

Location-Aided Fast Distributed Consensus in Wireless Networks

Wenjun Li, *Member*, Huaiyu Dai*, *Senior Member*, and Yanbing Zhang

Abstract

Existing works on distributed consensus explore linear iterations based on *reversible* Markov chains, which contribute to the slow convergence of the algorithms. It has been observed that by overcoming the diffusive behavior of reversible chains, certain nonreversible chains lifted from reversible ones mix substantially faster than the original chains. In this paper, we investigate the idea of accelerating distributed consensus via lifting Markov chains, and propose a class of Location-Aided Distributed Averaging (LADA) algorithms for wireless networks, where nodes' coarse location information is used to construct nonreversible chains that facilitate distributed computing and cooperative processing. First, two general pseudo-algorithms are presented to illustrate the notion of distributed averaging through chain-lifting. These pseudo-algorithms are then respectively instantiated through one LADA algorithm on grid networks, and one on general wireless networks. For a $k \times k$ grid network, the proposed LADA algorithm achieves an ϵ -averaging time of $O(k \log(\epsilon^{-1}))$. Based on this algorithm, in a wireless network with transmission range r , an ϵ -averaging time of $O(r^{-1} \log(\epsilon^{-1}))$ can be attained through a centralized algorithm. Subsequently, we present a distributed LADA algorithm for wireless networks, which utilizes only the direction information of neighbors to construct nonreversible chains. It is shown that this distributed LADA algorithm achieves the same scaling law in averaging time as the centralized scheme in wireless networks for all r satisfying the connectivity requirement. The constructed chain attains the optimal scaling law in terms of an important mixing metric, the fill time, among all chains lifted from

This research was supported in part by the National Science Foundation under Grant CCF-0515164, CNS-0721815, and CCF-0830462.

W. Li is with Qualcomm Inc, San Diego, CA 92121 (e-mail: wenjunl@qualcomm.com). The work was done when she was with NC State University.

H. Dai (corresponding author) is with the ECE department of NC State University, Raleigh, NC 27695 (e-mail: huaiyu_dai@ncsu.edu).

Y. Zhang is with Broadcom Inc, Matawan, NJ 07747 (e-mail: ybzhang@broadcom.com). The work was done when he was with NC State University.

one with an approximately uniform stationary distribution on geometric random graphs. Finally, we propose a cluster-based LADA (C-LADA) algorithm, which, requiring no central coordination, provides the additional benefit of reduced message complexity compared with the distributed LADA algorithm.

Index Terms

Clustering, Distributed Computation, Distributed Consensus, Message Complexity, Mixing Time, Nonreversible Markov Chains, Time Complexity

I. INTRODUCTION

As a basic building block for networked information processing, distributed consensus underlies many important applications where information sharing and operation coordination are desired, such as distributed estimation and data fusion, cooperation of autonomous agents, as well as network optimization. Compared with the centralized counterpart, distributed algorithms do not rely on specialized routing services, scale well as the network grows, and exhibit robustness to node and link failures. Among various consensus problems, the distributed averaging problem where the consensus to be reached is over the average of node values has gained great interest [1]–[6]¹. Distributed averaging with deterministic linear iteration is studied in [1]. It is shown that under the assumption of symmetric weight matrices, the optimal linear iteration that results in the fastest convergence can be found through a semi-definite program. For linear iteration with time-varying weight matrices, convergence is guaranteed under mild conditions [2], [3]. Recently, another class of distributed consensus algorithms, the gossip algorithms, has received much interest [5], [7], [8]. Under the gossip constraint, a node can communicate with at most one node at a time. In particular, the randomized gossip algorithm studied by Boyd *et al.* [5] realizes distributed averaging through asynchronous pairwise relaxation, and its performance is governed by the second largest eigenvalue of the expectation of the independent and identically distributed random weight matrix.

Typically, governing matrices in distributed consensus algorithms are chosen to be stochastic, which connects them closely to Markov chain theory. It is also convenient to view the evolution of a Markov chain \mathbf{P} as a random walk on a graph (with vertex set V being the state space of the chain, and edge set $E = \{uv : P_{uv} > 0\}$). In both fixed and random algorithms studied in [1], [4], [5], mainly a symmetric,

¹With appropriate modification, distributed averaging algorithms can also be extended to the computation of weighted sums, linear synopses, histograms and types, and can address a large class of distributed computing and statistical inference problems.

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. It has been shown in [5] that in a wireless network of size n with a common transmission range r , the optimal gossip algorithm requires $\Theta(r^{-2} \log(\epsilon^{-1}))$ time for the relative error to be bounded by ϵ . This means that for a small radius of transmission, even the fastest gossip algorithm converges slowly.

Reversible Markov chains are dominant in research literature, as they are mathematically more tractable – see [9] and references therein. However, it is observed by Diaconis *et al.* [10] and later by Chen *et al.* [11] that certain nonreversible chains mix substantially faster than corresponding reversible chains, by overcoming the diffusive behavior of reversible random walks. This is achieved by a technique called *chain-lifting*, where states in the original chain are replicated, and transition probabilities of the new chain are designed such that, when replica states of the same original state are viewed as a single entity, the transition probabilities and the stationary distribution of the original chain are maintained. With properly chosen (typically asymmetric) transition probabilities, the resulted lifted chain could mix faster than the original chain. Motivated by this finding, as well as the close relationship between distributed consensus algorithms and Markov chains, in this paper we give an in-depth study of fast distributed averaging through chain-lifting. We propose a class of Location-Aided Distributed Averaging (LADA) algorithms that result in significantly improved averaging times compared with existing algorithms. As the name implies, the algorithms utilize (coarse) location information to construct nonreversible chains that prevent the same information being “bounced” forth and back, thus accelerating information dissemination.

A. Our Contribution

In this paper, we mainly consider synchronous algorithms, where in each time slot every node updates its values based on its neighbors’ values in the previous iteration. It may be possible to realize these algorithms in a deterministic gossip fashion, i.e., each node only communicates with one other node at a time. In that case, each effective iteration of the algorithm takes a maximum of d_{\max} rounds to complete, where d_{\max} is the maximum node degree.

We first present in Section III two generic pseudo-algorithms to illustrate the idea of distributed averaging through Markov chain lifting. Specifically, each node maintains multiple values, corresponding to multiple states lifted from a single state; the multiple values are updated using neighbors’ values in each iteration (following possibly different rules), simulating a new chain on the lifted graph. The next and more challenging step is to explicitly construct fast-mixing non-reversible chains given the network graphs. Two important types of networks, grid networks and general wireless networks modeled by

geometric random graphs, are considered in this work. For a $k \times k$ grid, we propose a LADA algorithm as an application of our Pseudo-Algorithm 1 in Section IV, and show that it takes $O(k \log(\epsilon^{-1}))$ time to reach a relative error within ϵ . Then, for the celebrated geometric random graph $G(n, r)$ with a common transmission range r , we present a centralized grid-based algorithm which exploits clustering and the LADA algorithm on a grid to achieve an ϵ -averaging time of $O(r^{-1} \log(\epsilon^{-1}))$.

While the grid-based algorithm requires a central controller or global location information to operate, the distributed LADA algorithm proposed in Section V only requires the directional information of neighbors at each local node, which is easily attainable. As an instantiation of Pseudo-Algorithm 2, the distributed LADA is associated with a Markov chain that typically does not possess a uniform stationary distribution desirable for averaging. Nevertheless, we show that the non-uniformity for the stationary distribution can be compensated by weight variables which estimate the stationary probabilities, and that the algorithm achieves an ϵ -averaging time of $O(r^{-1} \log(\epsilon^{-1}))$ with any transmission range r guaranteeing network connectivity. It is not known whether the achieved averaging time is optimal for all ϵ . Nevertheless, we demonstrate that the constructed chain does attain the optimal scaling law in terms of the fill time [12], another mixing metric, among all chains lifted from one with an approximately (on the order sense) uniform stationary distribution on $G(n, r)$. It is also important to note that the chain lifting technique used in the LADA algorithm does not increase its sensitivity to node failures. As long as the network remains connected, the neighbors of a failed node simply modify their update rules to reflect that change.

Finally, we propose a cluster-based LADA (C-LADA) variant to further improve on the message complexity in Section VI. This is motivated by the common assumption that nodes in some networks, such as wireless sensor networks, are densely deployed, where it is often more efficient to have co-located nodes clustered, effectively behaving as a single entity. In this scenario, after initiation, only inter-cluster communication and intra-cluster broadcast are needed to update the values of all nodes. Different from the centralized algorithm, clustering is performed through a distributed clustering algorithm; the induced graph is usually not a grid, so the distributed LADA algorithm, rather than the grid-based one, is suitably modified and applied. The same time complexity as LADA is achieved, but the number of messages per iteration is reduced from $\Theta(n)$ to $\Theta(r^{-2})$. Note that while most of our analysis is conducted on the geometric random graph, the LADA and C-LADA algorithms can generally be applied on any network topology.

B. Related Works

In this section we summarize some important related works and discuss the connection with our work. In the randomized gossip algorithm, at each time instant, a random node v communicates with one of its neighbors u randomly according to a given probability P_{vu} , and both nodes update their values with the average. On a geometric random graph with transmission radius $\Theta\left(\sqrt{\log n/n}\right)$, the time complexity and message complexity to reach ϵ -accuracy are respectively $\Theta\left(n \log \epsilon^{-1}/\log n\right)$ and $\Theta\left(n^2 \log \epsilon^{-1}/\log n\right)$. In comparison, our LADA algorithm reduces the time and message complexity both by a factor of $\Theta\left(\sqrt{n/\log n}\right)$.

A lot of recent works have endeavored to improve the convergence time of the standard gossip algorithm. A recent work by Moalleimi and Roy [6] proposed consensus propagation, a special form of Gaussian belief propagation, as an alternative for distributed averaging. By avoiding passing information back to where it is received, consensus propagation suppresses to some extent the diffusive nature of a reversible random walk. However, the gain of consensus propagation in time complexity over gossip algorithms quickly diminishes as the average node degrees grow, in which case the diffusive behavior is not effectively reduced.

Motivated by the observation that standard gossip algorithms can lead to significant energy waste by repeatedly circulating redundant information, the geographic gossip algorithm proposed by Dimakis *et al.* [13] reduces the message complexity by greedy geographic routing, for which an overlay network is built so that every pair of nodes can communicate. Note that such a modification entails the absolute location (coordinates) knowledge of the node itself and its neighbors, while our algorithm only requires direction knowledge of neighbors. A notable recent work by Bénézit *et al.* [14] further improves the geographic gossip algorithm by allowing averaging along routing paths. Under the box-greedy routing scheme proposed, further reduction in time and message complexity is achieved. Both the time and the message complexity of the algorithms in [14] are essentially $\Omega(n \log \epsilon^{-1})$ on geometric random graphs. In comparison, the LADA and C-LADA algorithm we propose both reduce the time complexity by a factor of $O\left(\sqrt{n \log n}\right)$, but increase the message complexity by a factor of $O\left(\sqrt{n/\log n}\right)$ and $O\left(\sqrt{n}/(\log n)^{1.5}\right)$ respectively. The optimal tradeoff between time and message complexity of distributed consensus warrants further study.

More closely related to this paper are the independent works by Jung *et al.* [15], [16], which also explored Markov chain lifting ideas for distributed averaging. The distributed algorithm in [15] adopts a nonreversible lifting construction first proposed in [11], which is based on multi-commodity flow

with minimum congestion. In their subsequent work [16], an enhanced notion called “pseudo-lifting” is introduced, where only a subset of the states in the lifted chain preserving a scaled version of the original stationary distribution are used for computation purpose. It was shown that this new chain achieves the optimal mixing time of $O(D)$, where D is the diameter of the network graph. Although the motivation is similar, our algorithms are considerably different from theirs. To construct the desired chains considered in [15], [16], each node in the network must have global knowledge of the network (in particular, the paths in the optimal multi-commodity flow that pass through the node), and the size of the state space of the lifted chain for the two approaches grow as $O(n^3)$ and $O(nD)$, respectively. On the other hand, the chain used in the LADA algorithms can be formed in a distributed fashion exploiting only local information and simple computation, and the size of the state space is linear in n . These are desirable features making our algorithm robust to node failures and amenable to distributed implementation in dynamic large-scale networks.

Savas *et al.* [17] recently proposed two distributed sequential algorithms, SIMPLE-WALK and COALESCENT, with which the transmission tokens follow a simple and a coalescing random walk respectively. The average time taken by the SIMPLE-WALK to compute the average is exactly the mean cover time of the network graph, which for a geometric random graph is $\Theta(n \log n)$ with high probability²(w.h.p.). Both algorithms require no location information, and provide a gain in message complexity at a cost of higher time complexity compared with gossip algorithms.

Finally, a few recent works investigated distributed algorithms that exploit the broadcast nature of wireless communications to accelerate convergence, bearing a similarity to C-LADA in spirit. The neighborhood gossip algorithm [18], [19] reduces the delay of each neighborhood averaging round through a computation coding scheme. In the broadcast gossip algorithm [20], all neighbors that overhear a transmission execute a local update. In the greedy gossip with eavesdropping [21], nodes eavesdrop on their neighbors’ communication and then use this information to strategically select which neighbor to gossip with.

C. Organization and Notations

Our paper is organized as follows. In Section II, we formulate the problem and review some important results in Markov chain theory. In Section III, we introduce the notion of lifting Markov chains and present two pseudo-algorithms for distributed consensus based on chain-lifting. In Section IV, the LADA

²With probability approaching 1 as $n \rightarrow \infty$.

algorithm for grid networks is proposed, which is then extended to a centralized algorithm for geometric random graphs. In Section V, we present the distributed LADA algorithm for wireless networks and analyze its performance. The C-LADA algorithm is treated in Section VI. Finally, conclusions are given in Section VII.

We use the following order notations throughout our paper. Given non-negative functions $f(n)$ and $g(n)$:

- $f(n) = O(g(n))$ and $g(n) = \Omega(f(n))$ if there exists some k and $c > 0$, such that $f(n) \leq cg(n)$ for $n \geq k$.
- $f(n) = \Theta(g(n))$ if $f(n) = O(g(n))$ as well as $f(n) = \Omega(g(n))$.
- $f(n) = o(g(n))$ and $g(n) = \omega(f(n))$ if $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0$.

II. PROBLEM FORMULATION AND PRELIMINARIES

In this section, we first formulate the problem, and then introduce some existing results needed for our analysis.

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 $x_{\text{ave}} = \frac{1}{n} \sum_{v=1}^n x_v$ denote the average. The goal is to compute x_{ave} in a distributed and robust fashion. As we mentioned, such designs are basic building blocks for distributed and cooperative information processing in wireless networks. Let $\mathbf{x}(t)$ be the vector containing node values at the t th iteration. 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

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

where $\|\mathbf{x}\|_1 = \sum_i |x_i|$ is the l_1 norm. For the more general case $\mathbf{x}(0) \in \mathbb{R}^n$, the ϵ -averaging time can be slightly modified as the earliest time such that $\|\mathbf{x}(t) - x_{\text{ave}} \mathbf{1}\|_1 \leq \epsilon \|\mathbf{x}(0) - \min_v x_v(0) \mathbf{1}\|_1$, and the results in this paper continue to hold. Note that in the literature of distributed consensus, the l_2 norm $\|\mathbf{x}\|_2 = \sqrt{\sum_i |x_i|^2}$ has also been used in measuring the averaging time [1], [5]. The two metrics are closely related. Define $T_{\text{ave},2}(\epsilon) = \sup_{\mathbf{x}(0) \in \mathbb{R}^{+n}} \inf \{t : \|\mathbf{x}(t) - x_{\text{ave}} \mathbf{1}\|_2 \leq \epsilon \|\mathbf{x}(0)\|_2\}$. It is not difficult to show that when $\epsilon = O\left(\frac{1}{n}\right)$, then $T_{\text{ave},2}(\epsilon) = O(T_{\text{ave}}(\epsilon))$.

We will mainly use the geometric random graph [22], [23] to model a wireless network in our analysis. In the 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 $r(n) \geq \sqrt{\frac{2 \log n}{n}}$ is required to ensure the graph is connected with high probability.

For a graph $G = (V, E)$, a matrix \mathbf{W} of size $|V| \times |V|$ is said to be G -conformant, if $W_{uv} \neq 0$ only if $(u, v) \in E$. Distributed averaging can be realized through fixed linear iteration in the form $\mathbf{x}(t+1) = \mathbf{W}\mathbf{x}(t)$ where \mathbf{W} is G -conformant. We are most interested in the case where the weight matrix $\mathbf{W} = \mathbf{P}^T$ and \mathbf{P} is stochastic, i.e., $P_{vu} \geq 0$, $\forall u, v$, and $\sum_u P_{vu} = 1$, $\forall v$, with which the linear iteration evolves according to the stationary Markov chain defined by the transition matrix \mathbf{P} . By letting the initial probability distribution of the chain $\mathbf{p}(0) = \mathbf{x}(0) / \sum_v x_v(0)$, and recalling that the probability distribution of the Markov chain \mathbf{P} also evolves according to $\mathbf{p}(t+1) = \mathbf{P}^T \mathbf{p}(t)$, we obtain $\mathbf{x}(t) / nx_{\text{ave}} = \mathbf{p}(t)$, and $\mathbf{x}(t) / nx_{\text{ave}} \rightarrow \boldsymbol{\pi}$, where $\boldsymbol{\pi}$ denotes the stationary distribution of \mathbf{P} . In the case where the stationary distribution is uniform, i.e., $\boldsymbol{\pi} = (1/n) \cdot \mathbf{1}$, the linear iteration converges to the average of node values.

B. Markov Chain Preliminaries

It can be expected from our previous discussion that the averaging time of the consensus algorithm is closely related to the associated chain's convergence time. In the following, we briefly review two metrics that characterize the convergence time of a Markov chain, i.e., the mixing time and the fill time. For $\epsilon > 0$, the ϵ -mixing time of an irreducible and aperiodic Markov chain \mathbf{P} with stationary distribution $\boldsymbol{\pi}$ is defined as [9]

$$T_{\text{mix}}(\mathbf{P}, \epsilon) \triangleq \sup_v \inf \left\{ t : \frac{1}{2} \|\mathbf{P}^t(v, \cdot) - \boldsymbol{\pi}\| \leq \epsilon \right\}, \quad (2)$$

where $\mathbf{P}^t(v, \cdot)$ is the v -th row of the t -step transition matrix. By the convexity of the l_1 norm, one can show that

$$T_{\text{mix}}(\mathbf{P}, \epsilon) = \sup_{\mathbf{p}(0)} \inf \{ t : \|\mathbf{p}(t) - \boldsymbol{\pi}\|_1 \leq 2\epsilon \}. \quad (3)$$

Note that given $\mathbf{p}(0) = \mathbf{e}_v^T$, where \mathbf{e}_i is the vector with 1 at the i -th position and 0 elsewhere, $\mathbf{p}(t) = \mathbf{P}^t(v, \cdot)$.

Another related metric, known as the fill time [12] (or the separate time [24]), is defined for $0 < c < 1$ as

$$T_{\text{fill}}(\mathbf{P}, c) \triangleq \sup_v \inf \{ t : \mathbf{P}^t(v, \cdot) > (1-c)\boldsymbol{\pi} \}. \quad (4)$$

For certain Markov chains, it is (relatively) easier to obtain an estimate for T_{fill} than for T_{mix} . The following lemma comes handy in establishing an upper bound for the mixing time in terms of T_{fill} , and will be used in our analysis.

Lemma 2.1: For any irreducible and aperiodic Markov chain \mathbf{P} ,

$$T_{\text{mix}}(\mathbf{P}, \epsilon) \leq \lceil \log(\epsilon^{-1}) / \log(c^{-1}) + 1 \rceil T_{\text{fill}}(\mathbf{P}, c). \quad (5)$$

Proof: The lemma follows directly from a well-known result in Markov chain theory (see the fundamental theorem in Section 3.3 of [25]). It states that for a stationary Markov chain \mathbf{P} on a finite state space with a stationary distribution $\boldsymbol{\pi}$, if there exists a constant $0 < c < 1$ such that $P(v, u) > (1 - c)\pi_u$ for all v, u , then the distribution of the chain at time t can be expressed as a mixture of the stationary distribution and another arbitrary distribution $\mathbf{r}(t)$ as

$$\mathbf{p}(t) = (1 - c^t)\boldsymbol{\pi} + c^t\mathbf{r}(t). \quad (6)$$

Thus

$$\|\mathbf{p}(t) - \boldsymbol{\pi}\|_1 = c^t\|\boldsymbol{\pi} - \mathbf{r}(t)\|_1 \leq 2c^t. \quad (7)$$

Now, for any irreducible and aperiodic chain, by (4), we have $P^\tau(v, u) > (1 - c)\pi_u$ for any v, u when $\tau > T_{\text{fill}}(\mathbf{P}, c)$. It follows from the above that for any starting distribution,

$$\frac{1}{2}\|\mathbf{p}(t) - \boldsymbol{\pi}\|_1 \leq c^{\lfloor t/T_{\text{fill}}(\mathbf{P}, c) \rfloor}, \quad (8)$$

and the desired result follows immediately by equating the right hand side of (8) with ϵ . \blacksquare

III. FAST DISTRIBUTED CONSENSUS VIA LIFTING MARKOV CHAINS

The idea of the Markov chain lifting was first investigated in [10], [11] to accelerate the mixing of Markov chains. A lifted chain is constructed by creating multiple replica states corresponding to each state in the original chain, such that the transition probabilities and stationary probabilities of the new chain conform to those of the original chain. Formally, for a given Markov chain \mathbf{P} defined on state space V with stationary probabilities $\boldsymbol{\pi}$, a chain $\tilde{\mathbf{P}}$ defined on state space \tilde{V} with stationary probability $\tilde{\boldsymbol{\pi}}$ is a lifted chain of \mathbf{P} if there is a mapping $f : \tilde{V} \rightarrow V$ such that

$$\pi_v = \sum_{\tilde{v} \in f^{-1}(v)} \tilde{\pi}_{\tilde{v}}, \quad \forall v \in V \quad (9)$$

and

$$P_{uv} = \sum_{\tilde{u} \in f^{-1}(u), \tilde{v} \in f^{-1}(v)} \frac{\tilde{\pi}_{\tilde{u}}}{\pi_u} \tilde{P}_{\tilde{u}\tilde{v}}, \quad \forall u, v \in V. \quad (10)$$

Moreover, \mathbf{P} is called a collapsed chain of $\tilde{\mathbf{P}}$.

Lifting reversible chains to certain nonreversible ones has been shown to yield significantly reduced mixing times by overcoming the diffusive behavior of the reversible chain on regular graphs [10], [11]. In particular, it is shown that the ϵ -mixing time for a constant ϵ can be reduced from $\Theta(n^2)$ to $\Theta(n)$ on an n -path, and (somewhat informally) from $\Theta(k^2)$ to $\Theta(k)$ on a $k \times k$ 2-dimensional (2-d) torus. Given the close relationship between Markov chains and distributed consensus algorithms, it is natural to ask whether the nonreversible chain-lifting technique could be used to speed up distributed consensus in wireless networks. We answer the above question in two steps. First, we show that by allowing each node to maintain multiple values, mimicking the multiple states lifted from a single state, a nonreversible chain on a lifted state space can be simulated. In this section, we provide two pseudo-algorithms to illustrate this idea. Note that in this paper, a node refers to a physical node in a network which has to be differentiated from a state (in a Markov chain). With the pseudo-algorithms in place, the second step is to explicitly construct fast-mixing non-reversible chains that result in improved averaging times compared with existing algorithms. The latter part will be treated in Section IV and V, where we provide detailed algorithms for both grid networks as well as general wireless networks modeled by geometric random graphs.

Consider a wireless network modeled as $G(V, E)$ with $|V| = n$. Let \mathbf{P} be some G -conformant ergodic chain on V with a uniform stationary distribution, and $\tilde{\mathbf{P}}$ be an ergodic chain on S lifted from \mathbf{P} according to the lifting mapping $f : S \rightarrow V$. Denote the set of lifted states for a given state $v \in V$ by $s_v^1, \dots, s_v^{b_v} \in S$. Then a procedure for distributed averaging is given in Pseudo-algorithm 1.

Algorithm 1 Pseudo-Algorithm 1

1) Initiation ($t = 0$): for each $v \in V$ and $l = 1, \dots, b_v$, set $y_v^l(0) = x_v(0)/b_v$.

2) At time t , each node $v \in V$ updates all its values:

$$y_v^l(t+1) = \sum_{u \in V} \sum_{l'=1}^{b_u} \tilde{P}(s_u^{l'}, s_v^l) y_u^{l'}(t).$$

3) Each node $v \in V$ updates its estimate of the average value:

$$x_v(t+1) = \sum_{l=1}^{b_v} y_v^l(t+1).$$

Set $t = t + 1$ and go back to step 2).

Lemma 3.1: In Pseudo-Algorithm 1, if the collapsed chain \mathbf{P} has a uniform stationary distribution on

V , then $\mathbf{x}(t) \rightarrow x_{\text{ave}}\mathbf{1}$, and the averaging time $T_{\text{ave}}(\epsilon) \leq T_{\text{mix}}(\tilde{\mathbf{P}}, \epsilon/2)$.

Proof: Let the vector \mathbf{y} contain the copies of values of all nodes, i.e., $\mathbf{y} = [\mathbf{y}_1^T, \dots, \mathbf{y}_{|V|}^T]^T$ with $\mathbf{y}_v = [y_v^1, \dots, y_v^{b_v}]^T$. From Pseudo-Algorithm 1, it can be seen that the values are updated according to the linear iteration $\mathbf{y}(t+1) = \tilde{\mathbf{P}}^T \mathbf{y}(t)$. Let $\tilde{\mathbf{p}}(t)$ be the distribution of $\tilde{\mathbf{P}}$ at time t starting from $\tilde{\mathbf{p}}(0) = \mathbf{y}(0)/nx_{\text{ave}}$, and $\tilde{\pi}$ the stationary distribution of $\tilde{\mathbf{P}}$. We have $\mathbf{y}(t) = nx_{\text{ave}}\tilde{\mathbf{p}}(t)$, and for $t \geq T_{\text{mix}}(\tilde{\mathbf{P}}, \epsilon/2)$,

$$\begin{aligned} \|\mathbf{x}(t) - x_{\text{ave}}\mathbf{1}\|_1 &= \sum_{v \in V} |x_v(t) - x_{\text{ave}}| = \sum_{v \in V} \left| \sum_{l=1}^{b_v} y_v^l - nx_{\text{ave}} \right| = \sum_{v \in V} \left| \sum_{l=1}^{b_v} \left(y_v^l - \tilde{\pi}_{s_v^l} nx_{\text{ave}} \right) \right| \\ &\leq \sum_{v \in V} \sum_{l=1}^{b_v} |y_v^l - \tilde{\pi}_{s_v^l} nx_{\text{ave}}| = nx_{\text{ave}} \sum_{s \in S} |\tilde{p}_s(t) - \tilde{\pi}_s| \leq nx_{\text{ave}}\epsilon = \epsilon \|\mathbf{x}(0)\|_1, \end{aligned}$$

where the third equality is by $\pi_v = \sum_{l=1}^{b_v} \tilde{\pi}_{s_v^l} = \frac{1}{n}$, $\forall v \in V$, the first inequality is by the triangle inequality, and the last inequality is by the definition of mixing time in (3). ■

From the above discussion, we see that for a wireless network modeled as $G = (V, E)$, as long as we can find a fast-mixing chain whose collapsed chain is G conformant and has a uniform stationary distribution on V , we automatically obtain a fast distributed averaging algorithm on G . The crux is then to design such lifted chains which are typically nonreversible to ensure fast-mixing. While the fact that the collapsed Markov chain possesses a uniform stationary distribution facilitates distributed consensus, this does not preclude the possibility of achieving consensus by lifting chains with non-uniform stationary distributions. In fact, the non-uniformity of stationary distribution can be “smoothed out” by incorporating some auxiliary variables that asymptotically estimate the stationary distribution. We will show that under mild conditions on the stationary distribution of the original chain, similar upper bounds on the averaging time can be obtained. Such a procedure allows us more flexibility in finding a fast-mixing chain on a given graph.

Similar to Pseudo-Algorithm 1, let \mathbf{P} be some G -conformant ergodic chain on V (whose stationary distribution is not necessarily uniform), and $\tilde{\mathbf{P}}$ be an ergodic chain on S lifted from \mathbf{P} according to the lifting mapping $f : S \rightarrow V$, with the set of lifted states for $v \in V$ denoted by $s_v^1, \dots, s_v^{b_v}$. Each node $v \in V$ maintains b_v pairs of values (y_v^l, w_v^l) , $l = 1, \dots, b_v$. A procedure for distributed averaging is presented in Pseudo-Algorithm 2.

Lemma 3.2: Using Pseudo-algorithm 2, $\mathbf{x}(t) \rightarrow x_{\text{ave}}\mathbf{1}$. If there exists some constant $c' > 0$ such that the stationary distribution $\pi_v \geq \frac{c'}{n}$ for all $v \in V$, then $T_{\text{ave}}(\epsilon) = O\left(\log \epsilon^{-1} T_{\text{fill}}(\tilde{\mathbf{P}}, c)\right)$ for any constant $0 < c < 1$.

Proof: Let the vector \mathbf{y} contain the copies y_v^l for all $v \in V$ and $l_v = 1, \dots, b_v$, and similarly denote \mathbf{w} . At each time instant, the values are updated with $\mathbf{y}(t+1) = \tilde{\mathbf{P}}^T \mathbf{y}(t)$ and $\mathbf{w}(t+1) = \tilde{\mathbf{P}}^T \mathbf{w}(t)$. Denote

Algorithm 2 Pseudo-Algorithm 2

- 1) Initiation ($t = 0$): for each $v \in V$ and $l = 1, \dots, b_v$, set $y_v^l(0) = x_v(0)/b_v$ and $w_v^l(0) = 1/b_v$.
- 2) At time t , each node $v \in V$ updates all its values:

$$y_v^l(t+1) = \sum_{u \in V} \sum_{l'=1}^{b_u} \tilde{P}(s_u^{l'}, s_v^l) y_u^{l'}(t)$$

$$w_v^l(t+1) = \sum_{u \in V} \sum_{l'=1}^{b_u} \tilde{P}(s_u^{l'}, s_v^l) w_u^{l'}(t).$$

- 3) Each node $v \in V$ updates its estimates of the average value:

$$x_v(t) = \frac{\sum_{l=1}^{b_v} y_v^l(t)}{\sum_{l=1}^{b_v} w_v^l(t)}.$$

Set $t = t + 1$ and go back to step 2).

the stationary distribution of $\tilde{\mathbf{P}}$ by $\tilde{\boldsymbol{\pi}}$. By a similar argument as that of Lemma 3.1, $\lim_{t \rightarrow \infty} \mathbf{y}(t) = nx_{\text{ave}} \tilde{\boldsymbol{\pi}}$ and $\lim_{t \rightarrow \infty} \mathbf{w}(t) = n\tilde{\boldsymbol{\pi}}$. It follows that $\lim_{t \rightarrow \infty} \mathbf{x}(t) = nx_{\text{ave}} \boldsymbol{\pi} / n\boldsymbol{\pi} = \lim_{t \rightarrow \infty} \mathbf{x}(t) = x_{\text{ave}} \mathbf{1}$. Let $\tilde{\mathbf{p}}(t)$ be the distribution of $\tilde{\mathbf{P}}$ at time t . For any $\epsilon > 0$ and any constant $0 < c < 1$, Lemma 2.1 says that there exists some time $\tau = O(\log \epsilon^{-1} T_{\text{fill}}(\tilde{\mathbf{P}}, c))$, such that for any $t \geq \tau$ and any initial distribution $\tilde{\mathbf{p}}(0)$,

$$\|\tilde{\mathbf{p}}(t) - \tilde{\boldsymbol{\pi}}\|_1 \leq \frac{\epsilon(1-c)c'}{2}. \quad (11)$$

Moreover, for $t \geq T_{\text{fill}}(\tilde{\mathbf{P}}, c)$, we have for $\forall v \in V$,

$$\sum_{l=1}^{b_v} w_v^l(t) \geq (1-c) \sum_{l=1}^{b_v} \tilde{\pi}_{s_v^l}(t) n = (1-c) \pi_v n \geq (1-c)c'. \quad (12)$$

Let $\tilde{\mathbf{p}}(t)$ be the distribution of $\tilde{\mathbf{P}}$ at time t starting from $\tilde{\mathbf{p}}(0) = \mathbf{y}(0)/nx_{\text{ave}}$, and $\tilde{\mathbf{p}}'(t)$ be the distribution

of $\tilde{\mathbf{P}}$ at time t starting from $\tilde{\mathbf{p}}'(0) = \mathbf{w}(0)/n$. Thus, for $\forall t \geq \tau$,

$$\begin{aligned}
\|\mathbf{x}(t) - x_{\text{ave}}\mathbf{1}\|_1 &= \sum_{v \in V} |x_v(t) - x_{\text{ave}}| \\
&= \sum_{v \in V} \left| \frac{\sum_{l=1}^{b_v} y_v^l(t)}{\sum_{l=1}^{b_v} w_v^l(t)} - x_{\text{ave}} \right| \\
&\leq \frac{1}{(1-c)c'} \sum_{v \in V} \left| \sum_{l=1}^{b_v} (y_v^l(t) - w_v^l(t)x_{\text{ave}}) \right| \\
&\leq \frac{1}{(1-c)c'} \sum_{s \in S} |\tilde{p}_s(t)nx_{\text{ave}} - \tilde{p}'_s(t)nx_{\text{ave}}| \\
&\leq \frac{nx_{\text{ave}}}{(1-c)c'} \left[\sum_{s \in S} |\tilde{p}_s(t) - \tilde{\pi}_s| + \sum_{s \in S} |\tilde{p}'_s(t) - \tilde{\pi}_s| \right] \\
&\leq \frac{nx_{\text{ave}}}{(1-c)c'} \left[\frac{\epsilon(1-c)c'}{2} + \frac{\epsilon(1-c)c'}{2} \right] = \epsilon \|\mathbf{x}(0)\|_1.
\end{aligned}$$

■

Remark: It can be seen that the auxiliary variables w 's serve to approximate the stationary distribution at each iteration. Alternatively, a pre-computation phase can be employed where each node v obtains $n \sum_{l=1}^{b_v} \tilde{\pi}_{s_v^l}$. Then only y values need to be communicated.

In the above, we have proposed two pseudo-algorithms to illustrate the idea of distributed consensus through lifting Markov chains, leaving out the details of constructing fast-mixing Markov chains. In the following two sections, we present one efficient realization for each of these two pseudo-algorithms, on regular networks and geometric random networks, respectively.

IV. LADA ALGORITHM ON GRID

In this section, we present a LADA algorithm on a $k \times k$ grid. In literature, a torus is often used to model wireless networks to simplify the analysis [5], [22]. We consider the grid structure as a more realistic model for planar networks, and explicitly deal with the edge effects. In particular, the lifting involved in the proposed design does not depend on the network size, and our algorithm can be applied to moderate-sized graphs where the torus approximation is too loose and edge effects are non-negligible. This algorithm utilizes the direction information (not the absolute geographic location) of neighbors to construct a fast-mixing Markov chain, and is a specific example of Pseudo-Algorithm 1 described in Section III. This algorithm is then extended to a centralized algorithm for a general wireless network modeled by a geometric random graph. Besides interest in its own right, results in this section will also facilitate our analysis in the following sections.

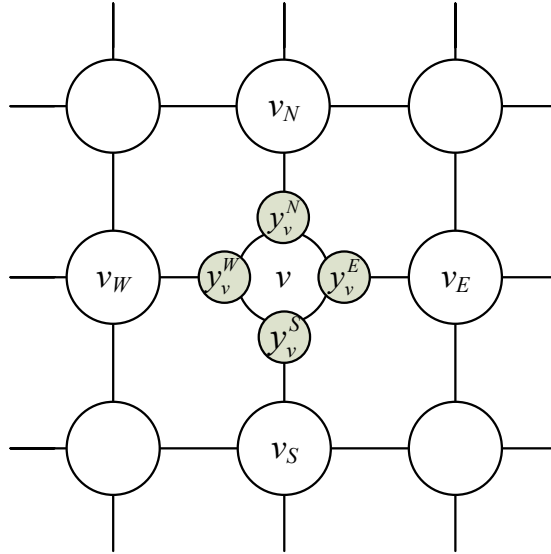


Fig. 1. Node neighbors and values in the grid

A. Algorithm

Consider a $k \times k$ grid. For each node v , denote its east, west, north and south neighbors (if they exist) respectively by v_E , v_W , v_N and v_S , as shown in Fig. 1. Each node v maintains four values corresponding to the four directions, denoted by y_v^E , y_v^W , y_v^N , and y_v^S , as shown in Fig. 1. In the rest of the paper, we denote the set of directions $L = \{E, W, N, S\}$. The values are initialized with

$$y_v^l(0) = \frac{x_v(0)}{4}, \quad l \in L. \quad (13)$$

At each time instant t , the east value of node v is updated with

$$y_v^E(t+1) = \left(1 - \frac{1}{k}\right) y_{v_w}^E(t) + \frac{1}{2k} (y_{v_n}^N(t) + y_{v_s}^S(t)). \quad (14)$$

That is, the east value of v is a weighted sum of the previous values of its west neighbor, with the majority $(1 - \frac{1}{k})$ coming from the east value, and a fraction of $\frac{1}{2k}$ coming from the north value as well as the south value. Note that the weight for the current east value of v is 0, which is typically not true for other distributed averaging algorithms. The only transitions allowed in our design are “keeping moving in the same direction” and “90 degree turns”. As will be seen in our analysis, the former allows the information to propagate fast on a 1-dimensional path, while the latter ensures the information propagates over the whole space; both combine to ensure fast mixing on a grid. Keeping a mass of a state’s own value would slow down the mixing and hence the distributed averaging. There is one exception to the above,

for $l \in \{N, W, S\}$):

$$\tilde{\mathbf{P}}((i, j, E), (i+1, j, E)) = 1 - \frac{1}{k}, \quad i < k-1 \quad (17)$$

$$\tilde{\mathbf{P}}((i, j, E), (i, j, W)) = 1 - \frac{1}{k}, \quad i = k-1 \quad (18)$$

$$\tilde{\mathbf{P}}((i, j, E), (i, j+1, N)) = \tilde{\mathbf{P}}((i, j, E), (i, j-1, S)) = \frac{1}{2k}, \quad 0 < j < k-1 \quad (19)$$

$$\tilde{\mathbf{P}}((i, j, E), (i, j, S)) = \tilde{\mathbf{P}}((i, j, E), (i, j-1, S)) = \frac{1}{2k}, \quad j = k-1 \quad (20)$$

$$\tilde{\mathbf{P}}((i, j, E), (i, j+1, N)) = \tilde{\mathbf{P}}((i, j, E), (i, j, N)) = \frac{1}{2k}, \quad j = 0. \quad (21)$$

It can be verified that $\tilde{\mathbf{P}}$ is doubly stochastic, irreducible and aperiodic. Therefore, $\tilde{\mathbf{P}}$ has a uniform stationary distribution on its state space, and so does its collapsed chain. Consequently each $x_v(t) \rightarrow x_{\text{ave}}$ by Lemma 3.1. Moreover, since the nonreversible random walk $\tilde{\mathbf{P}}$ most likely keeps its direction, occasionally makes a turn, and never turns back, it mixes substantially faster than a simple random walk (where the next node is chosen uniformly from the neighbors of the current node). Our main results on the mixing time of this chain, and the averaging time of the corresponding LADA algorithm are given below.

Lemma 4.1: The ϵ -mixing time of $\tilde{\mathbf{P}}$ is a) $T_{\text{mix}}(\tilde{\mathbf{P}}, \epsilon) = O(k \log(\epsilon^{-1}))$, for any $\epsilon > 0$;

b) $T_{\text{mix}}(\tilde{\mathbf{P}}, \epsilon) = \Theta(k)$, for a sufficiently small constant ϵ .

Proof: a) See Appendix A. The key is to show that $T_{\text{fill}} = O(k)$. The desired result then follows from Lemma 2.1.

b) We are left to show that $T_{\text{mix}}(\tilde{\mathbf{P}}, \epsilon) = \Omega(k)$ for a constant ϵ which is sufficiently small (less than $2/32$ in this case). For the random walk starting from $s_0 \in \mathcal{S}$, denote by \hat{s}_t the state it visits at time t if it never makes a turn. Note that $(1 - \frac{1}{k})^k$ is an increasing function in k , hence $(1 - \frac{1}{k})^k \geq \frac{1}{4}$ for $k \geq 2$. Thus we have for $t \leq k$,

$$\left\| \tilde{\mathbf{P}}^t(s_0, \cdot) - \frac{1}{4k^2} \cdot \mathbf{1} \right\|_1 \geq \left| \tilde{\mathbf{P}}^t(s_0, \hat{s}_t) - \frac{1}{4k^2} \right| = \left| \left(1 - \frac{1}{k}\right)^t - \frac{1}{4k^2} \right| \quad (22)$$

$$\geq \left(1 - \frac{1}{k}\right)^k - \frac{1}{4k^2} \geq \frac{1}{4} - \frac{1}{16} = \frac{3}{16} > 2\epsilon, \quad (23)$$

for $0 < \epsilon < \frac{3}{32}$, where the second inequality follows from $(1 - \frac{1}{k})^t \geq (1 - \frac{1}{k})^k \geq \frac{1}{4} \geq \frac{1}{4k^2}$. This, combined with a), yields the desired result. \blacksquare

Theorem 4.1: For the LADA algorithm on a $k \times k$ grid, a) $T_{\text{ave}}(\epsilon) = O(k \log(\epsilon^{-1}))$ for any $\epsilon > 0$;

b) $T_{\text{ave}}(\epsilon) = \Theta(k)$ for a sufficiently small constant ϵ .

Proof: a) Follows from Lemma 3.1 and Lemma 4.1 a).

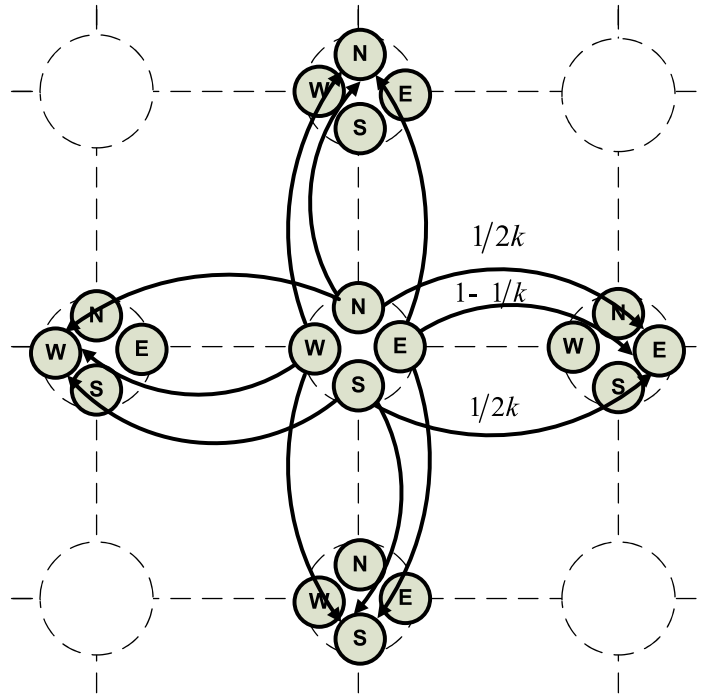


Fig. 3. Nonreversible chain used in the LADA algorithm on a grid: outgoing probabilities for the states of node v are depicted.

b) Note that the proof of Lemma 4.1 b) also implies that for $k \geq 3$, for any initial state $s_0 \in \mathcal{S}$, when $t \leq k$, there is at least one state $\hat{s} \in \mathcal{S}$ with which $\tilde{\mathbf{P}}^t(s_0, \hat{s}) \geq (1 - \frac{1}{k})^k \geq \frac{8}{27}$. Suppose state \hat{s} is some state lifted from v under the mapping f . Thus for $t \leq k$ ($k \geq 3$)

$$|x_v(t) - x_{ave}| = \left| \sum_{s \in f^{-1}(v)} \tilde{\mathbf{P}}^t(s_0, s) - \frac{1}{k^2} \right| \cdot \|\mathbf{x}(0)\|_1 \geq \left| \tilde{\mathbf{P}}^t(s_0, \hat{s}) - \frac{1}{k^2} \right| \cdot \|\mathbf{x}(0)\|_1 \geq \frac{5}{27} \|\mathbf{x}(0)\|_1, \quad (24)$$

i.e., node v has not reached an average estimate in this scenario (when $0 < \epsilon < \frac{5}{27}$). ■

C. A Centralized Grid-based Algorithm for Wireless Networks

The regular grid structure considered above does appear in some applications, and often serves as a first step towards modeling a realistic network. In this section, we explore a celebrated model for wireless networks, the geometric random graph $G(n, r)$, and present a centralized algorithm which achieves an ϵ -averaging time of $O(r^{-1} \log(\epsilon^{-1}))$ on $G(n, r)$. The algorithm relies on a central controller to perform tessellation and clustering, and simulates the LADA algorithm on the grid proposed above on the resultant

2-d grid. This is a common approach in literature (e.g., [22]), where the main purpose is to explore the best achievable performance in wireless networks, with implementation details ignored.

Assume that the unit area is tessellated into $k^2 \triangleq \lceil \frac{\sqrt{5}}{r} \rceil^2$ squares, or clusters. By this tessellation, a node in a given cluster is adjacent to all nodes in the four edge-neighboring clusters. Let n_C denote the number of nodes in a given cluster C . Then for a geometric random graph $n_C \geq 1$ for all C w.h.p. [22]. One node in each cluster is selected as a cluster-head. For each cluster C , denote its east, west, north and south neighboring cluster (if exists) respectively by C_E, C_W, C_N and C_S . Every cluster-head maintains four values corresponding to the four directions, denoted respectively by y_C^E, y_C^N, y_C^W and y_C^S for cluster C . In the initialization stage, every node transmits its value to the cluster-head. The cluster-head of cluster C computes the sum of the values within the cluster and initializes all its four values to

$$y_C^l(0) = \frac{1}{4} \sum_{v \in C} x_v(0), \quad l \in L. \quad (25)$$

At each time instant t , the cluster-heads of neighboring clusters communicate and update their values following exactly the same rules as the LADA algorithm on the grid. 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. Each node updates its estimate with

$$x_v(t+1) = \frac{k^2}{n} \sum_{l \in L} y_C^l(t+1), \quad \forall v \in C. \quad (26)$$

Theorem 4.2: The centralized algorithm has an ϵ -averaging time $T_{\text{ave}}(\epsilon) = O(r^{-1} \log(\epsilon^{-1}))$ on the geometric random graph $G(n, r)$ with common transmission radius $r > \sqrt{\frac{20 \log n}{n}}$ w.h.p. Moreover, for a sufficiently small constant ϵ , $T_{\text{ave}}(\epsilon) = \Theta(r^{-1})$.

Proof: We can appeal to the uniform convergence in the law of large numbers using Vapnik-Chervonenkis theory as in [22] to bound the number of nodes in each cluster:

$$\Pr \left(\max_{1 \leq C \leq k^2} \left| \frac{n_C}{n} - \frac{1}{k^2} \right| \leq \epsilon(n) \right) > 1 - \delta(n) \quad (27)$$

when $n \geq \max \left\{ \frac{3}{\epsilon(n)} \log \frac{16e}{\epsilon(n)}, \frac{4}{\epsilon(n)} \log \frac{2}{\delta(n)} \right\}$. This is satisfied if we choose $\epsilon(n) = \delta(n) = \frac{4 \log n}{n}$. Thus we have for all C , $n_C \geq \frac{n}{k^2} - 4 \log n = \frac{nr^2}{5} - 4 \log n$, which is at least 1 for sufficiently large n if $r > \sqrt{\frac{20 \log n}{n}}$. In this case, we have that $\frac{c_2 n}{k^2} \leq n_C \leq \frac{c_1 n}{k^2}$ for all C for some constants $c_1, c_2 > 0$ w.h.p. By Lemma 4.1 a), for any $\epsilon > 0$, there exists some $\tau = T_{\text{mix}}(\tilde{\mathbf{P}}, \frac{\epsilon}{2c_1}) = O(r^{-1} \log(\epsilon^{-1}))$ such that for all $t \geq \tau$,

$$\begin{aligned} \|\mathbf{x}(t) - x_{\text{ave}} \mathbf{1}\|_1 &= \sum_{C=1}^{k^2} n_C \left| \frac{k^2}{n} \sum_{l \in L} y_C^l(t) - x_{\text{ave}} \right| \leq \sum_{C=1}^{k^2} \frac{n_C k^2}{n} \sum_{l \in L} \left| y_C^l(t) - \frac{n x_{\text{ave}}}{4k^2} \right| \\ &\leq \epsilon \|\mathbf{x}(0)\|_1, \end{aligned}$$

where the last inequality follows a similar argument as in the proof of Lemma 3.1.

To prove the latter part of the theorem, note that $\|\mathbf{x}(t) - x_{\text{ave}}\mathbf{1}\|_1 \geq c_2 \sum_{C=1}^{k^2} |\sum_{l \in L} y_C^l(t) - \frac{nx_{\text{ave}}}{k^2}|$. The rest follows a similar argument as in the proof of Theorem 4.1 b). ■

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 the following sections, we investigate some more practical algorithms, which can be applied to wireless networks with no central controller or global knowledge available to nodes.

V. DISTRIBUTED LADA ALGORITHM FOR WIRELESS NETWORKS

In practice, distributed algorithms requiring no central coordination are generally preferred. In this section, we propose a distributed LADA algorithm for wireless networks, which is an instantiation of Pseudo-Algorithm 2 in Section III. As we mentioned, while our analysis is conducted on $G(n, r(n))$, our design can generally be applied to any network topology.

A. Neighbor Classification

As the LADA algorithm on a grid, LADA for general wireless networks utilizes coarse location information of neighbors to construct fast-mixing nonreversible chains. Due to the irregularity of node locations, a neighbor classification procedure is needed. Specifically, for each node v , we divide the plane into four quadrants with origin at v and the axis tilted by 45 degrees as shown in Fig. 4. Thus, the neighbors of node v are categorized into four subsets and denoted respectively by \mathcal{N}_v^E , \mathcal{N}_v^W , \mathcal{N}_v^N and \mathcal{N}_v^S . That is,

$$u \in \begin{cases} \mathcal{N}_v^E, & \text{if } \angle(X_u - X_v) \in (-\frac{\pi}{4}, \frac{\pi}{4}], \\ \mathcal{N}_v^W, & \text{if } \angle(X_u - X_v) \in (\frac{3\pi}{4}, \frac{5\pi}{4}], \\ \mathcal{N}_v^N, & \text{if } \angle(X_u - X_v) \in (\frac{\pi}{4}, \frac{3\pi}{4}], \\ \mathcal{N}_v^S, & \text{if } \angle(X_u - X_v) \in (\frac{5\pi}{4}, \frac{7\pi}{4}]. \end{cases} \quad (28)$$

where X_v denotes the geometric location of node v (whose accurate information is not required). Note that if $u \in \mathcal{N}_v^E$, then $v \in \mathcal{N}_u^W$, and so on. We denote the number of type l neighbors for node v by $d_v^l \triangleq |\mathcal{N}_v^l|$ (except for boundary cases discussed below).

Similar to Section IV, we consider a unit square rather than a unit torus for generality. Specifically, the edge effects are taken into account with the following modification, as illustrated in Fig. 4. A boundary node is a node within distance r from one of the boundaries, e.g., node v in Fig. 4. For a boundary node v , we create mirror images of its neighbors with respect to the boundary. If a neighbor u has an image

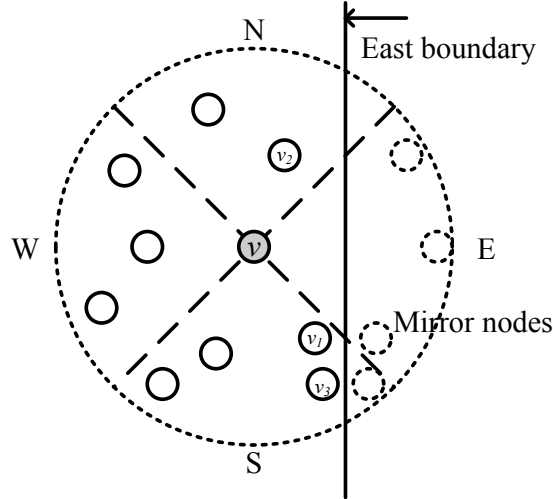


Fig. 4. Illustration of neighbor classification and virtual neighbors for boundary nodes. Note that for an east boundary node v , there can only be virtual east neighbors of the first category ($v, v_1, v_2 \in \tilde{\mathcal{N}}_v^E$), and virtual north and south neighbors of the second category ($v_3 \in \hat{\mathcal{N}}_v^S$)

located within the transmission range of v , node u (besides its original role) is considered as a virtual neighbor of v , whose direction is determined by the image's location with respect to the location of v . For example, in Fig. 4, node v_2 is both a north and a virtual east neighbor of v , and node v is a virtual east neighbor of itself. Specifically, we use $\tilde{\mathcal{N}}_v^E$ to denote the set of virtual east neighbors of an east boundary node v , and use $\hat{\mathcal{N}}_v^E$ to denote the set of virtual east neighbors of a north or south boundary node v . Similarly, $\tilde{\mathcal{N}}_v^N$ denotes the set of virtual north neighbors of a north boundary node v , and $\hat{\mathcal{N}}_v^N$ denotes that of an east or west boundary node, and so on for virtual west and south neighbors. Informally, $\tilde{}$ is used for the case the direction of the virtual neighbors and the boundary “match”, while $\hat{}$ is used for the “mismatch” scenarios. As we will see, they play different roles in the LADA algorithm. For example, in Fig. 4, we have $v, v_1, v_2 \in \tilde{\mathcal{N}}_v^E$, and $v_3 \in \hat{\mathcal{N}}_v^S$. For a boundary node v , d_v^l is instead defined as the total number of physical and virtual neighbors in direction l , i.e., $d_v^l \triangleq |\mathcal{N}_v^l| + |\tilde{\mathcal{N}}_v^l| + |\hat{\mathcal{N}}_v^l|$. With this modification, every type- l neighborhood has an effective area $\frac{\pi r^2}{4}$, hence d_v^l is roughly the same for all v and l . We also expect that as n increases, the fluctuation in d_v^l diminishes. This is summarized in the following lemma, which will be used in our subsequent analysis.

Lemma 5.1: With high probability, the number of type l neighbors of i satisfies $d_v^l = \Theta(nr^2)$ if $r > \sqrt{\frac{16 \log n}{\pi n}}$.

Proof: We can appeal to the Vapnik-Chervonenkis theory as in [22] to bound the number of nodes

in each cluster as follows:

$$\Pr \left\{ \sup_{v,l} \left| \frac{d_v^l}{n} - \frac{\pi r^2}{4} \right| \leq \frac{4 \log n}{n} \right\} > 1 - \frac{4 \log n}{n}. \quad (29)$$

Hence, we have $\left| d_v^l - \frac{n\pi r^2}{4} \right| \leq 4 \log n$ with probability at least $1 - \frac{4 \log n}{n}$ for all node i and direction l . Therefore, if $r > \sqrt{\frac{16 \log n}{\pi n}}$, we have $d_v^l = \frac{n\pi r^2}{4} \left(1 \pm O\left(\frac{\log n}{nr^2}\right) \right) = \Theta(nr^2)$. ■

The following lemma will also be useful and is straightforward to show.

Lemma 5.2: if $v \in \mathcal{N}_u^E \cup \tilde{\mathcal{N}}_u^E$, then $u \in \mathcal{N}_v^W \cup \tilde{\mathcal{N}}_v^W$, and if $v \in \tilde{\mathcal{N}}_u^E$, then $u \in \tilde{\mathcal{N}}_v^E$. Similarly for other cases.

B. Algorithm

We assume that each node has the knowledge of the number of neighboring nodes in each of the four directions, and a commonly-agreed value of the update probability $p = \Theta(r)$ through some local communication protocols. Note that the exact knowledge of a node's own location, its neighbors' location, or the total number of nodes in the network is not required. The LADA algorithm works as follows. Each node v holds four pairs of values (y_v^l, w_v^l) , $l \in L = \{E, W, N, S\}$ corresponding to the four directions. The values are initialized with

$$y_v^l(0) = \frac{x_v(0)}{4}, \quad w_v^l(0) = \frac{1}{4}, \quad l \in L. \quad (30)$$

At time t , each node i broadcasts its four values. In turn, it updates its east value y_v^E with

$$y_v^E(t+1) = \sum_{u \in \mathcal{N}_v^W} \frac{1}{d_u^E} \left[(1-p)y_u^E(t) + \frac{p}{2} (y_u^N(t) + y_u^S(t)) \right], \quad (31)$$

where $p = \Theta(r)$ is assumed. This is illustrated in Fig. 5. That is, the east value of node v is updated by a sum contributed by all its west neighbors $u \in \mathcal{N}_v^W$; each contribution is a weighted sum of the values of node u in the last slot, with the major portion $(1-p)/d_u^E$ coming from the east value, and a fraction of $p/2d_u^E$ coming from the north as well as the south value.

As in the grid case, boundary nodes must be treated specially. Let us consider two specific cases:

- 1) If v is a west boundary node (as shown in Fig. 6), then we must include an additional term

$$\sum_{u \in \tilde{\mathcal{N}}_v^W} \frac{1}{d_u^W} \left[(1-p)y_u^W(t) + \frac{p}{2} (y_u^N(t) + y_u^S(t)) \right] \quad (32)$$

in (31), i.e. values from both physical and virtual west neighbors (of the first category) are used. Moreover, for the virtual west neighbors, the west rather than east values are used. This is similar to the grid case, where the west values are bounced back and become east values when they reach

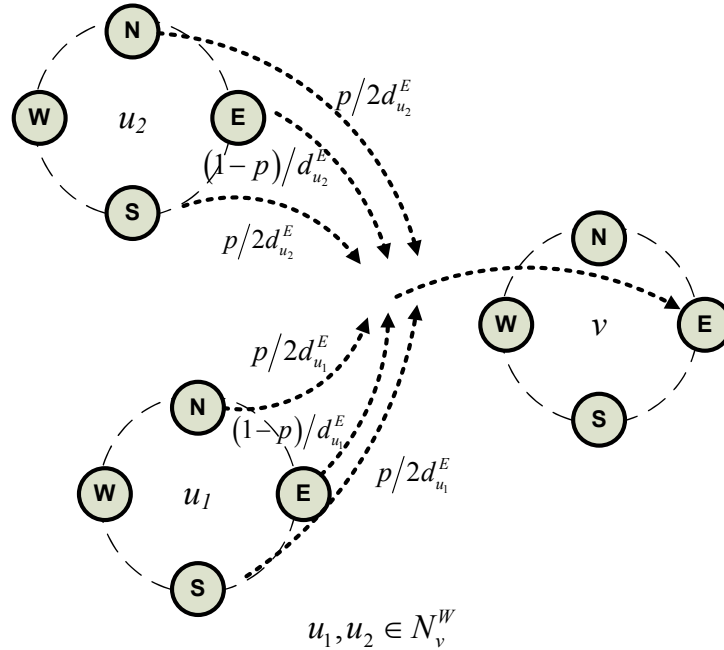


Fig. 5. Update of east value of a normal node v : weighted sums of the east, north and south values of west neighbors u_1, u_2

the west boundary, so that the information continues to propagate. The factor $\frac{1}{d_v^W}$ rather than $\frac{1}{d_u^E}$ is adopted here to ensure the outgoing probabilities of each state of each node $u \in \tilde{\mathcal{N}}_v^W$ sum to 1.

- 2) If v is a north or south boundary node (as shown in Fig. 7), however, the sum in (31) is replaced with

$$\sum_{u \in \mathcal{N}_v^W \cup \hat{\mathcal{N}}_v^W} \frac{1}{d_u^E} \left[(1-p)y_u^E(t) + \frac{p}{2} (y_u^N(t) + y_u^S(t)) \right], \quad (33)$$

i.e., the east, north and south values of both physical and virtual west neighbors (of the second category) are used. Note that $\hat{\mathcal{N}}_v^W$ are meant only for compensating the loss of neighbors for north or south boundary nodes, so unlike the previous case, their east or west values continue to propagate in the usual direction.

If v is both a west and north (or south) boundary node, both types of virtual neighbors will be involved. The purpose of introducing virtual neighbors described above is to ensure the approximate regularity of the underlying graph of the associated chain, so that the randomized effect is evenly spread out over the network. The north, west and south values, as well as the corresponding w values are updated in the same fashion. Node v computes its estimate of x_{ave} with

$$x_v(t+1) = \frac{\sum_{l \in L} y_v^l(t+1)}{\sum_{l \in L} w_v^l(t+1)}. \quad (34)$$

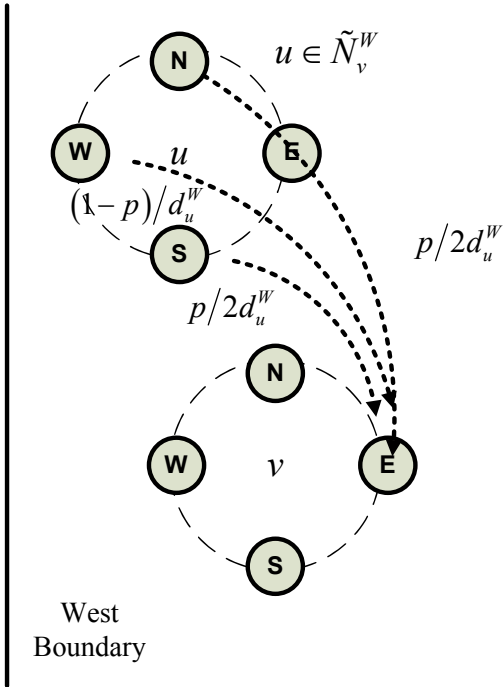


Fig. 6. Update of east value of a west boundary node v : west value of virtual west neighbor $u \in \tilde{\mathcal{N}}_v^W$ is used

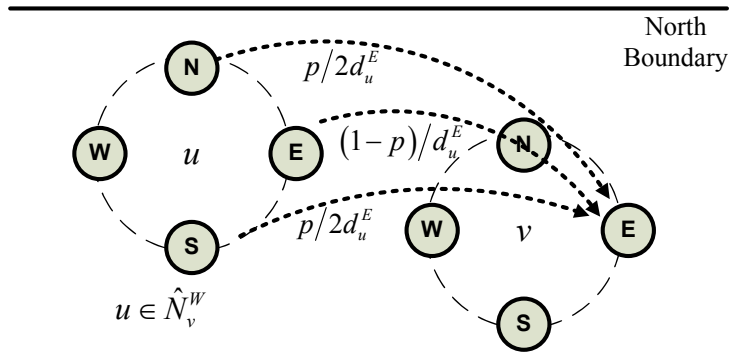


Fig. 7. Update of east value of a north boundary node v : east value of virtual west neighbor $u \in \hat{\mathcal{N}}_v^W$ is used

The algorithm is given in Algorithm 3, where for notational simplicity, the behavior of boundary nodes is not included.

Algorithm 3 LADA Algorithm

for $v = 1$ **to** n **do**

$$y_v^l(0) \leftarrow x_v(0)/4, w_v^l(0) \leftarrow 1/4, l \in \{E, W, N, S\}$$

end for

$$p \leftarrow \frac{r}{2}, t \leftarrow 0$$

while $\|\mathbf{x}(t) - x_{\text{ave}}\mathbf{1}\|_1 > \epsilon$ **do****for** $v = 1$ **to** n **do**

$$y_v^E(t+1) \leftarrow \sum_{u \in \mathcal{N}_v^W} \frac{1}{d_u^E} [(1-p)y_u^E(t) + \frac{p}{2}(y_u^N(t) + y_u^S(t))]$$

$$w_v^E(t+1) \leftarrow \sum_{u \in \mathcal{N}_v^W} \frac{1}{d_u^E} [(1-p)w_u^E(t) + \frac{p}{2}(w_u^N(t) + w_u^S(t))]$$

$$y_v^W(t+1) \leftarrow \sum_{u \in \mathcal{N}_v^E} \frac{1}{d_u^W} [(1-p)y_u^W(t) + \frac{p}{2}(y_u^N(t) + y_u^S(t))]$$

$$w_v^W(t+1) \leftarrow \sum_{u \in \mathcal{N}_v^E} \frac{1}{d_u^W} [(1-p)w_u^W(t) + \frac{p}{2}(w_u^N(t) + w_u^S(t))]$$

$$y_v^N(t+1) \leftarrow \sum_{u \in \mathcal{N}_v^S} \frac{1}{d_u^N} [(1-p)y_u^N(t) + \frac{p}{2}(y_u^E(t) + y_u^W(t))]$$

$$w_v^N(t+1) \leftarrow \sum_{u \in \mathcal{N}_v^S} \frac{1}{d_u^N} [(1-p)w_u^N(t) + \frac{p}{2}(w_u^E(t) + w_u^W(t))]$$

$$y_v^S(t+1) \leftarrow \sum_{u \in \mathcal{N}_v^N} \frac{1}{d_u^S} [(1-p)y_u^S(t) + \frac{p}{2}(y_u^E(t) + y_u^W(t))]$$

$$w_v^S(t+1) \leftarrow \sum_{u \in \mathcal{N}_v^N} \frac{1}{d_u^S} [(1-p)w_u^S(t) + \frac{p}{2}(w_u^E(t) + w_u^W(t))]$$

$$x_v(t+1) \leftarrow \frac{\sum_{l \in L} y_v^l(t+1)}{\sum_{l \in L} w_v^l(t+1)}$$

end for

$$t \leftarrow t + 1$$

end while

C. Analysis

Denote $\mathbf{y} = [\mathbf{y}_E^T, \mathbf{y}_W^T, \mathbf{y}_N^T, \mathbf{y}_S^T]^T$, with $\mathbf{y}_l = [y_1^l, y_2^l, \dots, y_n^l]^T$, and similarly denote \mathbf{w} . The above iteration can be written as $\mathbf{y}(t+1) = \tilde{\mathbf{P}}_1^T \mathbf{y}(t)$ and $\mathbf{w}(t+1) = \tilde{\mathbf{P}}_1^T \mathbf{w}(t)$. Using Lemma 5.2, it can be shown that each row in $\tilde{\mathbf{P}}_1$ (i.e., each column in $\tilde{\mathbf{P}}_1^T$) sums to 1, hence $\tilde{\mathbf{P}}_1$ is a stochastic matrix (see Fig. 8 for an illustration). On a finite connected 2-d network, the formed chain $\tilde{\mathbf{P}}_1$ is irreducible and aperiodic by construction. Due to irregularity of the network, all west neighbors of a node don't have exactly the same number of east neighbors. Consequently, the incoming probabilities of a state do not sum to 1 (see Eq. (31) and Fig. 5), i.e., $\tilde{\mathbf{P}}_1$ is not doubly stochastic and does not have a uniform stationary distribution. The LADA algorithm for general wireless networks is a special case of the Pseudo-Algorithm 2 in Section III, and it converges to the average of node values by Lemma 3.2 a). In the rest of this section, we analyze the performance of the LADA algorithm on geometric random graphs.

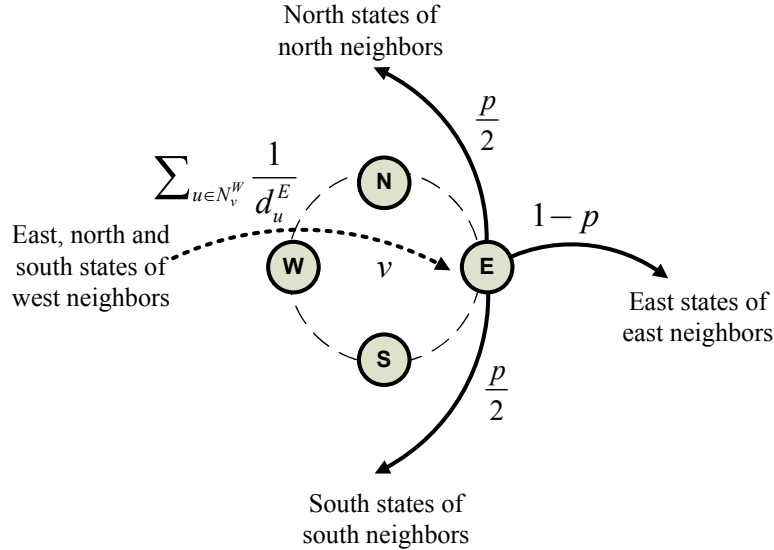


Fig. 8. The Markov chain used in LADA: combined outgoing probabilities (solid lines) and combined incoming probabilities (dotted line) for the east state of node v are depicted

Lemma 5.3: On the geometric random graph $G(n, r)$ with $r = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$, with high probability, the Markov chain $\tilde{\mathbf{P}}_1$ constructed in the LADA algorithm has an approximately uniform stationary distribution, i.e., for any $s \in \mathcal{S}$, its stationary probability $\tilde{\pi}(s) = \Theta\left(\frac{1}{n}\right)$, and $T_{\text{fill}}(\tilde{\mathbf{P}}_1, c) = O(r^{-1})$ for some constant $0 < c < 1$.

The proof is given in Appendix B. Essentially, we first consider the expected location of the random walk $\tilde{\mathbf{P}}_1$ (with respect to the node distribution), which is shown to evolve according to the random walk $\tilde{\mathbf{P}}$ on a $k \times k$ grid with $k = \Theta(r^{-1})$ when $p = \Theta(r)$. Thus the expected location of $\tilde{\mathbf{P}}_1$ can be anywhere on the grid in $O(k)$ steps (see Section IV). Then, we take the random node location into account and further show that when $n \rightarrow \infty$, the exact location of the random walk $\tilde{\mathbf{P}}_1$ can be anywhere in the network in $O(r^{-1})$ steps.

Theorem 5.1: On the geometric random graph $G(n, r)$ with $r = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$, the LADA algorithm has an ϵ -averaging time $T_{\text{ave}}(\epsilon) = O(r^{-1} \log(\epsilon^{-1}))$ with high probability.

Proof: Since when $r = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$, the Markov chain $\tilde{\mathbf{P}}_1$ constructed in the LADA algorithm has an approximately uniform stationary distribution from Lemma 5.3, so does its collapsed chain. Thus Lemma 3.2 b) can be invoked to show that $T_{\text{ave}}(\epsilon) = O\left(T_{\text{fill}}(\tilde{\mathbf{P}}_1, c) \log(\epsilon^{-1})\right) = O(r^{-1} \log(\epsilon^{-1}))$. ■

We also refer the interested reader to a variant of the LADA algorithm, called LADA-U in our previous work [26], where the underlying nonreversible chain is designed to ensure a uniform stationary distribution by allowing some diffusive behavior. It is shown that LADA-U can achieve the same scaling law in averaging time as LADA, but with a larger minimum transmission range requirement.

D. T_{fill} Optimality of LADA Algorithm

To conclude this section, we would like to discuss the following question: what is the optimal performance of distributed consensus through lifting Markov chains on a geometric random graph, and how close is the LADA performance relative to the optimum? A straightforward lower bound of the averaging time of this class of algorithms would be given by the diameter of the graph, hence $T_{ave}(\epsilon) = \Omega(r^{-1})$. Therefore, for a constant ϵ , LADA algorithm is optimal in the ϵ -averaging time. For $\epsilon = O(1/n)$, it is not known whether the lower bound $\Omega(r^{-1})$ can be further tightened, and whether LADA achieves the optimal ϵ -averaging time in scaling law. Nevertheless, we provide a partial answer to the question by showing that the constructed chain attains the optimal scaling law of $T_{fill}(\tilde{\mathbf{P}}, c)$ for a constant $c \in (0, 1)$, among all chains lifted from one with an approximately uniform stationary distribution on $G(n, r)$. For our analysis, we introduce two invariants of a Markov chain, the conductance and the resistance. The conductance measures the chance of a Markov chain \mathbf{P} leaving a set after a single step, and is defined as [27]

$$\Phi(\mathbf{P}) = \min_{S \subset V, 0 < \pi(S) < 1} \frac{Q(S, \bar{S})}{\pi(S)\pi(\bar{S})} \quad (35)$$

where \bar{S} is the complement of S in V , $Q(A, B) = \sum_{u \in A} \sum_{j \in B} Q_{uj}$, and for edge $e = uv$, $Q(e) = Q_{uv} = \pi_u P_{uv}$ is often interpreted as the capacity of the edge. The resistance is defined in terms of multi-commodity flows. A flow³ in the underlying graph $G(\mathbf{P})$ of \mathbf{P} is a function $f : \Gamma \rightarrow \mathbb{R}^+$ which satisfies

$$\sum_{\gamma \in \Gamma_{uv}} f(\gamma) = \pi(u)\pi(v) \quad \forall u, v \in V, u \neq v \quad (36)$$

where Γ_{uv} is the set of all simple directed paths from u to v in $G(\mathbf{P})$ and $\Gamma = \bigcup_{u \neq v} \Gamma_{uv}$. The congestion parameter $R(f)$ of a flow f is defined as

$$R(f) \triangleq \max_e \frac{1}{Q(e)} \sum_{\gamma \in \Gamma; \gamma \ni e} f(\gamma). \quad (37)$$

³An alternative and equivalent definition of a flow as a function on the edges of graphs can be found in [28].

The resistance of the chain \mathbf{P} is defined as the minimum value of $R(f)$ over all flows,

$$R(\mathbf{P}) = \inf_f R(f). \quad (38)$$

It has been shown that the resistance of an ergodic reversible Markov chain \mathbf{P} satisfies $R(\mathbf{P}) \leq 16T_{\text{mix}}(\mathbf{P}, 1/8)$ [27]. This result does not readily apply to nonreversible chains. Instead, the following lemma bounding T_{fill} applies to both reversible and non-reversible chains and can be proved following similar arguments as in [27].

Lemma 5.4: For any irreducible and aperiodic Markov chain \mathbf{P} , the resistance satisfies

$$T_{\text{fill}}(\mathbf{P}, c) \geq \frac{R(\mathbf{P})}{1-c}. \quad (39)$$

Lemma 5.5: For the geometric random graph $G(n, r)$ with $r = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$, the resistance of any G -conformant Markov chain with $\pi(v) = \Theta\left(\frac{1}{n}\right)$, $\forall v \in V$ satisfies the following with high probability: a) the conductance $\Phi(\mathbf{P}) = \Theta(r)$, and b) the resistance $R(\mathbf{P}) = \Omega(r^{-1})$.

Lemma 5.5 a) is known [29]. By the max-flow min-cut theorem [27], [30], the resistance R is related to the conductance Φ as $R \geq \frac{1}{\Phi}$, thus we have $R(\mathbf{P}) = \Omega(r^{-1})$ w.h.p.

Note that the resistance cannot be reduced by lifting [11]. Combining this fact with Lemma 5.4 and Lemma 5.5 yields the following result.

Theorem 5.2: Consider a chain \mathbf{P} on the geometric random graph $G(n, r) = (V, E)$ with $r = \Omega\left(\sqrt{\frac{\log n}{n}}\right)$ and $\pi(v) = \Theta\left(\frac{1}{n}\right)$, $\forall v \in V$. For any chain $\tilde{\mathbf{P}}$ lifted from \mathbf{P} and any constant $0 < c < 1$, $T_{\text{fill}}(\tilde{\mathbf{P}}, c) = \Omega(r^{-1})$ with high probability.

The above suggests that the constructed chain in LADA is optimal in the scaling law for the mixing parameter T_{fill} for any chains lifted from one with an approximately uniform stationary distribution on $G(n, r)$.

VI. CLUSTER-BASED LADA ALGORITHM FOR WIRELESS NETWORKS

In Section IV-C, we have presented a centralized algorithm, where the linear iteration is performed on the 2-d grid obtained by tessellating the geometric random graph. Only the cluster-heads are involved in the message exchange. Therefore, compared to the distributed counterpart, the centralized LADA algorithm offers an additional gain in terms of the message complexity, which translates directly into power savings for sensor nodes. However, as we have mentioned previously, the assumption of a central controller with knowledge of global coordinates might be unrealistic. This motivates us to study a more general cluster-based LADA (C-LADA) algorithm which alleviates such requirements, and still reaps the benefit of reduced message complexity.

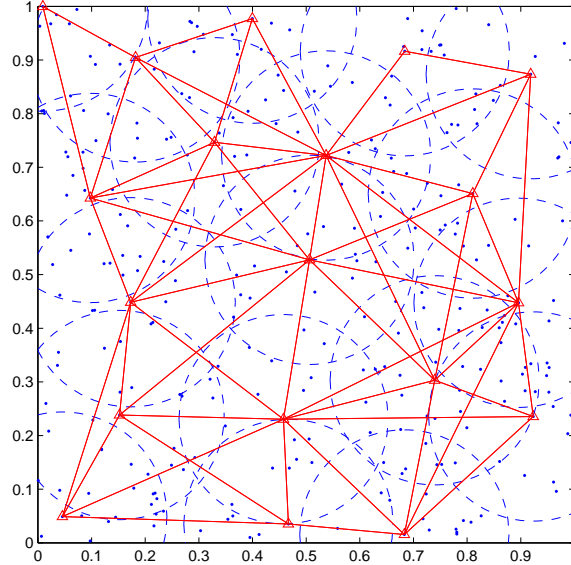


Fig. 9. Illustration of the induced graph from distributed clustering of a realization of $G(300, r(300))$. Nodes are indicated with small dots, cluster-heads are indicated with small triangles, cluster adjacency are indicated with solid lines, and the transmission range (not clusters) of cluster-heads are indicated with dashed circles.

A. C-LADA Algorithm

The idea of C-LADA can be described as follows. The nodes are first clustered using a distributed clustering algorithm given in Appendix C, where no global location information is required. Two clusters are considered adjacent (or neighbors) if there is a direct link joining them. Assume that through some local information exchange, a cluster-head knows all its neighboring clusters. In the case that two clusters are joined by more than one links, we assume that the cluster-heads of both clusters agree on one single such link to be activated. The end nodes of active links are called gateway nodes. The induced graph \tilde{G} from clustering is a graph with the vertex set consisting of all cluster-heads and the edge set obtained by joining the cluster-heads of neighboring clusters. In Fig. 9, we illustrate the induced graph as a result of applying our distributed clustering algorithm to a realization of $G(300, r(300))$, where $r(n) = \sqrt{\frac{2 \log n}{n}}$.

As can be seen, the induced graph typically has an arbitrary topology. Neighbor classification on the induced graph is based on the relative location of the cluster-heads, according to a similar rule as described in Section V-A. Let \mathcal{N}_C^l denote the set of type- l neighboring clusters (including virtual neighbors) for cluster C , and $d_C^l = |\mathcal{N}_C^l|$. It can be shown that $d_C^l \geq 1$ for any C and l w.h.p.. Let C_v be the index of the cluster node v belongs to, and n_C be the number of nodes in cluster C . It is convenient to consider another relevant graph $\hat{G} = (V, \hat{E})$ constructed from the original network graph $G = (V, E)$ as follows:

for any $u, v \in V$, $uv \in \hat{E}$ if and only if C_u and C_v are neighbors. Moreover, u is considered as a type- l neighbor of v if and only if C_u is a type- l neighboring cluster of C_v . For any cluster C , let $\hat{d}_C^l = \sum_{C' \in \mathcal{N}_C^l} n_{C'}$ be the total number of nodes in the type- l neighboring clusters of C , which is equal to the number of type- l neighbors of any node in cluster C in \hat{G} .

Every cluster-head maintains four pairs of values (y_C^l, w_C^l) , $l = 0, \dots, 3$, initialized with $y_C^l(0) = \sum_{v \in C} x_v(0)/(4n_C)$, and $w_C^l(0) = 1/4$, $l \in \{E, W, N, S\}$. At time t , the gateways nodes of neighboring clusters exchange values and forward the received values to the cluster-heads. The cluster-head of cluster C updates its east y value according to

$$y_C^E(t+1) = \sum_{C' \in \mathcal{N}_C^E} \frac{n_{C'}}{\hat{d}_C^E} \left[(1-p)y_{C'}^E(t) + \frac{p}{2} (y_{C'}^N(t) + y_{C'}^S(t)) \right], \quad (40)$$

and similarly for other y values and w values. The cluster-heads then computes the estimate of the average and broadcast to its members such that

$$x_v(t) = \sum_{l \in L} \frac{y_C^l(t)}{\sum_{l \in L} w_C^l(t)}, \quad \forall v \in C. \quad (41)$$

It can be verified that, the above C-LADA algorithm essentially simulates the LADA algorithm on graph \hat{G} with the aforementioned neighbor classification rule; for any node in cluster C , the update rule in (40) is equivalent to the update rule in (31). It follows that $\mathbf{x}(t)$ converges to $x_{\text{ave}}\mathbf{1}$ as $t \rightarrow \infty$, and C-LADA also achieves an ϵ -averaging time of $O(r^{-1} \log(\epsilon^{-1}))$ on geometric random graphs.

B. Message Complexity

Finally, we demonstrate that C-LADA further reduces the message complexity, and hence the energy consumption. For LADA, each node must broadcast its values during each iteration, hence the number of messages transmitted in each iteration is $\Theta(n)$. For C-LADA, there are three types of messages: transmissions between gateway nodes, transmissions from gateway nodes to cluster-heads and broadcasts by cluster-heads. Thus, the number of messages transmitted in each iteration is on the same order as the number of gateway nodes, which is between Kd_{\min} and Kd_{\max} , where K is the number of clusters, and d_{\min} and d_{\max} are respectively the maximum and the maximum number of neighboring clusters in the network.

Lemma 6.1: Using the Distributed Clustering Algorithm in Appendix D, the number of neighboring clusters for any cluster C satisfies $4 \leq d_C \leq 48$, and the number of clusters satisfies $\pi^{-1}r^{-2} \leq K \leq 2r^{-2}$.

Proof: The lower bound $d_C \geq 4$ follows from $d_C^l \geq 1$ for any C and l . Note that the cluster-heads are at least at a distance r from each other (see Appendix D). Hence, the circles with the cluster-heads as

the centers and radius $0.5r$ are non-overlapping. Note also that, for a cluster C , the cluster-heads of all its neighboring clusters must lie within distance $3r$ from the cluster-head of C . Within the neighborhood of radius $3.5r$ of a cluster-head, there are no more than $\left(\frac{3.5}{0.5}\right)^2$ non-overlapping circles of radius $0.5r$. This means that the number of neighboring clusters is upper bounded by 48.

Consider the tessellation of the unit square into squares of side $\frac{r}{\sqrt{2}}$. Thus, every such square contains at most one cluster-head, so there are at most $2r^{-2}$ clusters. On the other hand, in order to cover the whole unit square, there must be at least $\pi^{-1}r^{-2}$ clusters. ■

The theorem below on the message complexity follows immediately.

Theorem 6.1: The ϵ -message complexity, defined as the total number of messages transmitted in the network to achieve ϵ -accuracy, is $O(nr^{-1} \log(\epsilon^{-1}))$ for the LADA algorithm, and $O(r^{-3} \log(\epsilon^{-1}))$ for the C-LADA algorithm with high probability in the geometric random graph $G(n, r)$ with $r = \Theta(\sqrt{\log n/n})$.

It can be shown that for $r = \Theta(\sqrt{\log n/n})$, C-LADA reduces the message complexity by $\Theta(\log n)$. While maybe not very significant in the scaling law, the saving is still very important for practical implementations.

As a side note, cluster-based algorithms haven also been designed based on reversible chains [31] to reduce the message complexity.

VII. CONCLUSION

We propose a class of Location-Aided Distributed Averaging (LADA) algorithms for grid networks and wireless networks, which achieve fast convergence via constructing nonreversible lifting of Markov chains. Our algorithms can realize an ϵ -averaging time of $O(r^{-1} \log(\epsilon^{-1}))$ for all transmission range r that guarantees network connectivity, a significant improvement over existing algorithms based on reversible chains. The cluster-based LADA (C-LADA) variant requires no central controller to perform clustering, while provides the benefit of reduced message complexity.

APPENDIX

A. Proof of Lemma 4.1

We will show that by the time $t = 6k$, the random walk \mathbf{P} starting from any state visits every state with probability at least $\frac{C}{4k^2}$ for some constant $C > 0$. The desired result then follows from Lemma 2.1. Denote each state $s \in \mathcal{S}$ by a triplet $s = (c_x, c_y, l)$. with $c_x, c_y \in \{0, 1, \dots, k-1\}$ and $l \in \{E, W, N, S\}$.

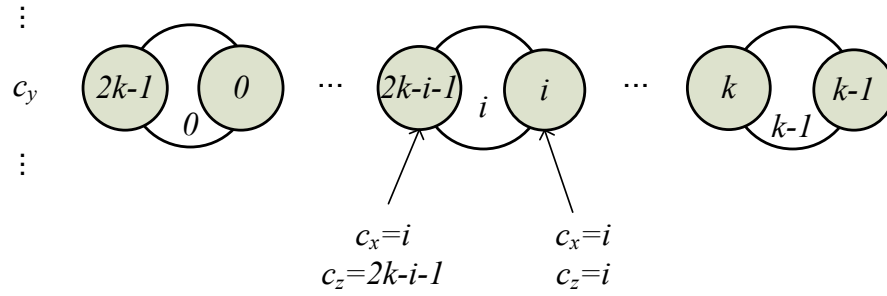


Fig. 10. Illustration of circular numbering of east and west states within a row

To facilitate the analysis, we define an auxiliary parameter c_z for a state s as follows:

$$c_z \triangleq \begin{cases} c_x, & l = \text{E} \\ 2k - c_x - 1, & l = \text{W} \\ c_y, & l = \text{N} \\ 2k - c_y - 1, & l = \text{S}. \end{cases} \quad (42)$$

For example, the numbering for east and west states in a given row is illustrated in Fig. 10. Due to the circular numbering, a horizontal movement of the random walk that keeps the direction (and bounces back at the boundary) can be written as $(c_y, c_z) \rightarrow (c_y, c_z + 1 \pmod{2k})$, and similarly for a vertical movement. Note that by defining the function

$$g(c_z) = \min(c_z, 2k - c_z - 1), \quad (43)$$

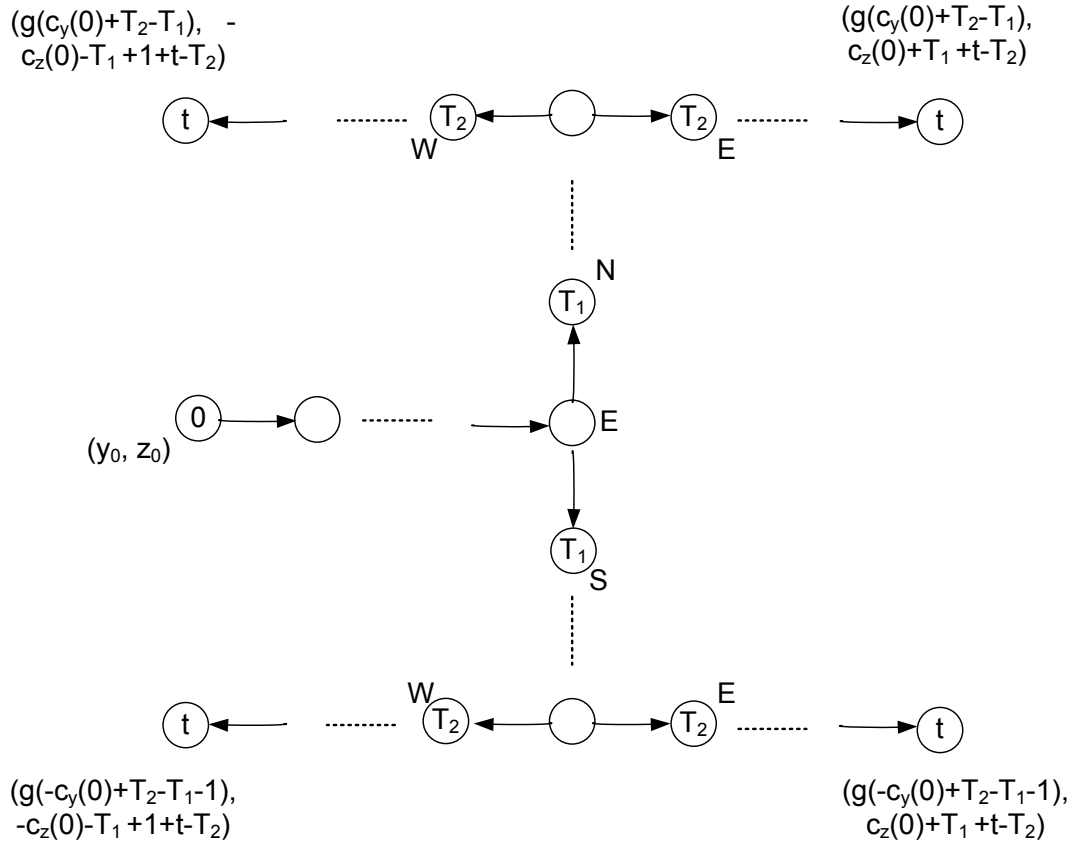
we have $g(c_z) = c_x$ when $l \in \{\text{E}, \text{W}\}$, and $g(c_z) = c_y$ when $l \in \{\text{N}, \text{S}\}$.

Without loss of generality, we assume that the chain starts from some horizontal state $s_0 = (c_x(0), c_y(0), l(0))$ with $l(0) \in \{\text{E}, \text{W}\}$. 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 (i.e., in the t -th step, the random walk goes from state s_{t-1} to s_t), and A_t be the number of turns made by the random walk up to time t . In the following, we consider two cases: (1) a target state $s = (c_x, c_y, l)$ with $l \in \{\text{E}, \text{W}\}$, i.e., a horizontal state, and (2) a target state with $l \in \{\text{N}, \text{S}\}$, i.e., a vertical state, and show that at $t = 6k$, for both cases

$$\Pr\{s_t = s\} \geq \frac{C}{4k^2}.$$

- 1) **s is a horizontal state.** In this case, we focus on $A_t = 2$ (so s_t is also a horizontal state), and show that

$$\Pr\{s_t = s\} \geq \Pr\{s_t = s, A_t = 2\} \geq \frac{C}{4k^2}. \quad (44)$$

Fig. 11. Illustration of states traversed till time t with two turns

Note that a horizontal state s is fully characterized by c_y and c_z (since $c_x = g(c_z)$). Thus, the state at time 0 can be represented as $(c_y(0), c_z(0))$, as illustrated in Fig. 11. Now, consider the state at time t . First, observe that $c_y(t)$ is determined only by the direction of the first turn at T_1 , which may be towards north or south, as illustrated by the two states labeled with T_1 in Fig. 11. If the turn is towards north, we have

$$c_y(t) = g(c_y(0) + T_2 - T_1 \pmod{2k}); \quad (45)$$

if it is towards south, we have

$$c_y(t) = g(2k - 1 - c_y(0) + T_2 - T_1 \pmod{2k}) = g(-c_y(0) + T_2 - T_1 - 1 \pmod{2k}). \quad (46)$$

Second, observe that $c_z(t)$ is determined only by the direction of the second turn at T_2 , which may be the same as the one in which the random walk is moving at time $T_1 - 1$, or the opposite. In the former case (the two east states at time T_2 shown in Fig. 11), it can be shown (by observing

the two periods $[1, T_1 - 1]$ and $[T_2, t]$ within which the random walk is traveling horizontally) that

$$c_z(t) = c_z(0) + T_1 - 1 + (t - T_2 + 1) \pmod{2k} = c_z(0) + T_1 - T_2 + t \pmod{2k}; \quad (47)$$

in the latter case (the two west states at time T_2 shown in Fig. 11), we have

$$\begin{aligned} c_z(t) &= 2k - 1 - (c_z(0) + T_1 - 1) + (t - T_2 + 1) \pmod{2k} \\ &= -c_z(0) - T_1 - T_2 + t + 1 \pmod{2k}. \end{aligned} \quad (48)$$

Therefore, we have at $t = 6k$,

$$\begin{aligned} &\Pr\{s_t = s\} \geq \Pr\{s_t = s, A_t = 2\} \\ &\geq \Pr\{g(c_y(0) + T_2 - T_1) = c_y \pmod{2k}, -c_z(0) - T_1 - T_2 + t + 1 = c_z \pmod{2k}, A_t = 2\} \\ &\quad + \Pr\{g(-c_y(0) + T_2 - T_1 - 1) = c_y \pmod{2k}, -c_z(0) - T_1 - T_2 + t + 1 = c_z \pmod{2k}, \\ &\quad A_t = 2\}, \end{aligned}$$

where the second inequality comes from picking two combinations of $c_y(t)$ and $c_z(t)$ out of the four possible combinations formed from (45) - (48). Assuming that $g(x) = x$ (the case for $g(x) = 2k - 1 - x$ can be similarly argued), and letting $a = c_y - c_z(0)$, $b = t - c_z(0) - c_z + 1$ and $c = c_y + c_y(0) + 1$, we get

$$\begin{aligned} \Pr\{s_t = s\} &\geq \Pr\{T_2 - T_1 = a \pmod{2k}, T_1 + T_2 = b \pmod{2k}, A_t = 2\} \\ &\quad + \Pr\{T_2 - T_1 = c \pmod{2k}, T_1 + T_2 = b \pmod{2k}, A_t = 2\}. \end{aligned}$$

Note that $T_2 - T_1$ and $T_1 + T_2$ must have the same parity, so we need to consider two cases: if a and b have the same parity, then there exists at least a pair of (T_1, T_2) with $1 \leq T_1 < T_2 \leq t$ (e.g., $T_1 = \frac{b-a}{2} - 1 \pmod{2k} + 1$ and $T_2 = \frac{a+b}{2} - 1 \pmod{2k} + 2k + 1$) such that $T_2 - T_1 = a \pmod{2k}$ and $T_1 + T_2 = b \pmod{2k}$ are satisfied; if a and b have different parities, then c and b must have the same parity, and there exists at least a pair of (T_1, T_2) with $1 \leq T_1 < T_2 \leq t$ such that the second set of equations above is satisfied. Either of the two cases occurs with a probability $\frac{1}{4k^2} \left(1 - \frac{1}{k}\right)^{t-2}$. Using the fact that $\left(1 - \frac{1}{k}\right)^k \geq 1/4$ for $k > 2$, at $t = 6k$ we get

$$\Pr\{s_t = s\} \geq \frac{1}{4k^2} \left(1 - \frac{1}{k}\right)^{t-2} > \frac{2^{-12}}{4k^2}. \quad (49)$$

- 2) **s is a vertical state.** We show that in this case it is sufficient to consider the case of $A_t = 3$. Similarly as above, a vertical state s is fully characterized by c_x and c_z . Note that $c_x(t)$ is only

determined by the direction of the second turn. Similar to (47) and (48) two possible values for $c_x(t)$ are given by

$$c_x(t) = \begin{cases} g(c_z(0) + T_1 - T_2 + T_3 - 1 \pmod{2k}) \\ g(-c_z(0) - T_1 - T_2 + T_3 \pmod{2k}). \end{cases} \quad (50)$$

Also $c_z(t)$ is only determined by the direction of the first turn and third turn. It can be shown that the four possible values of $c_z(t)$ are given by

$$c_z(t) = \begin{cases} c_y(0) + t - T_1 + T_2 - T_3 + 1 \pmod{2k} \\ -c_y(0) + t + T_1 - T_2 - T_3 \pmod{2k} \\ -c_y(0) + t - T_1 + T_2 - T_3 \pmod{2k} \\ c_y(0) + t + T_1 - T_2 - T_3 + 1 \pmod{2k}. \end{cases} \quad (51)$$

Therefore,

$$\begin{aligned} & \Pr\{s_t = s\} \geq \Pr\{s_t = s, A_t = 3\} \\ & \geq \Pr\{c_z(0) + T_1 - T_2 + T_3 - 1 = c_x \pmod{2k}, \\ & \quad c_y(0) + t + T_1 - T_2 - T_3 + 1 = c_z \pmod{2k}, A_t = 3\} \\ & \quad + \Pr\{c_z(0) + T_1 - T_2 + T_3 - 1 = c_x \pmod{2k}, \\ & \quad -c_y(0) + t + T_1 - T_2 - T_3 = c_z \pmod{2k}, A_t = 3\} \\ & = \Pr\{T_3 - (T_2 - T_1) = a \pmod{2k}, \quad T_3 + (T_2 - T_1) = b \pmod{2k}, A_t = 3\} \quad (52) \\ & \quad + \Pr\{T_3 - (T_2 - T_1) = a \pmod{2k}, \quad T_3 + (T_2 - T_1) = c \pmod{2k}, A_t = 3\}, \quad (53) \end{aligned}$$

where the second inequality comes from picking two combinations out of eight possible combinations formed from (50) and (51), and in the last inequality, we have substituted $a = c_x - c_z(0) + 1$, $b = c_y(0) + t - c_z + 1$ and $c = -c_y(0) + t - c_z$. Same as 1), we must consider two cases on parity.

For a and b with the same parity, consider the $2k$ triplets of (T_1, T_2, T_3) given by

$$\left(T_1, \frac{b-a}{2} - 1 \pmod{2k} + 1 + T_1, \frac{b+a}{2} - 1 \pmod{2k} + 1 + 4k \right), \quad T_1 = 1, 2, \dots, 2k.$$

It is obvious that any such triplet satisfies $1 \leq T_1 < T_2 < T_3 \leq 6k$, as well as the conditions in (52). For a and b with different parity, a and c must have the same parity, and similarly there exists at least $2k$ valid triplets of (T_1, T_2, T_3) satisfying the conditions in (53). Thus, for any target vertical state s , we can always find $2k$ turning times (T_1, T_2, T_3) with proper turning directions to reach s at $t = 6k$ with probability

$$\Pr\{s_t = s\} \geq 2k \cdot \frac{1}{8k^3} \left(1 - \frac{1}{k}\right)^{t-3} > \frac{2^{-12}}{4k^2}. \quad (54)$$

This completes the proof.

B. Proof of Lemma 5.3

Assume the unit square is coordinated by (c_x, c_y) with $c_x, c_y \in [0, 1]$, starting from the south-west corner. Denote the state space of the chain $\tilde{\mathbf{P}}_1$ by \mathcal{S} . A state $s \in \mathcal{S}$ is represented with a triplet $s = (c_x, c_y, l)$ following the grid case in Appendix A. Define an auxiliary parameter c_z for a state s as follows:

$$c_z \triangleq \begin{cases} c_x, & l = \text{E} \\ 2 - c_x, & l = \text{W} \\ c_y, & l = \text{N} \\ 2 - c_y, & l = \text{S}. \end{cases}$$

We will show that by the time $t = 6k + 1$, for any state $s \in \mathcal{S}$, $\Pr\{s_t = s\} \geq C_1 \tilde{\pi}(s)$ for some positive constant C_1 .

Consider a movement of the random walk. Denote the distance traveled in the direction of movement, and that orthogonal to the direction of movement at time t respectively by α_t and β_t , as shown in Fig. 12. Since nodes are randomly and uniformly distributed and the transition probability is uniform for all neighbors in the same direction, we can calculate the expected value of α_t and β_t (with respect to the node distribution) as follows:

$$\mathbb{E}(\alpha_t) = \frac{4}{\pi r^2} \int_{-\pi/4}^{\pi/4} \int_0^r x^2 \cos \theta \, dx \, d\theta = \frac{4\sqrt{2}}{3\pi} r \triangleq \mu_\alpha, \quad (55)$$

$$\mathbb{E}(\beta_t) = \frac{4}{\pi r^2} \int_{-\pi/4}^{\pi/4} \int_0^r x^2 \sin \theta \, dx \, d\theta = 0. \quad (56)$$

Similarly, their second-order moments can be readily computed as

$$\mathbb{E}(\alpha_t^2) = \frac{4}{\pi r^2} \int_{-\pi/4}^{\pi/4} \int_0^r x^3 \cos^2 \theta \, dx \, d\theta = \frac{\pi + \sqrt{2}}{4\pi} r^2, \quad (57)$$

$$\mathbb{E}(\beta_t^2) = \frac{4}{\pi r^2} \int_{-\pi/4}^{\pi/4} \int_0^r x^3 \sin^2 \theta \, dx \, d\theta = \frac{\pi - \sqrt{2}}{4\pi} r^2, \quad (58)$$

and the variances of α_t and β_t are given by

$$\left(\frac{\pi + \sqrt{2}}{4\pi} - \frac{32}{9\pi^2} \right) r^2 \triangleq \sigma_\alpha^2, \quad (59)$$

$$\frac{\pi - \sqrt{2}}{4\pi} r^2 \triangleq \sigma_\beta^2. \quad (60)$$

Note that α_t and β_t are uncorrelated, i.e.,

$$\mathbb{E}((\alpha_t - \mu_\alpha)\beta_t) = \mathbb{E}(\alpha_t\beta_t) = \frac{4}{\pi r^2} \int_{-\pi/4}^{\pi/4} \int_0^r x^3 \cos \theta \sin \theta \, dx \, d\theta = 0. \quad (61)$$

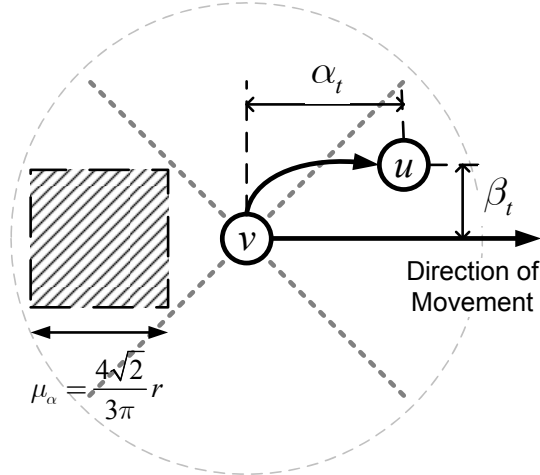


Fig. 12. Illustration of moving distances and target set

In the following, we assume $k = \lceil \frac{1}{\mu_\alpha} \rceil$ and the turning probability $p = \frac{1}{k} = \Theta(r)$.

Without loss of generality, we assume that the random walk starts from some arbitrary horizontal state $s_0 = (c_x(0), c_y(0), l(0))$ with $l(0) \in \{E, W\}$. Recall that a horizontal node is completely characterized by c_y and c_z . For simplicity of discussion, also assume that $c_y(0) = a_0\mu_\alpha$ for some $a_0 \in \{0, 1, \dots, k-1\}$ and the corresponding $c_z(0) = b_0\mu_\alpha$ for some $b_0 \in \{0, 1, \dots, 2k-1\}$ (the proof is essentially the same for non-integer a_0 and b_0 , with a little more complicated notation). Similar to Appendix A, we need to consider two cases: the target state s being a horizontal state and the target state s being a vertical state. In the following, we will focus on the former case, and the proof for the latter case is similar.

First consider the *expected* location $\mathbb{E}(s_t)$ of the random walk at t . It depends only on the turning times and turning directions, and evolves according to the random walk $\tilde{\mathbf{P}}$ on the $k \times k$ grid (see Section IV)⁴. Thus, according to Appendix A, at $t = 6k$, for any $a' \in \{0, 1, \dots, k-1\}$ and $b' \in \{0, 1, \dots, 2k-1\}$, we have

$$\Pr\{\mathbb{E}(c_y(t)) = a'\mu_\alpha, \mathbb{E}(c_z(t)) = b'\mu_\alpha\} \geq \Pr\{\mathbb{E}(c_y(t)) = a'\mu_\alpha, \mathbb{E}(c_z(t)) = b'\mu_\alpha, A_t = 2\} \geq \frac{C_2}{4k^2} \quad (62)$$

for some $C_2 > 0$.

In order to obtain a lower bound for the probability of reaching a target horizontal state s at $t = 6k+1$, we first obtain a lower bound for the probability of reaching any ancestor of s in the underlying graph

⁴If $p = \frac{C}{k}$ for some positive $C \neq 1$, then the expected location would evolve according to another chain which differs from $\tilde{\mathbf{P}}$ only in the turning probability, and has the same scaling law in the mixing time as $\tilde{\mathbf{P}}$.

of the chain at $t = 6k$. For example, consider an east state s of node v as in Fig. 12. Note that the effective west neighboring region of node v covers a circular sector of 90 degrees (for boundary nodes virtual neighbors are considered). It can be shown that such a circular sector contains a square of side μ_α as depicted in Fig. 12. Denote the set of east states in $\mathcal{N}_v^W \cup \widehat{\mathcal{N}}_v^W$ and west states in $\widetilde{\mathcal{N}}_v^W$ in this square by $\hat{\mathcal{S}} = \{\hat{s} : \hat{c}_y \in \hat{C}_y, \hat{c}_z \in \hat{C}_z, l \in \{E, W\}\}$, where generally for a non-boundary node, we have $\hat{C}_y = [a\mu_\alpha, (a+1)\mu_\alpha]$ and $\hat{C}_z = [b\mu_\alpha, (b+1)\mu_\alpha]$ for some $a \in [0, k-2]$ and $b \in [0, 2k-2]$, and $\hat{l} = l$ (the direction of the target state)⁵. In the following, we assume v is not a boundary node for simplicity, but the proof extends easily to the boundary nodes.

We claim that at $t = 6k$,

$$\sum_{a'=0}^{k-1} \sum_{b'=0}^{2k-1} \Pr \left\{ s_t \in \hat{\mathcal{S}} \mid \mathbb{E}(c_y(t)) = a'\mu_\alpha, \mathbb{E}(c_z(t)) = b'\mu_\alpha, A_t = 2 \right\} \geq C' \quad (63)$$

for some constant C' w.h.p. Based on this result and (62), we have at $t = 6k$,

$$\begin{aligned} \Pr\{s_t \in \hat{\mathcal{S}}\} &\geq \sum_{a'=0}^{k-1} \sum_{b'=0}^{2k-1} \Pr \left\{ s_t \in \hat{\mathcal{S}} \mid \mathbb{E}(c_y(t)) = a'\mu_\alpha, \mathbb{E}(c_z(t)) = b'\mu_\alpha, A_t = 2 \right\} \\ &\cdot \Pr\{A_t = 2, \mathbb{E}(c_y(t)) = a'\mu_\alpha, \mathbb{E}(c_z(t)) = b'\mu_\alpha\} \geq \frac{C' C_2}{4k^2}. \end{aligned} \quad (64)$$

By Lemma 5.1, when $r > \sqrt{\frac{16 \log n}{\pi n}}$, $d_{\max} \triangleq \max_{v,l} d_v^l \leq C_3 n r^2$ for some constant $C_3 > 0$ w.h.p., thus we have

$$\Pr\{s_{6k+1} = s\} \geq \frac{1}{2} \sum_{\hat{s} \in \hat{\mathcal{S}}} \frac{\Pr\{s_{6k} = \hat{s}\}}{d_{\max}} \geq \frac{1/2}{C_3 n r^2} \frac{C' C_2}{4k^2} \triangleq \frac{C_4}{4n}. \quad (65)$$

Note that, the random walk $\tilde{\mathbf{P}}$ has a uniform stationary distribution on the $k \times k$ grid. Using the argument as above, it can be shown that for any set $\hat{\mathcal{S}}$ containing states of the same type in a square of side μ_α , the stationary probability of $\tilde{\mathbf{P}}_1$ satisfies $\tilde{\pi}(\hat{\mathcal{S}}) = \frac{1}{4k^2}$, and consequently the stationary probability of any state of $\tilde{\mathbf{P}}_1$ is lower bounded by $\frac{C_5}{4n}$ for some $C_5 > 0$ (c.f.(65)). For an upper bound, note that in Fig. 12 the effective west neighboring region of v is also contained in an area A consisting of 2×3 squares of side μ_α . Let \mathcal{S}^E , \mathcal{S}^N and \mathcal{S}^S respectively denote the set of east states⁶, the set of north states and the set of south states of west neighbors of v that lie in A . By Lemma 5.1, when $r > \sqrt{\frac{16 \log n}{\pi n}}$,

⁵In the above example, if v is a west boundary node, then the square under consideration is folded along the west boundary, such that $\hat{C}_z = [0, (1-b)\mu_\alpha] \cup [2-b\mu_\alpha, 2]$ for some $b \in (0, 1)$, with the latter corresponding to west states of nodes in $\widetilde{\mathcal{N}}_v^W$. Note that in all cases, both \hat{C}_z and $\widetilde{\hat{C}}_z$ consist of intervals with a total length μ_α .

⁶For nodes in $\widetilde{\mathcal{N}}_v^W$, their west states are considered instead.

$d_{\min} \triangleq \min_{i,l} d_v^l \geq C_6 n r^2$ w.h.p. Hence for any state s ,

$$\tilde{\pi}(s) \leq (1-p) \sum_{s \in \mathcal{S}^E} \frac{\tilde{\pi}(s)}{d_{\min}} + \frac{p}{2} \left[\sum_{s \in \mathcal{S}^N} \frac{\tilde{\pi}(s)}{d_{\min}} + \sum_{s \in \mathcal{S}^S} \frac{\tilde{\pi}(s)}{d_{\min}} \right] \leq \left[(1-p) + \frac{p}{2} \cdot 2 \right] \frac{1}{C_6 n r^2} \cdot \frac{6}{4k^2} \triangleq \frac{C_7}{4n}. \quad (66)$$

We conclude that the stationary distribution of $\tilde{\mathbf{P}}_1$ is approximately uniform, i.e., for any $s \in \mathcal{S}$, $\frac{C_5}{4n} \leq \tilde{\pi}(s) \leq \frac{C_7}{4n}$ for some $C_5, C_7 > 0$. It follows from (65) that $\Pr\{s_{6k+1} = s\} \geq \frac{C_4}{C_7} \tilde{\pi}(s) \triangleq C_1 \tilde{\pi}(s)$ w.h.p., which implies that the fill time of $\tilde{\mathbf{P}}_1$ is $T_{\text{fill}}(\tilde{\mathbf{P}}_1, \epsilon) = O(r^{-1})$ w.h.p.

We are left to verify the claim (63). It is sufficient to consider the case that the random walk makes two turns in first $6k$ steps, with the turning times T_1 and T_2 . Denote the distance vector traveled at the t -th step by

$$\Lambda_t \triangleq \begin{cases} [\alpha_t \ \beta_t]^T & t \in [1, T_1) \cup [T_2, 6k] \\ [\beta_t \ \alpha_t]^T & t \in [T_1, T_2), \end{cases} \quad (67)$$

with mean

$$\mathbb{E}(\Lambda_t) \triangleq \mu_\Lambda = \begin{cases} [\mu_\alpha \ 0]^T & t \in [1, T_1) \cup [T_2, 6k] \\ [0 \ \mu_\alpha]^T & t \in [T_1, T_2), \end{cases} \quad (68)$$

and covariance matrix (note α_t and β_t are uncorrelated)

$$\Sigma_\Lambda = \begin{cases} \begin{bmatrix} \sigma_\alpha^2 & 0 \\ 0 & \sigma_\beta^2 \end{bmatrix} & t \in [1, T_1) \cup [T_2, 6k] \\ \begin{bmatrix} \sigma_\beta^2 & 0 \\ 0 & \sigma_\alpha^2 \end{bmatrix} & t \in [T_1, T_2). \end{cases} \quad (69)$$

As the distance vectors in different steps are independent, the covariance matrix of the total distance vector $\Lambda = \sum_{t=1}^{6k} \Lambda_t$ is given by

$$\Sigma_{\Lambda|T_1, T_2} = \begin{pmatrix} \sigma_{\alpha|T_1, T_2}^2 & 0 \\ 0 & \sigma_{\beta|T_1, T_2}^2 \end{pmatrix}, \quad (70)$$

where

$$\sigma_{\alpha|T_1, T_2}^2 = [T_1 + (6k - T_2)]\sigma_\alpha^2 + (T_2 - T_1)\sigma_\beta^2 = (\sigma_\beta^2 - \sigma_\alpha^2)(T_2 - T_1) + 6k\sigma_\alpha^2 \quad (71)$$

and

$$\sigma_{\beta|T_1, T_2}^2 = [T_1 + (6k - T_2)]\sigma_\beta^2 + (T_2 - T_1)\sigma_\alpha^2 = (\sigma_\alpha^2 - \sigma_\beta^2)(T_2 - T_1) + 6k\sigma_\beta^2 \quad (72)$$

are the respective variance of the total distance traveled horizontally and vertically in $6k$ steps. As $\sigma_\beta^2 > \sigma_\alpha^2$, it is easy to verify that the maximum of $\sigma_{\alpha|T_1, T_2}^2$ and $\sigma_{\beta|T_1, T_2}^2$ (with respect to T_1 and T_2) are

the same:

$$\sigma_{\alpha,\max}^2 = \sigma_{\beta,\max}^2 = \sigma_{\alpha}^2 + (6k - 1)\sigma_{\beta}^2. \quad (73)$$

Let

$$\Lambda_{k,t} \triangleq \Sigma_{\Lambda|T_1,T_2}^{-1/2} (\Lambda_t - \mu_{\Lambda}) = \begin{cases} \begin{bmatrix} (\alpha_t - \mu_{\alpha})/\sigma_{\alpha|T_1,T_2} \\ \beta_t/\sigma_{\beta|T_1,T_2} \end{bmatrix} & t \in [1, T_1] \cup [T_2, 6k] \\ \begin{bmatrix} \beta_t/\sigma_{\alpha|T_1,T_2} \\ (\alpha_t - \mu_{\alpha})/\sigma_{\beta|T_1,T_2} \end{bmatrix} & t \in [T_1, T_2], \end{cases} \quad (74)$$

we have $\mathbb{E}(\Lambda_{k,t}) = \mathbf{0}$ and $\lim_{n \rightarrow \infty} \sum_{t=1}^{6k} \mathbb{E}(\Lambda_{k,t} \Lambda_{k,t}^T) = \mathbf{I}$, where \mathbf{I} is the 2×2 identity matrix. In addition, by defining $\mathbb{E}(Y; C) = \mathbb{E}(Y 1_C)$ with 1_C being the indicator function of C , for any $\epsilon > 0$

$$\lim_{n \rightarrow \infty} \sum_{t=1}^{6k} \mathbb{E}(|\Lambda_{k,t}|^2; |\Lambda_{k,t}| > \epsilon) = 0, \quad (75)$$

since $|\Lambda_{k,t}|$ is always less than ϵ when n is sufficiently large such that $\frac{r}{\max\{\sigma_{\alpha|T_1,T_2}, \sigma_{\beta|T_1,T_2}\}} < \epsilon/2$. Then according to the multivariate Lindeberg-Feller Theorem ([32] Proposition 2.27), the conditional probability density function (PDF) of

$$\sum_{t=1}^{6k} \Lambda_{k,t} = \Sigma_{\Lambda|T_1,T_2}^{-1/2} \sum_{t=1}^{6k} (\Lambda_t - \mu_{\Lambda}) = \begin{bmatrix} (c_z(6k) - \mathbb{E}(c_z(6k)))/\sigma_{\alpha|T_1,T_2} \\ (c_y(6k) - \mathbb{E}(c_y(6k)))/\sigma_{\beta|T_1,T_2} \end{bmatrix}, \quad (76)$$

given T_1 and T_2 ⁷ converges in distribution to the standard multivariate normal distribution $\mathcal{N}(0, \mathbf{I})$.

Suppose $\mathcal{T}_{\{a', b'\}}$ is the set of turning times combination that result in $\mathbb{E}(c_z(t)) = b'\mu_{\alpha}$, $\mathbb{E}(c_y(t)) = a'\mu_{\alpha}$, and

$$\{T_{1,\{a', b'\}}, T_{2,\{a', b'\}}\} = \operatorname{argmin}_{\{T_1, T_2\} \in \mathcal{T}_{\{a', b'\}}} \Pr\{c_z(t) \in [b\mu_{\alpha}, (b+1)\mu_{\alpha}), c_y(t) \in [a\mu_{\alpha}, (a+1)\mu_{\alpha}) \mid T_1, T_2\}$$

for any $a \in [0, k-2]$ and $b \in [0, 2k-2]$. Define

$$\Pi(X; \Lambda, \Sigma) = \frac{1}{2\pi\sqrt{|\Sigma|}} \exp\left\{-\frac{1}{2}(X - \Lambda)^T \Sigma^{-1}(X - \Lambda)\right\}$$

as the PDF value of the multivariate normal distribution $\mathcal{N}(\Lambda, \Sigma)$ at X , and (c.f. (70))

$$\Pi'_{\{a', b'\}}(X) = \Pi(X; [b'\mu_{\alpha} \ a'\mu_{\alpha}]^T, \Sigma_{\Lambda|T_{1,\{a', b'\}}, T_{2,\{a', b'\}}}).$$

⁷which determine $\mathbb{E}(c_z(t))$ and $\mathbb{E}(c_y(t))$ (for fixed turning directions), but not vice versa. There may exist multiple combinations of $\{T_1, T_2\}$ which can result in the same $\{\mathbb{E}(c_z(t)), \mathbb{E}(c_y(t))\}$.

Then for any $a \in [0, k-2]$ and $b \in [0, 2k-2]$, we can always find a matrix (c.f. (73))

$$\Sigma_0 = \begin{pmatrix} \sigma_{\alpha_0}^2 & 0 \\ 0 & \sigma_{\beta_0}^2 \end{pmatrix}$$

satisfying

$$\frac{1}{2\pi\sqrt{|\Sigma_0|}} \leq \min_{a'=0,\dots,k-1,b'=0,1,\dots,2k-1} \left\{ \begin{aligned} &\Pi'_{\{a',b'\}}([b\mu_\alpha \ a\mu_\alpha]^T), \\ &\Pi'_{\{a',b'\}}([(b+1)\mu_\alpha \ a\mu_\alpha]^T), \\ &\Pi'_{\{a',b'\}}([b\mu_\alpha \ (a+1)\mu_\alpha]^T), \\ &\Pi'_{\{a',b'\}}([(b+1)\mu_\alpha \ (a+1)\mu_\alpha]^T) \end{aligned} \right\}. \quad (77)$$

This allows us to define an auxiliary normal distribution with an arbitrary mean and covariance matrix Σ_0 whose maximal PDF value is less than the minimum PDF values of all $\Pr\{c_z(6k), c_y(6k) \mid \mathbb{E}(c_z(6k)) = b'\mu_\alpha, \mathbb{E}(c_y(6k)) = a'\mu_\alpha, A_{6k} = 2\}$ ($a' = 0, \dots, k-1, b' = 0, \dots, 2k-1$) in the square $\{\hat{s}(\hat{c}_z, \hat{c}_y) : \hat{c}_z \in [b\mu_\alpha, (b+1)\mu_\alpha], \hat{c}_y \in [a\mu_\alpha, (a+1)\mu_\alpha]\}$. Therefore, as $n \rightarrow \infty$,

$$\begin{aligned} &\sum_{a'=0}^{k-1} \sum_{b'=0}^{2k-1} \Pr\{c_y(6k) \in [a\mu_\alpha, (a+1)\mu_\alpha], c_z(6k) \in [b\mu_\alpha, (b+1)\mu_\alpha] \mid \mathbb{E}(c_y(6k)) = a'\mu_\alpha, \mathbb{E}(c_z(6k)) = b'\mu_\alpha, A_{6k} = 2\} \\ &\geq \sum_{a'=0}^{k-1} \sum_{b'=0}^{2k-1} \int_{a\mu_\alpha}^{(a+1)\mu_\alpha} \int_{b\mu_\alpha}^{(b+1)\mu_\alpha} \frac{1}{2\pi\sigma_\beta|T_{1,\{a',b'\}}, T_{2,\{a',b'\}}\sigma_\alpha|T_{1,\{a',b'\}}, T_{2,\{a',b'\}}} \\ &\quad \exp\left\{-\frac{(c_y - a'\mu_\alpha)^2}{2\sigma_\beta^2|T_{1,\{a',b'\}}, T_{2,\{a',b'\}}} - \frac{(c_z - b'\mu_\alpha)^2}{2\sigma_\alpha^2|T_{1,\{a',b'\}}, T_{2,\{a',b'\}}}\right\} dc_z dc_y \\ &\geq \sum_{a'=0}^{k-1} \sum_{b'=0}^{2k-1} \int_{a\mu_\alpha}^{(a+1)\mu_\alpha} \int_{b\mu_\alpha}^{(b+1)\mu_\alpha} \frac{1}{2\pi\sigma_{\beta_0}\sigma_{\alpha_0}} \exp\left\{-\frac{(c_y - a'\mu_\alpha)^2}{2\sigma_{\beta_0}^2} - \frac{(c_z - b'\mu_\alpha)^2}{2\sigma_{\alpha_0}^2}\right\} dc_z dc_y \\ &= \sum_{a'=0}^{k-1} \int_{(a-a')\mu_\alpha}^{(a+1-a')\mu_\alpha} \frac{1}{\sqrt{2\pi}\sigma_{\beta_0}} \exp\left\{-\frac{c_y^2}{2\sigma_{\beta_0}^2}\right\} dc_y \sum_{b'=0}^{2k-1} \int_{(b-b')\mu_\alpha}^{(b+1-b')\mu_\alpha} \frac{1}{\sqrt{2\pi}\sigma_{\alpha_0}} \exp\left\{-\frac{c_z^2}{2\sigma_{\alpha_0}^2}\right\} dc_z \\ &\geq \sum_{a'=1}^{k-2} \int_{a'\mu_\alpha}^{(a'+1)\mu_\alpha} \frac{1}{\sqrt{2\pi}\sigma_{\beta_0}} \exp\left\{-\frac{c_y^2}{2\sigma_{\beta_0}^2}\right\} dc_y \sum_{b'=1}^{2k-2} \int_{b'\mu_\alpha}^{(b'+1)\mu_\alpha} \frac{1}{\sqrt{2\pi}\sigma_{\alpha_0}} \exp\left\{-\frac{c_z^2}{2\sigma_{\alpha_0}^2}\right\} dc_z \\ &\rightarrow \sum_{a'=1}^{k-2} \frac{\mu_\alpha}{\sqrt{2\pi}\sigma_{\beta_0}} \exp\left\{-\frac{a'^2\mu_\alpha^2}{2\sigma_{\beta_0}^2}\right\} \sum_{b'=1}^{2k-2} \frac{\mu_\alpha}{\sqrt{2\pi}\sigma_{\alpha_0}} \exp\left\{-\frac{b'^2\mu_\alpha^2}{2\sigma_{\alpha_0}^2}\right\}, \quad (78) \end{aligned}$$

where the first inequality is based on the definition of $\{T_{1,\{a',b'\}}, T_{2,\{a',b'\}}\}$, and the second one comes from (77). Noting that $\mu_\alpha/\sigma_{\alpha_0}$ and $\mu_\alpha/\sigma_{\beta_0}$ scale as $\Theta(\sqrt{r})$, while $k\mu_\alpha/\sigma_{\alpha_0}$ and $k\mu_\alpha/\sigma_{\beta_0}$ go to ∞ as $n \rightarrow \infty$, the last line in (78) converges to

$$\int_0^\infty \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{x^2}{2}\right\} dx \int_0^\infty \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{y^2}{2}\right\} dy = 1/4,$$

which concludes the proof.

C. Distributed Clustering

We assume each node v has an initial seed q_v which is unique within its neighborhood. This can be realized through, e.g., drawing a random number from a large common pool, or simply using nodes' IDs. From time 0, each node v starts a timer with length $t_v = q_v$, which is decremented by 1 at each time instant as long as it is greater than 0. If node v 's timer expires (reaches 0), it becomes a cluster-head, and broadcasts a "cluster_initialize" message to all its neighbors. Each of its neighbors with a timer greater than 0 signals its intention to join the cluster by replying with a "cluster_join" message, and also sets the timer to 0. If a node receives more than one "cluster_initialize" messages at the same time, it randomly chooses one cluster-head and replies with the "cluster_join" message. At the end, clusters are formed such that every node belongs to one and only one cluster. The uniqueness of seeds within the neighborhood ensures that cluster-heads are at least of distance r from each other. We assume that clusters are formed in advance and the overhead is amortized over the multiple computations. The detailed algorithm is given in Algorithm 4.

Algorithm 4 Distributed Clustering

```

 $K \leftarrow 0$   $\{K$ : number of clusters $\}$ 
for all  $v \in V$  do
     $t_v \leftarrow q_v$ 
end for
repeat
    for all  $v$  with  $t_v > 0$  do
         $t_v \leftarrow t_v - 1$ 
        if  $t_v = 0$  then
             $K \leftarrow K + 1$ ,  $C_K \leftarrow \{v\}$   $\{C_k$ : nodes in cluster  $k\}$ 
            for all  $u \in \mathcal{N}_v$  and with  $t_u > 0$  do
                 $t_u \leftarrow 0$ ,  $C_K \leftarrow C_K \cup \{u\}$ 
            end for
        end if
    end for
until  $\bigcup_k C_k = V$ 

```

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] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation*. Englewood Cliffs, NJ: Prentice Hall, 1989.
- [3] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Convergence in multiagent coordination, consensus and flocking," in *IEEE Conf. on Decision and Control*, Seville, Spain, Dec. 2005.
- [4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Gossip algorithms: design, analysis and applications," in *IEEE INFOCOM*, Miami, FL, Mar. 2005.
- [5] —, "Randomized gossip algorithms," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2506–2530, Jun. 2006.
- [6] C. C. Moallemi and B. V. Roy, "Consensus propagation," *IEEE Trans. Inform. Theory*, vol. 52, no. 11, pp. 4753–4766, Nov. 2006.
- [7] R. Karp, C. Schindelhauer, S. Shenker, and B. Vcking, "Randomized rumor spreading," in *IEEE Symp. Foundations of Computer Science (FOCS)*, Redondo Beach, CA, Nov. 2000.
- [8] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *IEEE Symp. on Foundations of Computer Science (FOCS)*, Cambridge, MA, Oct. 2003.
- [9] 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>.
- [10] 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.
- [11] 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.
- [12] L. Lovász and P. Winkler, "Reversal of Markov chains and the forget time," *Combinatorics, Probability and Computing*, vol. 7, pp. 189–204, 1998.
- [13] A. G. Dimakis, A. D. Sarwate, and M. J. Wainwright, "Geographic gossip: efficient aggregation for sensor networks," *IEEE Trans. Signal Processing*, vol. 56, no. 3, pp. 1205–1216, Mar. 2008.
- [14] F. Bénézit, A. G. Dimakis, P. Thiran, and M. Vetterli, "Gossip along the way: Order-optimal consensus through randomized path averaging," in *Allerton Conference*, University of Illinois at Urbana-Champaign, IL, Sep. 2006.
- [15] K. Jung and D. Shah, "Fast gossip via nonreversible random walk," in *IEEE Information Theory Workshop (ITW'06)*, Punta del Este, Uruguay, Mar. 2006.
- [16] K. Jung, D. Shah, and J. Shin, "Minimizing the rate of convergence for iterative algorithms," to appear in *IEEE Transactions on Information Theory*.
- [17] 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.
- [18] B. Nazer, A. G. Dimakis, and M. Gastpar, "Neighborhood gossip: Concurrent averaging through local interference," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP) 2009*, Taipei, Taiwan, Mar. 2009.
- [19] —, "Local interference can accelerate gossip algorithms," in *46th Allerton Conf. on Communication, Control and Computing*, Monticello, IL, Sep. 2008.
- [20] T. Aysal, M. Yildiz, A. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2748–2761, Jul. 2009.

- [21] D. Ustebay, B. Oreshkin, M. Coates, and M. Rabbat, "Rates of convergence for greedy gossip with eavesdropping," in *46th Allerton Conf. on Communication, Control and Computing*, Monticello, IL, Sep. 2008.
- [22] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.
- [23] M. Penrose, *Random geometric graphs*. Oxford, UK: Oxford Univ. Press, 2003.
- [24] D. Aldous, L. Lovász, and P. Winkler, "Mixing times for uniformly ergodic Markov chains," *Stochastic Processes and Their Applications*, vol. 71, no. 2, pp. 165–185, Nov. 1997.
- [25] 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/>.
- [26] W. Li and H. Dai, "Location-aided fast distributed consensus," in *IEEE Statistical Signal Processing Workshop*, Madison, WI, Aug. 2007.
- [27] A. Sinclair, "Improved bounds for mixing rates of Markov chains and multicommodity flow," *Combinatorics, Probability and Computing*, vol. 1, pp. 351–370, 1992.
- [28] B. Bollobas, *Graph Theory: An Introductory Course*. New York: Springer-Verlag, 1979.
- [29] C. Avin and G. Ercal, "On the cover time and mixing time of random geometric graphs," *Theoretical Computer Science*, vol. 380, no. 2-22, pp. 165–185, Jun. 2007.
- [30] T. Leighton and S. Rao, "Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms," *Journal of the ACM*, vol. 46, no. 6, pp. 787–832, Nov. 1999.
- [31] W. Li and H. Dai, "Cluster-based fast distributed consensus," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP) 2007*, Honolulu, Hawaii, Apr. 2007.
- [32] A. W. van der Vaart, *Asymptotic Statistics*. Cambridge, UK: Cambridge University Press, 2000.