distances and target set

The third inequality follows from the existence of at least one pair of values  $(T_1, T_2)$  with  $1 \leq T_1 \leq T_2 \leq t$  for which both  $T_2 - T_1 = x \pmod{2k}$  and  $T_1 + T_2 = y \pmod{2k}$  are satisfied for any  $x, y$  with the same parity in the range 0 to  $2k - 1$ . (Both two cases above must be considered for the parity reason.) The last inequality is due to  $(1 - \frac{1}{k})^k \geq 1/4$  for  $k > 2$ .

Similarly, given  $A_t = 3$ , we have  $s_t^0 = -1$ , and we can write out the expressions of  $s_t^1$  (2 possibilities) and  $s_t^2$  (4 possibilities), which are independent. It can be shown that, for any target state  $s$  with  $s^0 = -1$ , there are at least  $2k$  sets of values for  $T_1, T_2$  and  $T_3$  ( $1 \leq T_1 \leq T_2 \leq T_3 \leq 6k$ ) with corresponding turning directions such that  $s_{6k} = s$ . The probability of this event is at least  $2k \frac{1}{8k^3} (1 - \frac{1}{k})^{t-3} \geq \frac{2^{-15}}{4k^2}$ . (The details are omitted due to the space constraint.)

### B. Proof of Lemma 4.1

Denote the state space of the chain  $\mathbf{P}_1$  by  $\mathcal{S}$ . A state  $s \in \mathcal{S}$  is represented as  $s = (s^0, s^1, s^2)$ , where  $s^0, s^1$  and  $s^2$  are defined in the same way as for the grid case, except that  $x_s$  and  $y_s$  now specify the horizontal and vertical coordinates instead of the indices. Similar to the grid case, we will show that by the time  $t = 6k + 1$ , for any state  $s \in \mathcal{S}$ ,  $\Pr\{s_t = s\} \geq c_1 \pi_s$  for some positive constant  $c_1$ .

Consider a movement of the random walk that keeps the direction. Denote the distance traveled in the direction of movement, and orthogonal to the direction of movement at time  $t$  respectively by  $\alpha_t$  and  $\beta_t$ , as shown in Fig. 4. Since nodes are randomly and uniformly distributed, it can be calculated that  $E(\alpha_t) = \frac{4\sqrt{2}}{3\pi}r \triangleq \mu_\alpha$ , and  $E(\beta_t) = 0$ . Let the turning probability  $p = \mu_\alpha = \Theta(r)$ , and  $k = \frac{1}{\mu_\alpha}$ . For simplicity of exposition, we assume that the random walk starts from some horizontal state  $s_0$  with  $s_0^1 = i_0 \mu_\alpha$  and  $s_0^2 = j_0 \mu_\alpha$ . We can write out the *expected* location  $E(s_t)$  of the random walk at time  $t$  by assuming that  $\alpha_t = \mu_\alpha$  and  $\beta_t = 0$  for all  $t$  in which it moves to a state of a neighboring node. Then  $E(s_t)$  depends only on the times of turns and turning directions, and evolves according to the same nonreversible chain  $\mathbf{P}$  on the grid as in Section 3. Thus, from the proof of Lemma 1, at  $t = 6k$ , for any  $s^0 \in \{+1, -1\}$ ,  $i \in \{0, 1, \dots, k-1\}$  and  $j \in \{0, 1, \dots, 2k-1\}$ ,

we have  $\Pr\{E(s_t^0) = s^0, E(s_t^1) = i\mu_\alpha, E(s_t^2) = j\mu_\alpha\} \geq \frac{2^{-15}}{4k^2}$ . Consider a target set of states  $\hat{\mathcal{S}}$  consisting of states of the same type in any given square of side  $\mu_\alpha$  in the network. Note that as  $n \rightarrow \infty$ , the conditional distribution of  $s_t^2$  given  $E(s_t^2) = j\mu_\alpha$  is simply a shifted version of that given  $E(s_t^2) = 0$ . Therefore, by the property of modulo-2 operation and the Bayesian rule, it can be shown that at  $t = 6k$ ,  $\Pr\{s_t \in \hat{\mathcal{S}}\} \geq \frac{2^{-15}}{4k^2}$ .

Now, consider for example an east state of node  $i$ , denoted by  $\hat{s}$ . The west neighboring region of  $i$  contains a square of side  $\mu_\alpha$ , and let the set of east states in this square be the target set  $\hat{\mathcal{S}}$ , as depicted in Fig. 4. By the uniform convergence in the law of large numbers [8], when  $r = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$ ,  $d_i^l \leq \frac{n\pi r^2}{2}$  for any node  $i$  and direction  $l$  w.h.p., thus we have

$$\Pr\{s_{6k+1} = \hat{s}\} \geq \frac{1}{2} \sum_{s \in \hat{\mathcal{S}}} \frac{\Pr\{s_{6k} = s\}}{d_{\max}} \geq \frac{1/2}{n\pi r^2/2} \frac{2^{-15}}{4k^2} \triangleq \frac{c_2}{4n}.$$

It can be shown that the stationary distribution of  $\mathbf{P}_1$  is approximately uniform, i.e., for any  $s \in \mathcal{S}$ ,  $\frac{c_3}{4n} \leq \pi_s \leq \frac{c_4}{4n}$  for some positive constants  $c_3$  and  $c_4$ . Therefore,  $\Pr\{s_{6k+1} = s\} \geq \frac{c_2}{c_4} \pi_s \triangleq c_1 \pi_s$ .

### C. Proof of Theorem 4.2

For any  $\epsilon \geq 0$ , from the proof of Lemma 4.1, there exists some  $\tau \triangleq T_{\text{mix}}(\mathbf{P}_1, \frac{c_2 \epsilon}{2}) = \Theta(r^{-1} \log(1/\epsilon))$ , such that for any  $t \geq \tau$ , we have  $\hat{w}_j(t) \geq c_2$  for all  $j$ , and

$$\begin{aligned} \sum_{j=1}^{4n} \left| \frac{\hat{y}_j(t)}{\hat{w}_j(t)} - \bar{x} \right| &= \sum_{j=1}^{4n} \left| \frac{\hat{y}_j(t) - \hat{w}_j(t)\bar{x}}{\hat{w}_j(t)} \right| \\ &\leq \frac{1}{c_2} \left[ \sum_{j=1}^{4n} |\hat{y}_j(t) - 4n\pi_j \bar{x}| + \sum_{j=1}^{4n} |\hat{w}_j(t) - 4n\pi_j \bar{x}| \right] \\ &\leq \frac{1}{c_2} \left( \frac{c_2 \epsilon}{2} \|\hat{\mathbf{y}}(0)\|_1 + \frac{c_2 \epsilon}{2} 4n\bar{x} \right) = \epsilon \|\hat{\mathbf{y}}(0)\|_1. \end{aligned}$$

It follows that  $\|\mathbf{x}(t) - \bar{x}\mathbf{1}\|_1 \leq \frac{1}{4} \sum_{j=1}^{4n} \left| \frac{\hat{y}_j(t)}{\hat{w}_j(t)} - \bar{x} \right| \leq \frac{1}{4} \|\hat{\mathbf{y}}(0)\|_1 \epsilon = \|\mathbf{x}(0)\|_1 \epsilon$ .

## REFERENCES

- [1] P. Diaconis, S. Holmes, and R. M. Neal, "Analysis of a non-reversible markov chain sampler," Biometrics Unit, Cornell University, Tech. Rep. BU-1385-M, 1997.
- [2] F. Chen, L. Lovász, and I. Pak, "Lifting markov chains to speed up mixing," in *31st Annual ACM Symposium on Theory of Computing (STOC'99)*, Atlanta, Georgia, May 1999.
- [3] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," in *IEEE Conf. on Decision and Control*, Maui, Hawaii, Dec. 2003.
- [4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2506–2530, 2006.
- [5] C. C. Moallemi and B. V. Roy, "Consensus propagation," *IEEE Trans. Inform. Theory*, vol. 52, no. 11, pp. 4753–4766, Nov. 2006.
- [6] 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>.
- [7] K. Jung and D. Shah, "Fast gossip via nonreversible random walk," in *IEEE Information Theory Workshop (ITW'06)*, Punta del Este, Uruguay, Mar. 2006.
- [8] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.
- [9] R. M. Neal, "Probabilistic inference using markov chain monte carlo methods," Dept. of Computer Science, University of Toronto, Tech. Rep. CRG-TR-93-1, 1993. [Online]. Available: <http://www.cs.utoronto.ca/~radford/>.