# On the Convergence of Distributed Random Grouping for Average Consensus on Sensor Networks with Time-varying Graphs

Jen-Yeu Chen and Jianghai Hu

***Abstract*— Dynamical connection graph changes are inherent in networks such as peer-to-peer networks, wireless ad hoc networks, and wireless sensor networks. Considering the influence of the frequent graph changes is thus essential for precisely assessing the performance of applications and algorithms on such networks. In this paper, we analyze the performance of an epidemic-like algorithm, DRG (Distributed Random Grouping), for average aggregate computation on a wireless sensor network with dynamical graph changes. Particularly, we derive the convergence criteria and the upper bounds on the running time of the DRG algorithm for a set of graphs that are individually disconnected but jointly connected in time. An effective technique for the computation of a key parameter in the derived bounds is also developed. Numerical results and an application extended from our analytical results to control the graph sequences are presented to exemplify our analysis.**

## I. Introduction

Dynamical graph changes are inherent in networks such as peer-to-peer networks, wireless ad hoc networks, and wireless sensor networks. Take the example of the wireless sensor networks, which have attracted tremendous research interests in the recent years. In a practical or even hostile environment, the connection graph of a sensor network may vary frequently in time due to various reasons. For instance, the communication links (edges of the graph) may fail for being interfered, jammed or obstructed; sensor nodes may be disabled or relocate in the field; to save energy, some sensor nodes may sleep or adjust their transmission ranges, thus altering the connection graph. Algorithms and protocols developed on these networks need to take this nature into consideration. In this setting, distributed and localized algorithms requiring no global data structure and centralized coordination are preferable for their scalability and robustness to the frequent graph changes [1], [2], [3], [4], [5], [6].

Although various distributed algorithms have been designed to work well on a time-varying graph, their performances are usually analyzed under the assumption of a fixed connection graph [2], [3], [4]. In this paper, as a particular example, we analyze the performance of a *distributed randomized algorithm*, namely, the DRG (Distributed Random Grouping) algorithm proposed in [2], for average aggregate computation on sensor networks with randomly changing graphs. The analysis techniques introduced in this paper can also be applied to other algorithms whose performances depend on the network connection graph.

This work was partially supported by the National Science Foundation under Grant CNS-0643805. The authors are with the School of Electrical and Computer Engineering, Purdue University, USA. E-mail: {jenyeu,jianghai}@purdue.edu.

Average aggregate computation is an essential operation in sensor networks. One typical category of algorithms to compute the average aggregate is the tree-based algorithms [7], [8]. In these algorithms, sensor nodes' values are gathered along an aggregation tree to the tree's root, the sink node. The other class of algorithms is the epidemic-like algorithms, such as gossip and the DRG algorithm. Unlike tree-based approaches where only the sink node obtains the global average value, epidemic-like algorithms obtain an distributed average consensus among all sensor nodes.

Distributed average consensus has been an important problem with many applications in distributed and parallel computing. Recently it also finds applications in the coordination of distributed dynamic systems and multi-agent systems [9], [10], [11], [12], as well as in distributed data fusion in sensor networks [2], [3], [4], [5]. In analyzing the performance of the proposed distributed schemes for average consensus, authors of [2], [3], [4] bound the running time of their schemes on fixed connected graphs. In [9] Olfati-Saber and Murray characterize the convergence speed of their scheme – over a set of possible directed graphs assumed to be balanced and strongly connected – by the minimal algebraic connectivity of the mirror (undirected) graph of each possible network graph. Authors of [5], [10], [11], [12] provide criteria for their schemes to converge on a dynamical changing graph which is possibly disconnected during some time period but do not characterize the convergence speed. Different from these works, the goal of this paper is to not only determine the convergence criteria but also bound the running time of the DRG algorithm on a randomly changing graph which especially could be *disconnected* all the time.

This paper is organized as follows. In Section II we briefly introduce the DRG algorithm and review its performance on a fixed graph. Then in Section III and Section IV, we analyze the performance of the DRG algorithm on a network graph randomly switching among a set of individually disconnected but jointly connected graphs. Numerical results on four typical sets of such graphs are provided in Section V. An application based on our analytical results to optimize and control the sensors' sleep/awake schedule is presented in section VI. Finally, we conclude our work in Section VII.

## II. Distributed random grouping

We present in [2] a distributed, localized, and randomized algorithm called the *Distributed Random Grouping (DRG)* algorithm to compute the aggregate statistics such as average, sum or maximum among all sensor nodes' measurements. The DRG algorithm is similar to the gossip algorithm [3],

[4] but with a better performance (shown in [2])—unlike the gossip algorithm which forms "pairs" of nodes to exchange information, the DRG algorithm constructs "groups" each of which typically includes more than two nodes to expedite the information mixing process. (Hence, in this paper, we focus on the DRG algorithm rather than the renowned gossip algorithm.) In the following, we will briefly describe the DRG algorithm, which will be the focus of this paper in a generalized setting of time-varying network graphs.

Each sensor node $i$ is associated with an initial observation or measurement value denoted by $v_i(0) \in \mathbb{R}$. The values over all nodes form a vector $\mathbf{v}(0)$. The goal is to compute (aggregate) functions such as the average, sum, max, min, etc of the entries of $\mathbf{v}(0)$ in a distributed manner. Throughout this paper we use $v_i(k)$ to denote the value of node $i$ and $\mathbf{v}(k) = [v_1(k), v_2(k), \ldots]^T$ the *value distribution vector* after running the DRG algorithm for $k$ rounds.

The main idea of the DRG algorithm is as follows. In each round of the iteration, each node independently becomes a group leader with a probability $p_g$ and then invites its one-hop neighbors to join its group by wireless broadcasting an invitation message[1]. A neighbor who successfully[2] receives the invitation message then join its group. Note that unlike the concept of a *cluster* in the sensor network literature, a group contains only the group leader and its *one-hop* neighbors. Several disjoint groups are thus formed over the network. Next, in each group, all members other than the group leader then send the leader their values so that the leader can compute the *local aggregate* (e.g., average in the group) and broadcast it back to the members to update their values. Since in each round, groups are formed at different places of the network, through this randomized process, the values of all nodes will diffuse and mix over the network and converge to the correct aggregate value asymptotically almost surely, provided that the graph is *connected*. DRG iterations stop when certain accuracy criteria are satisfied.

A high-level description of a round (iteration) of the DRG algorithm to compute the average aggregate, is shown in Fig. 1. Aggregates other than the average can be obtained by an easy modification of this algorithm [2]. For simplicity, in the paper we will focus on the average aggregate only.

In [2], a Lyapunov function called the *potential* (function) is defined to assess the convergence of the DRG algorithm.

***Definition** 1:* Consider a graph $G(\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n$ nodes. Given a value distribution $\mathbf{v}(k) = [v_1(k), ..., v_n(k)]^T$ where $v_i(k)$ is the value of node $i$ after $k$ rounds of the DRG algorithm, the potential $\phi_k$ of round $k$ is defined as $\phi_k = ||\mathbf{v}(k) - \overline{v}\mathbf{1}||_2^2 = \mathbf{x}^T(k)\mathbf{x}(k)$, where the constant $\overline{v} = \frac{1}{n}\sum_{i\in\mathcal{V}} v_i(k)$ is the global average value over the network; the vector $\mathbf{1}$ is the vector with all entries one and $\mathbf{x}(k) = [v_1(k) - \overline{v}, ..., v_n(k) - \overline{v}]^T$ is the error vector.

Provided the graph is connected and unchanged in time, it is shown in [2] that the potential $\phi_k$ will strictly decrease by a minimal convergence rate $\gamma := \inf_{\mathbf{v}\neq\overline{v}\mathbf{1}} \left\{ E\left[\frac{\phi_k - \phi_{k+1}}{\phi_k}\right]\right\} > 0$ from its initial value $\phi_0$ to zero, i.e., $\mathbf{v} = \overline{v}\mathbf{1}$, asymptotically almost surely. Further, by $\gamma$ we obtain the upper bounds of running time and the total number of transmissions for the DRG algorithm to reach the average consensus on the fixed connected graph.

**Alg: DRG: Distributed Random Grouping for Average**

**1.1** Each node in the ***idle mode*** independently originates to form a group and becomes the group leader with a probability $p_g$.

**1.2** A node $i$ that decides to become a group leader enters the ***leader mode*** and broadcasts a group call message, $GCM \equiv (group_{id} = i)$, to all its neighbors and waits for $JACK$ message from its neighbors.

**2.1** A neighboring node $j$, in the ***idle mode*** and successfully receiving a $GCM$, responds to the group leader by a joining acknowledgment, $JACK \equiv (group_{id} = i, v_j, join(j) = 1)$, with its value $v_j$ included. It then enters the ***member mode*** and waits for the group assignment message $GAM$ from its leader.

**3.1** The group leader, node $i$, gathers the received $JACKs$ from its neighbors; count the total number of group members, $J = \sum_{j\in g_i} join(j) + 1$; and compute the average value of the group, $Ave(i) = (\sum_{k\in g_i} v_k)/J$.

**3.2** The group leader, node $i$, broadcasts the group assignment message $GAM \equiv (group_{id} = i, Ave(i))$ to its group members and then returns to the ***idle mode***.

**3.3** A neighboring node $j$, in the ***member mode*** and upon receiving receiving $GAM$ from its leader node $i$, updates its value $v_j = Ave(i)$ and then returns to the ***idle mode***.

Fig. 1. A round of DRG algorithm to compute average aggregate

## III. TIME-VARYING NETWORK GRAPH

It is nontrivial to extend our results of the DRG algorithm on a fixed connected graph in [2] to the general case of time-varying graphs. For a fixed graph, we have shown in [2] that all the node values will eventually reach consensus by converging to the global average, starting from an arbitrary initial value if and only if the graph is connected. However, in the case when the graph is time-varying, even if the graph is disconnected in some time periods, it is still possible that consensus can be reached, provided that the union of the graphs appearing infinitely often is connected. (This convergence criterion has been shown in [5], [10] for their schemes. Later, we will briefly prove that it is also valid for the DRG algorithm. For the detail proof, please refer to [13])

Moreover, characterizing the convergence rate of the DRG algorithm in this case is a challenging task, as it depends on the possible graphs of the network, as well as the rules for the (random) evolution of the network graph in time. If some (at least one) of the possible graphs are connected and visited infinitely often, then our results in [2] can be directly extended. We can lower bound the convergence rate for the whole execution by the minimum among those non-zero convergence rates on connected and infinitely-often-visited graphs, though this bound is conservative. If these graphs are visited according to some particular frequency, e.g., the stationary distribution of an ergodic discrete-state

[1] A wireless broadcast transmission by the group leader can be received by all its one-hop neighbors.

[2] Collisions amid multiple invitation messages from different group leaders may occur at some nodes which hence will not join any group in that round. Also, a group leader will ignore invitations from its neighbors.

(graphs) stochastic process, then we can obtain a normalized convergence rate $\gamma* = 1 - \prod(1-\gamma_G)^{p_G}$, where $\gamma_G$ is the convergence rate on a candidate graph $G$ and $p_G$ is the frequency that the graph $G$ is visited.

If all graphs are disconnected, the results in [2] can not be directly applied to find the convergence rate since the convergence rate on each graph is zero ($\gamma = 0$) for all graphs. However, if the union of all graphs visited infinitely often is connected, the DRG algorithm can still converge. In the following sections we will show the convergence rate of the DRG algorithm running on a network graph randomly switching among *a set of individually disconnected but jointly connected graphs*.

## IV. INDIVIDUALLY DISCONNECTED BUT JOINTLY CONNECTED GRAPHS

### A. Convergence criteria

In this section, we consider a model where all the possible graphs are individually disconnected but jointly connected. We assume that each graph occurs with some positive probability in a round of the DRG algorithm and is visited infinitely often in the whole stochastic graph sequence. In such a model, the expected potential decrement in a single round in the worst case is uniformly zero. However, even though all possible graphs are disconnected, if their union graph is connected, the DRG algorithm can still converge to global average. We can show this convergence criterion, by extending the proof of Theorem 1 in [5]. Suppose there are $r < \infty$ possible disconnected graphs $\{G_i\}$ each of which is visited infinitely often. On each graph $G_i$ there are a set of possible group distributions $\{\mathcal{D}_{G_i}\}$ followed from the randomized grouping rule of the DRG algorithm. Each $\mathcal{D}_{G_i}$ is associated with a double stochastic, symmetric and paracontracting [3] matrix $W^{\mathcal{D}_{G_i}}$ (w.r.t. Euclidean norm) so that the value vector is updated by $\mathbf{v}(k+1) = W^{\mathcal{D}_{G_i}}\mathbf{v}(k)$ when $\mathcal{D}_{G_i}$ occurs at round $k$. (The value updating matrix $W_i$ of [5] depends only on the chosen network graph $G_i$ but our $W^{\mathcal{D}_{G_i}}$ is determined by the group distribution $\mathcal{D}_{G_i}$ which in turn depends on the graph $G_i$ and the *randomized* grouping strategy of the DRG algorithm.) Each $G_i$ is visited i.o., so is $\mathcal{D}_{G_i}$. Hence, there exists at least a set of updating matrices $\Gamma = \{W^{\mathcal{D}_{G_i}}\}$ for $i = 1 \cdots r$ such that $\mathcal{M} = \frac{1}{r}\sum_{i=1}^{r} W^{\mathcal{D}_{G_i}}$ is stochastic, symmetric and irreducible if the union of possible graphs is connected. This implies that $\mathcal{M}$'s fix-point subspace, i.e., the eigenspace associated with the eigenvalue 1, $\mathcal{H}(\mathcal{M}) = \mathbf{span}(\mathbf{1})$. Therefore, by [5], $\bigcap_\Gamma \mathcal{H}(W^{\mathcal{D}_{G_i}}) = \mathbf{span}(\mathbf{1})$, which by [14] leads to the conclusion that the DRG algorithm will asymptotically converge to the *unique fixed point* $\left(\frac{1}{n}\mathbf{1}^T\mathbf{v}\right)\mathbf{1}$, i.e., the status of the average consensus.

### B. Convergence rate and the upper bound of running time

To characterize the convergence rate, for illustration purpose, we analyze the simplest case where the network graph switches randomly (infinitely often) between two graphs that are individually disconnected but jointly connected. Our analysis can be easily extended to the general case of switching amid more than two graphs. As an example, see Fig. 4(a). In each round, the network graph can be either $G_1$ or $G_2$. The graph changing pattern can be characterized by a two-state Markov chain shown in Fig. 3(a) by setting each graph as a state of the Markov chain. (The dynamics of the execution of the DRG algorithm on a time-varying graph can be modeled by a stochastic hybrid system [13].)

Since $G_1$ and $G_2$ are each disconnected, the lower bound on the convergence rate for each of them in a single round is zero. However, in two rounds, the network may switch between these two jointly connected graphs with positive probability, resulting in a positive expected potential decrement. Thus to lower bound the expected potential decrement rate, we need to consider two rounds of DRG iterations. Since this is a worst-case analysis, we assume the worst scenario: only one group is formed in each round. The DRG algorithm can have more groups in a round and hence converge faster than the upper bound derived here. Also, we assume every node has equal probability to become a leader. Without loss of generality, define $\mathbf{x}(k) = \mathbf{v}(k) - \overline{v}\mathbf{1}$, which is orthogonal to the vector $\mathbf{1}$, i.e., $\mathbf{x}(k) \perp \mathbf{1}$. Then each round of DRG iteration can be expressed as $\mathbf{x}(k+1) = W(k)\mathbf{x}(k)$ for some random matrix $W(k)$ depending on the choice of the group leader. For example, if in round $k$, the network graph is $g_k \in \{G_1, G_2\}$ and node $i$ becomes the group leader, then $W(k) = W^{g_k,i} = [w_{\eta\varsigma}^{g_k,i}]$ where

$$w_{\eta\varsigma}^{g_k,i} = \begin{cases} \frac{1}{d_i+1}, & \text{if } \eta,\varsigma \in \{N_{g_k}(i) \cup i\}; \\ 1, & \text{if } \eta,\varsigma \notin \{N_{g_k}(i) \cup i\} \text{ and } \eta = \varsigma; \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Here $N_{g_k}(i)$ is the set of neighbors of node $i$ in graph $g_k$.

From the continuous dynamics, in two rounds, we have $\mathbf{x}(k+2) = W(k+1)W(k)\mathbf{x}(k) = \widetilde{W}\mathbf{x}(k)$. The ratio of potential decrement after two rounds is

$$\frac{\phi_k - \phi_{k+2}}{\phi_k} = \frac{\|\mathbf{x}(k)\|^2 - \|\mathbf{x}(k+2)\|^2}{\|\mathbf{x}(k)\|^2} = 1 - \frac{\mathbf{x}(k)^T\widetilde{W}^T\widetilde{W}\mathbf{x}(k)}{\mathbf{x}(k)^T\mathbf{x}(k)}. \quad (2)$$

Define the two-round convergence rate $\gamma_2$ as the lower bound on the expected convergence rate after two consecutive rounds:

$$\gamma_2 := \inf_{\substack{\mathbf{x}(k)\perp\mathbf{1};\\ \mathbf{x}(k)\neq\mathbf{0}}} \left\{ E\left[\frac{\phi_k - \phi_{k+2}}{\phi_k}\right]\right\}.$$

From (2), we have

$$E\left[\frac{\phi_k - \phi_{k+2}}{\phi_k}\right] = 1 - \frac{\mathbf{x}(k)^T E\left[\widetilde{W}^T\widetilde{W}\right]\mathbf{x}(k)}{\mathbf{x}(k)^T\mathbf{x}(k)}$$
$$= 1 - \frac{\mathbf{x}(k)^T\,\mathbf{K}\,\mathbf{x}(k)}{\mathbf{x}(k)^T\mathbf{x}(k)} \geq 1 - \lambda_2(\mathbf{K}) > 0, \quad (3)$$

where $\lambda_2(\mathbf{K})$ is the *second largest* eigenvalue of the *compound matrix* $\mathbf{K} = E\left[\widetilde{W}^T\widetilde{W}\right] =$

[3] A matrix $W$ is paracontracting w.r.t. a vector norm $\|\cdot\|$ if $Wx \neq x \Leftrightarrow \|Wx\| < \|x\|$. [5]

$\sum_{(g_k,g_{k+1})}\sum_{i,j}\frac{P_{g_k,g_{k+1}}}{n^2}(W^{g_{k+1},j}W^{g_k,i})^T(W^{g_{k+1},j}W^{g_k,i})$. In the above, $P_{g_k,g_{k+1}}=P(g_k)P(g_{k+1}|g_k)$ is the probability that the graph of round $k$ is $g_k$ and the graph of round $k+1$ is $g_{k+1}$. So, the lower bound on the expected convergence rate after two consecutive rounds is $\gamma_2=1-\lambda_2(\mathbf{K})>0$. The largest eigenvalue of $\mathbf{K}$ is always one so the *second largest* eigenvalue $\lambda_2(\mathbf{K})<1$ since the two possible graphs are jointly connected.

***Theorem** 2:* On a set of individually disconnected but jointly connected graphs which together have a compound matrix $\mathbf{K}$, with an arbitrary initial value distribution $\mathbf{v}(0)$ and the initial potential $\phi_0$, with high probability (at least $1-(\frac{\varepsilon^2}{\phi_0})^{\sigma-1}$ ; $\sigma>2$), the average consensus problem can be reached within an $\varepsilon>0$ accuracy, i.e., $|v_i-\overline{v}|\leq\varepsilon$ for all $i$, by running the DRG algorithm in

$$O\left(\sigma\log_{\lambda_2(\mathbf{K})}(\frac{\varepsilon^2}{\phi_0})\right)\text{ rounds.}$$

*Proof:* To meet the accuracy criterion after $2\tau$ rounds of the DRG algorithm, by (3), we need $E[\phi_{2\tau}]\leq(1-\gamma_2)^\tau\phi_0=(\lambda_2(\mathbf{K}))^\tau\phi_0\leq\varepsilon^2$, from which we get $\tau\geq\log_{\lambda_2(\mathbf{K})}(\frac{\varepsilon^2}{\phi_0})$. Choose $\tau=\sigma\log_{\lambda_2(\mathbf{K})}(\frac{\varepsilon^2}{\phi_0})$ where the $\sigma\geq2$. Then because $(\frac{\varepsilon^2}{\phi_0})\ll1$ and $(\sigma-1)\geq1$, the Markov inequality

$$P(\phi_{2\tau}>\varepsilon^2)<\lambda_2(\mathbf{K})^{\sigma\log_{\lambda_2(\mathbf{K})}(\frac{\varepsilon^2}{\phi_0})}(\frac{\phi_0}{\varepsilon^2})=(\frac{\varepsilon^2}{\phi_0})^{\sigma-1}.\quad(4)$$

Thus, $P(\phi_{2\tau}\leq\varepsilon^2)\geq1-(\frac{\varepsilon^2}{\phi_0})^{(\sigma-1)}$ is arbitrarily close to 1 by choosing large $\sigma$. (Since typically $\phi_0\gg\varepsilon^2$, taking $\sigma=2$ is sufficient to have high probability at least $1-O(\frac{1}{n})$; in case $\phi_0>\varepsilon^2$, then a larger $\sigma$ is needed to have a high probability). Hence running the DRG algorithm for $2\tau=O\left(\sigma\log_{\lambda_2(\mathbf{K})}(\frac{\varepsilon^2}{\phi_0})\right)$ rounds, , with high probability $1-(\frac{\varepsilon^2}{\phi_0})^{(\sigma-1)}$, the DRG algorithm converges to an $\varepsilon$ accuracy. ■

Similar procedures can be carried out to obtain the convergence rate for network graphs randomly switching among a set of individually disconnected but jointly connected graphs consisting of more than two graphs. In the following section, we provide an effective way to compute the compound matrix, $\mathbf{K}$, for a family of individually disconnected but jointly connected graphs.

### *C. Computation of the matrix* $\mathbf{K}$

As we see from the above, the compound matrix $\mathbf{K}$ is a key element in the upper bound of the running time of the DRG algorithm. Here we briefly introduce an effective way to compute the compound matrix $\mathbf{K}$. For the detail of this method we refer readers to [13]. As an example, we consider a set of $h=n-1$ graphs each with $n$ nodes positioned as a linear array and with only one edge connecting two consecutive nodes. Shown in Fig. 4(c), $G_1$, $G_2$, and $G_3$ are possible graphs with $n=4$. Because of the extreme sparsity of each graph, a time-varying network graph randomly switching among these graphs is among the worst cases for the DRG algorithm to converge. However, since these graphs are jointly connected, the DRG algorithm will converge to the global average. Denote the stochastic

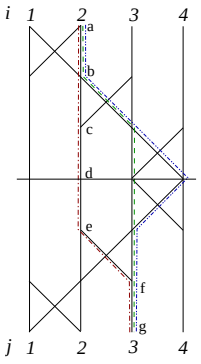



Fig. 2. The un-solid lines are the possible paths (multiplication combinations) for $\mathbf{K}_\Lambda(2,3)$.

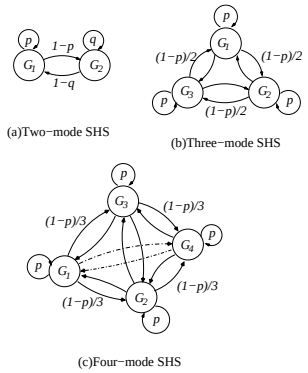



Fig. 3. The Markov chain for jointly connected switching graphs each of which is disconnected.

graph sequence over $h$ rounds as $\Lambda=\{g_k\}_{k=1}^h$. We rewrite $\mathbf{K}=\sum_\Lambda P(\Lambda)E\left[\widetilde{W}^T\widetilde{W}\middle|\Lambda\right]=\sum_\Lambda P(\Lambda)\mathbf{K}_\Lambda$. The graph sequence $\Lambda$ and $P(\Lambda)$ depend on the graph changing pattern. It turns out that we need to effectively compute $\mathbf{K}_\Lambda=E\left[\widetilde{W}^T\widetilde{W}\middle|\Lambda\right]$. To achieve this, we cascade computation blocks each of which represents a $W(k)$ of $\mathbf{K}_\Lambda$ in the way in Fig. 2 for the example sequence $\Lambda=\{g_1=G_1,g_2=G_2,g_3=G_3\}$ in Fig. 4(c). For an arbitrary $n$, from these cascading computation blocks, we obtain the analytical expression of $K_\Lambda$ as follows. Let $r=\frac{1}{2n}$, $\zeta=(1-3r)$ and $\Im=(2r)^{x-1}(r+r\zeta\frac{1-r^{n-x-i}}{1-r}+r^{n+1-x-i})$; $\mathbf{K}_\Lambda(i,j)=$

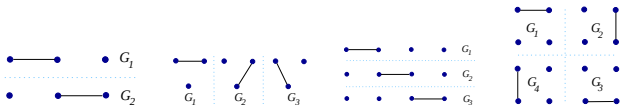
$$\begin{cases}\zeta\frac{1-r^h}{1-r}+r^h, & i=j=1;\\ r+\zeta^2\frac{1-r^{n-i}}{1-r}+\zeta r^{n-i}, & 1<i=j\leq n;\\ \Im, & i=1,\,j=1+x,1\leq x\leq n-1;\\ (1-2r)\Im, & 1<i\leq n,\,j=i+x,1\leq x\leq n-i;\\ \mathbf{K}_\Lambda(j,i), & i>j.\end{cases}$$

Note that $\mathbf{K}_\Lambda$ is a double stochastic matrix which can be easily verified from the above expression.

## V. Numerical results

We now present some numerical results on the convergence rate of the DRG algorithm on the randomly graph-changing models studied in Section IV. Recall that, in this case, there are a total of $h$ possible graphs for the network, each of which is disconnected. As a family, however, the $h$ graphs are jointly connected. A lower bound $\gamma_h$ on the $h$-step convergence rate is given by $\gamma_h=1-\lambda_2(\mathbf{K})$, where $\mathbf{K}=E\left[\widetilde{W}^T\widetilde{W}\right]$ and $\widetilde{W}=W(k+h-1)\cdots W(k+1)W(k)$.

In Fig. 4, four cases under study are plotted. In case I and case III, the union of possible graphs forms a linear array with three and four nodes, respectively. In case II and case IV, the union of possible graphs forms a ring with three and four nodes, respectively. The Markov chains (the randomly graph-changing model) describing the transitions among possible graphs are shown in Fig. 3: Fig. 3(a) for case I; Fig. 3(b) for case II and case III; and Fig. 3(c) for case IV. We compute $\gamma_2$ for case I, $\gamma_3$ for case II and III, and $\gamma_4$ for case IV, under different transition probabilities $p$ and $q$.



(a) Case I (b) Case II (c) Case III (d) Case IV

Fig. 4. Example cases.

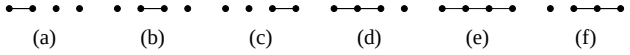
Fig. 5(a) plots the computed $\lambda_2(\mathbf{K})$ of case I as a function of the transition probabilities $p$ and $q$. It can be seen that, as $p$ and $q$ both approach 0, $\lambda_2(\mathbf{K})$ achieves its minimum; hence $\gamma_2 = 1 - \lambda_2(\mathbf{K})$ achieves its maximum, implying the fastest convergence rate of the DRG algorithm. This is understandable as, in this case, the transitions between the two possible graphs are the most frequent and occur in each round, remedying the slow convergence caused by the individual disconnected graph. On the other hand, by requiring that $p + q = 1$, $\lambda_2(\mathbf{K})$ becomes a function of $p$ only, and is plotted in Fig. 5(b). Note that the plot in Fig. 5(b) is a slice of the plot in Fig. 5(a) along the line $p + q = 1$. As can be seen from the plot, the minimum $\lambda_2(\mathbf{K})$, hence the maximal convergence rate $\gamma_2$, occurs at $p = q = 0.5$ when the two graphs have identical stationary probability 0.5. For all other choices of $p$ and $q$ satisfying $p+q = 1$, the transitions have a tendency of staying in one graph longer, which slows down the convergence of the DRG algorithm.

Fig. 5(c) compares the convergence rates of the DRG algorithm for these four cases as well as an additional case of the ring topology that is of five disconnected graphs, $h = 5$. The computed convergence rate $\gamma_h$ for these five cases are plotted in Fig. 5(c) as functions of the transition probability $p$ (in case I, we set $q = p$). We observe that, the larger the number of nodes, the slower the convergence rate. In addition, with the same number of nodes, the case whose union graph is a linear array has the slower convergence rate than the corresponding case whose union graph is a ring. This is because on a linear array each of the two end nodes has only one direction to spread out its value whereas all nodes in a ring have two.

## VI. An application: sleep/awake scheduling

One of the efforts to save energy consumption in sensor network is to let some sensor nodes sleep (in power saving mode) from time to time without affecting the correctness of the execution of the algorithm but possibly with some acceptable degradation on the performance of the algorithm. In our example, the network graph may become disconnected when some nodes sleep. This is especially true for sparse network graphs. However, from the results of previous sections, we know that the DRG algorithm still converges as long as the time-varying network graph is jointly connected. In this section we discuss the DRG's performance on several sleep/awake scheduling sequences and try to find the best controlled graph sequence in terms of both energy saving and convergence time.

We consider a sparse graph: a linear array with 4 sensor nodes in a row. The network graph "e" of Fig. 6(e) is the



(a) (b) (c) (d) (e) (f)

Fig. 6. Connection graphs for sleep/awake scheduling

connected graph while all four nodes are awake. Other graphs in Fig. 6 are disconnected because some sensor nodes are in sleep mode, e.g., in graph "a" (Fig. 6(a)) node 3 and node 4 are in sleep mode. When a node sleeps, its CPU is at power saving mode and its radio components are deactivated. We compare nine different periodic graph sequences (i.e., different sleeping schedules for sensor nodes): $\Lambda_1 = \{e\}$, $\Lambda_2 = \{abc\}$, $\Lambda_3 = \{abcb\}$, $\Lambda_4 = \{abce\}$, $\Lambda_5 = \{dc\}$, $\Lambda_6 = \{df\}$, $\Lambda_7 = \{adefc\}$, $\Lambda_8 = \{edef\}$, $\Lambda_9 = \{eaebec\}$, where $\{abc\}$ means that the network graph repeats in "abc" pattern periodically, i.e., in first round the network graph is Fig. 6(a), the second round Fig. 6(b), the third round Fig. 6(c) and the fourth round Fig. 6(a) again ... etc. Since the graphs in each sequence are joint connected, the DRG algorithm will converge for these graph sequences.

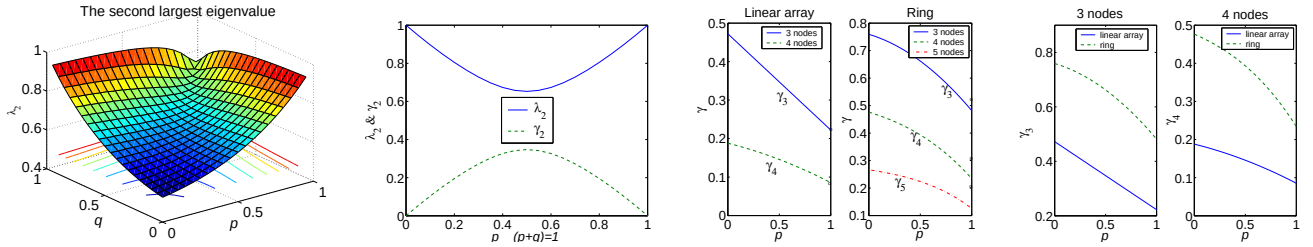
Running the DRG algorithm on an $n$-nodes linear array with $m_k$ awake nodes, we model the expected energy consumption in the round $k$ of the DRG algorithm as follows.

$$\begin{aligned} E_{DRG} &= \frac{2}{n}(3E_{tx} + 2(E_{r/w} + E_{CPU-active})) \\ &\quad + \frac{m_k - 2}{n}(4E_{tx} + 3(E_{r/w} + E_{CPU-active})) \\ &\quad + \frac{m_k}{n}(E_{CPU-idle} + E_{rx}) \\ &= \frac{m_k}{n}(3E_0 + E_{tx} + E_1) - \frac{2}{n}E_0, \end{aligned}$$

where $E_0 = E_{tx} + E_{r/w} + E_{CPU-active}$ and $E_1 = E_{CPU-idle} + E_{rx}$; $E_{tx}$ is the energy for transmitting a message and $E_{rx}$ is for nodes to listen to MAC channels and receive messages. In a round of the DRG algorithm, the CPU of an awake node consumes $E_{CPU-active}$ or $E_{CPU-idle}$ when it is busy or idle, respectively. $E_{r/w}$ is the total energy required for an awake node to read/ write information from/to its EEPROM in a round of the DRG algorithm. Except the number of awake nodes $m_k$, all the other parameters in the above equation are constants in rounds of the DRG algorithm. (For detail energy quantities consumed by a sensor node, we refer to [15].) The total energy consumption $E_{DRG}$ in a round of the DRG algorithm is a linear function of the number of awake nodes $m_k$ which may vary by rounds. To compare the average energy consumed in a round for different graph sequences, we take the average number of awake nodes $E^*_{DRG}(\Lambda) = \sum_{k\in\Lambda} m_k/|\Lambda|$, where $|\Lambda|$ is the number of graphs of a sequence pattern, as the normalized energy index for the sequence $\Lambda$, e.g., for $\{dc\}$ the normalized energy index is $E^*_{DRG}(\{dc\}) = (3 + 2)/2 = 2.5$.

From Theorem 2, given $\phi_0$ and $\epsilon$, the running time is proportional to $-|\Lambda| \log^{-1}(\lambda_2(\mathbf{K}_\Lambda))$. Thus, we define the normalized index for running time of the DRG algorithm $Time^*(\Lambda) = -\log^{-1}(\lambda^*_2(\mathbf{K}_\Lambda)) = -|\Lambda| \log^{-1}(\lambda_2(\mathbf{K}_\Lambda))$.

Fig. 7 shows our simulation results for the nine graph sequences mentioned previously. The x-axis is the normalized




(a) $\lambda_2(\mathbf{K})$ for the two-mode model. (b) $\lambda_2(\mathbf{K})$ and $\gamma_2$ for the two-mode model when $p+q=1$. (c) Lower bound $\gamma$ for different types of graphs. (d) $\gamma$ for various graphs with the same number of nodes.

Fig. 5. Numerical results

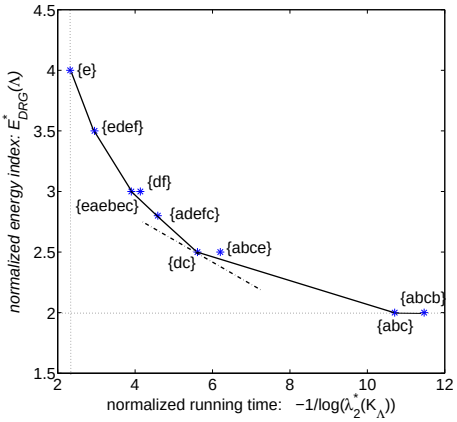



Fig. 7. The normalized energy per round and normalized convergence time for different graph sequences.

index for running time; the y-axis is the normalized energy index $E^*_{DRG}(\Lambda)$. The sequence $\Lambda_1 = \{e\}$ where nodes never sleep consumes the most energy but converges fastest. In contrast, the sequence $\Lambda_3 = \{abcb\}$ where two nodes sleep in turn round by round consumes least energy but converges the slowest. There will be always a tradeoff between these two performance indices. To incorporate both energy consumption and convergence time into the performance assessment, we can minimize the combined indices:

$$\mathbf{min}(\alpha \cdot E^*_{DRG}(\Lambda) + \beta \cdot Time^*(\Lambda)),$$

where $\alpha + \beta = 1$. Two extreme cases are $(\alpha = 1, \beta = 0)$ and $(\alpha = 0, \beta = 1)$, representing the consideration of only energy consumption and only convergence time correspondingly. By linear programming on the convex hull of Fig 7, we can find the proper graph sequence for a desired pair of $\alpha$ and $\beta$. For example, the sequence $\Lambda_5 = \{dc\}$ should be used when we set $0.0982 < \frac{\beta}{\alpha} < 0.2926$.

## VII. CONCLUSION

To guarantee the correctness and precisely bound the running time of an algorithm developed on a sensor network, we need to consider one of the senor network's salient nature: frequently changing graphs. In this paper, we model the execution of the Distributed Random Grouping (DRG) algorithm for computing the average aggregate on a sensor network with randomly changing graphs. Criteria are given for the convergence of the DRG algorithm on a randomly changing graph. Particularly for a family of graphs which are individually disconnected but jointedly connected, bounds on the convergence rate and the running time are presented. Numerical results are provided to illustrate our analytical results. An application built on our analytical results to optimize and control the sleep/awake schedule for sensor nodes is also introduced. The notions and techniques presented in this paper can be applied to other schemes for distributed average consensus in sensor networks with time-varying graphs.

## REFERENCES


[1] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar, "Next century challenges: scalable coordination in sensor networks," in *Mobicom*, 1999, pp. 263–270.

[2] J.-Y. Chen, G. Pandurangan, and D. Xu, "Robust aggregate computation in wireless sensor network: distributed randomized algorithms and analysis," in *IEEE Trans. on parallel and distributed system,*, Sep. 2006, pp. 987–1000.

[3] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *Proc. FOCS*, 2003.

[4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Gossip algorithms: Design, analysis and applications," in *INFOCOM'05*.

[5] L. Xiao, S. Boyd., and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. IPSN*, 2005.

[6] B. Krishnamachari, *Networking Wireless Sensors*. Cambridge University Press, 2005.

[7] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," in *Proc. ICDCS*, 2002.

[8] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," in *Proc. OSDI*, 2002.

[9] R. Olfati-Saber and R. M. Murray, "Consensus problem in networks of agents with switching topology and time delays," *IEEE Trans. on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, sep 2004.

[10] L. Fang and P. Antsaklis, "Information consensus of asynchronous discrete-time multi-agent systems," in *Proc. ACC'05*.

[11] W. Ren and R. W. Beard, "Consensus seeking in multi-agent systems under dynamically changing interaction topologies," *IEEE Trans. on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.

[12] L. Moreau, "Stability of multi-agent systems with time-dependent communication links," *IEEE Trans. on Automatic Control*, vol. 50, no. 2, pp. 169–182, 2005.

[13] J.-Y. Chen and J. Hu, "Performance analysis of distributed randomized algorithms on randomly changing graphs by stochastic hybrid systems," in *Tech. Report, Purdue Univ.*, 2006, submitted for publication.

[14] L. Elsner, I. Koltracht, and M. Neumann, "On the convergence of asynchronous paracontractions with applications to tomographic reconstruction from incomplete data," *Linear Algebra Appl.*, vol. 130, pp. 65–82, 1990.

[15] V. Shnayder, M. Hempstead, B. Chen, G. Allen, and M. Welsh, "Simulating the power consumption of large-scale sensor network applications," in *Proc. SenSys*, 2004.