

CLUSTER-BASED FAST DISTRIBUTED CONSENSUS

Wenjun Li and Huaiyu Dai

Department of Electrical and Computer Engineering
North Carolina State University, Raleigh, NC, 27695
{wli5, huaiyu.dai}@ncsu.edu

ABSTRACT

In this paper, we propose cluster-based distributed averaging algorithms in forms of fixed iteration and random gossiping. Nodes within a cluster maintain the same value via broadcasting by the cluster-head, and information exchange occurs between neighboring clusters. Clustering essentially allows nodes in neighboring clusters to be joined, hence the resultant graph is well-connected and the algorithm converges much faster. Moreover, since the number of clusters is much smaller than the number of nodes, the communication and computation burden of the consensus algorithm is significantly reduced.

Index Terms— Distributed Computing, Distributed consensus, Clustering, Sensor networks.

1. INTRODUCTION

The average of node values is desired in many applications, including distributed detection and estimation, as well as network coordination and optimization. The distributed averaging problem where nodes try to reach consensus on the average value through iterative local information exchange has been vigorously investigated recently, see [1, 2] and references contained therein. In contrast to schemes which rely on a spanning tree rooted at a fusion center, such distributed algorithms scale well as the network grows, and exhibit robustness to node and link failures. The optimal fixed iteration for fastest convergence is studied in [1]. The gossip algorithms [3] are designed under the practical constraint that a node can communicate with only one neighbor at any time instant. In particular, Boyd *et al.* [2] proposed a randomized gossip algorithm, which is more robust to link failures and can be implemented in an asynchronous manner. A general concern on this set of fully-distributed algorithms, however, is incurred delay upon convergence of iterations.

In this paper, we take a hybrid approach by incorporating clustering techniques into distributed algorithms to achieve faster convergence and reduce communication and computational complexity. Our work is motivated by two main ob-

servations. First, in a network where nodes are not well connected, convergence of both fixed iterative and gossiping algorithms is slow. Second, the broadcast nature of the wireless medium can be exploited to reduce the amount of communication. We first propose a distributed clustering algorithm. With one transmission by the cluster-head, nodes in a cluster can maintain the same value at any time instant, hence a cluster can be viewed as a single entity in the whole network. A pair of clusters are joined by *gateway* nodes residing in two different clusters. We propose fixed iteration and gossip algorithms among clusters through information exchange between gateway nodes. Essentially, clustering allows nodes in neighboring clusters to become neighbors, hence the connectivity of the resultant graph is greatly improved, leading to much faster convergence of the consensus algorithms. Clustering has been successfully employed to reduce the amount of communication for computing decomposable functions in networks with a fusion center [4]. Since the number of clusters is much smaller than the number of nodes, our algorithms also significantly reduce the communication and computational burden. The main overhead of our approach lies in clustering and the initial averaging within clusters. The paper is organized as follows. Section 2 describes the desired cluster properties together with a distributed clustering algorithm, and lays the foundation for analysis. In Section 3, we propose cluster-based distributed averaging algorithms with fixed iteration and random gossiping, whose performance is analyzed and verified with simulation results. Section 4 concludes the paper.

2. CLUSTERING

2.1. Clustering Algorithm

Consider a wireless network represented by a connected graph $G = (V, E)$, where the vertex set V contains n nodes and E is the edge set. We assume that the n nodes form K clusters in such a manner that the following assumptions are satisfied:

1. Each node belongs to one and only one cluster.
2. In each cluster, there is a node which is adjacent to all the remaining nodes in the cluster. Such a node is called

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

a cluster-head. (If more than one such nodes exist, only one is chosen).

Two clusters are called adjacent (or neighbors) if there is a direct link joining them. Assume that through some 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 a single such link being activated. The end nodes of active links are called gateway nodes.

The set of cluster-heads by our assumption is a dominating set, i.e., a subset of nodes which are at most 1 hop away from any node. The well-known minimum dominating set (MDS) problem seeks a dominating set of minimum size, and has been proven to be NP-hard [5]. For related works, see [6] and references therein. In the following, we describe a non-iterative decentralized clustering algorithm for choosing a dominating set.

Assume that each node i knows its one-hop neighbors, hence its degree d_i (i.e., the number of neighbors). Initially, each node sets its own flag to 0, meaning that it does not yet belong to any cluster. At a certain time, each node i starts a timer with length t_i drawn from an exponential distribution with rate d_i . If node i 's timer expires at t_i , it becomes a cluster-head. It sets its flag to 1, and broadcasts a "cluster_initialize" message to all its neighbors. Each of its neighbors with flag 0 signals its intention to join the cluster by replying with a "cluster_join" message. It also sets its own flag to 1 and stops the timer. At the end, clusters satisfying the two properties mentioned above are formed. The particular choice of timers ensures that high degree nodes have more chances to become cluster-heads, somewhat like a greedy algorithm. In this paper we assume that clusters are formed in advance and the overhead is amortized over the multiple computations.

In this work, we focus on wireless networks modeled by a geometric random graph $G(n, r)$ [7], where n nodes are uniformly and independently distributed on a unit square, and r is the common transmission range of all the n nodes, i.e., any two nodes are adjacent if they are within distance r of each other. The choice of $r(n) = \Theta\left(\sqrt{\frac{\log n}{n}}\right)$ is commonly used to ensure connectivity and good energy efficiency [7]. Since any two nodes in a cluster are at most 2 hops apart, the number of clusters $K = \Theta\left(\frac{n}{\log n}\right)$.

2.2. Graph Generated by Clustering

Denote the set of neighbors of i in $G = (V, E)$ by $N(i)$. The Laplacian matrix for a graph $G = (V, E)$ is defined as

$$L(i, j) = \begin{cases} -1, & j \in N(i) \\ d_i, & i = j \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Let $\lambda_1(\mathbf{L}) \geq \lambda_2(\mathbf{L}) \geq \dots \geq \lambda_n(\mathbf{L}) = 0$ be the eigenvalues of \mathbf{L} in nonincreasing order. The Gerschgorin cir-

cle theorem [8] guarantees that $0 < \lambda(\mathbf{L}) < 2d_{\max}$, with $d_{\max} = \max_{i \in V} d_i$.

Let n_k denote the size of cluster k (i.e., the number of nodes in the cluster), and $c(i)$ denote the index of the cluster that node i belongs to. Consider the new graph $\hat{G} = (V, \hat{E})$, where the vertex set is the same as in G , but the edge set \hat{E} is chosen as follows: $(i, j) \in \hat{E}$ if $c(i) = c(j)$ or $c(i)$ and $c(j)$ are adjacent. It is easy to see that if $e \in E$ then $e \in \hat{E}$. Moreover, since nodes in the same cluster and adjacent clusters all become neighbors, we should expect that the new graph \hat{G} is much better-connected than the original graph G . Denote the Laplacian matrix of \hat{G} by $\hat{\mathbf{L}}$, and the degree of vertex i in \hat{G} by \hat{d}_i . Note that nodes in the same cluster have the same degree in \hat{G} .

3. CLUSTER-BASED CONSENSUS ALGORITHM

Let vector $\mathbf{x}(0) = [x_1(0), \dots, x_n(0)]^T$ contain the initial values observed by node i , and $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ denote the average value of $\mathbf{x}(0)$. The initialization of cluster-based averaging algorithms is as follows. Each node is informed of the size of the cluster it belongs to from a broadcast message by the cluster-head. Each node (except the cluster-head) transmits its value to the cluster-head. The cluster-head computes the average value within the cluster and broadcasts it to all nodes within the cluster. Let $\mathbf{y}(t) = [y_1(t), \dots, y_K(t)]^T$ be the vector containing the average values each cluster agrees on at time t . We have

$$y_k(0) = \frac{1}{n_k} \sum_{c(i)=k} x_i(0), \quad k = 1, \dots, K, \quad (2)$$

and the node values $\hat{x}_i(0) = y_{c(i)}(0)$, $i = 1, \dots, n$. Each gateway node sends to its neighboring gateway node the size of the cluster it belongs to, and forwards the received message to the cluster-head. Thus a cluster-head has the knowledge of the sizes of all neighboring clusters.

3.1. Fixed Linear Iteration

For fixed linear iteration, all nodes in the network share a global schedule. At time instant t , $t = 0, 1, \dots$, each gateway node exchanges its value with its neighboring gateway nodes. It then forwards the received values to the cluster-head. The cluster-head of cluster k updates the common value in the cluster with

$$y_k(t+1) = y_k(t) + \alpha \sum_{l \in \hat{N}(k)} n_l (y_l(t) - y_k(t)), \quad (3)$$

where $0 < \alpha < 1/\hat{d}_{\max}$ with $\hat{d}_{\max} = \max_{i \in V} \hat{d}_i$, and $\hat{N}(k)$ denotes the set of neighboring clusters of k . This value is then broadcasted to all nodes in the cluster. Node i updates its current value with $\hat{x}_i(t) = y_{c(i)}(t)$. Therefore, from node

i 's point of view, we have

$$\begin{aligned}\hat{x}_i(t+1) &= \hat{x}_i(t) + \alpha \sum_{\substack{j \in \tilde{N}(i) \\ c(j) \neq c(i)}} (\hat{x}_j(t) - \hat{x}_i(t)) \\ &= \hat{x}_i(t) + \alpha \sum_{j \in \tilde{N}(i)} (\hat{x}_j(t) - \hat{x}_i(t)).\end{aligned}\quad (4)$$

In matrix form we have

$$\hat{\mathbf{x}}(t+1) = \hat{\mathbf{W}}\hat{\mathbf{x}}(t) = (\mathbf{I} - \alpha\hat{\mathbf{L}})\hat{\mathbf{x}}(t).\quad (5)$$

In essence, the above algorithm achieves the linear iteration with constant edge weights [1] in graph \hat{G} . In the following, we compare the performance of the above algorithm with linear iteration with constant edge weights on the original graph, i.e., the weight matrix is given by $\mathbf{W} = \mathbf{I} - \alpha\mathbf{L}$, with $0 < \alpha < 1/d_{\max}$.

Definition 1. The convergence rate for fixed iteration is defined as [1]

$$r = \sup_{\hat{\mathbf{x}}(t) \neq \bar{x}\mathbf{1}} \frac{\|\hat{\mathbf{x}}(t+1) - \bar{x}\mathbf{1}\|_2}{\|\hat{\mathbf{x}}(t) - \bar{x}\mathbf{1}\|_2}.\quad (6)$$

For linear iteration with a symmetric stochastic weight matrix \mathbf{W} , $r = \lambda_2(\mathbf{W})$, the second largest eigenvalue of \mathbf{W} [1]. Given $\mathbf{W} = \mathbf{I} - \alpha\mathbf{L}$, we have $\lambda_2(\mathbf{W}) = \max\{\alpha\lambda_1(\mathbf{L}) - 1, 1 - \alpha\lambda_{n-1}(\mathbf{L})\}$, hence the following theorem.

Theorem 1: When $0 < \alpha < \frac{1}{d_{\max}}$, the cluster-based fixed linear iteration converges to $\bar{x}\mathbf{1}$ with rate

$$r = \max\{\alpha\lambda_1(\hat{\mathbf{L}}) - 1, 1 - \alpha\lambda_{n-1}(\hat{\mathbf{L}})\}.\quad (7)$$

In particular, if $0 < \alpha < \frac{1}{2d_{\max}}$, then $r = 1 - \alpha\lambda_{n-1}(\hat{\mathbf{L}})$.

Remark 1: On a geometric random graph $G(n, r)$, numerical results indicate that $\lambda_{n-1}(\mathbf{L}) + \lambda_1(\mathbf{L}) < 2d_{\max}$ always holds, which implies $\lambda_2(\mathbf{W}) = 1 - \alpha\lambda_{n-1}(\mathbf{L})$ for all $0 < \alpha < \frac{1}{d_{\max}}$. Hence, on $G(n, r)$, the optimal convergence rate $r^* = 1 - \frac{\lambda_{n-1}(\mathbf{L})}{d_{\max}}$ is achieved at the optimal weight $\frac{1}{d_{\max}}$ for the original linear iteration. Note that the assumption of the cluster-heads knowing d_{\max} is not restrictive, since d_{\max} can be obtained through a simple ripple algorithm that computes the maximum [9]. Furthermore, the above also holds for the graph \hat{G} generated from $G(n, r)$ by clustering. Therefore, the optimal convergence rate for the cluster-based linear iteration on $G(n, r)$ is $r^* = 1 - \frac{\lambda_{n-1}(\hat{\mathbf{L}})}{d_{\max}}$. Due to clustering, both d_{\max} and the algebraic connectivity $\lambda_{n-1}(\mathbf{L})$ increase, but the relative increase in d_{\max} is more dramatic than that of $\lambda_{n-1}(\mathbf{L})$. On $G(100, r(100))$, where $r(n) = \sqrt{\frac{2\log n}{n}}$, clustering causes $\frac{\lambda_{n-1}(\mathbf{L})}{d_{\max}}$ to increase from 0.0521 to 0.3331 (averaged over 1000 realizations). A similar roughly 6-fold increase can be observed for other values of n .

Remark 2: The ϵ -averaging time $T_{\text{ave}}(\epsilon)$ is defined as the minimum time required for the relative l_2 error to be bounded

by ϵ . Given the definition of the convergence rate r , it is easy to see that $T_{\text{ave}}(\epsilon) = \log(\frac{1}{\epsilon}) / \log(\frac{1}{r})$. Thus on $G(n, r)$ and for $\frac{\lambda_{n-1}(\mathbf{L})}{d_{\max}}$ small, $T_{\text{ave}}(\epsilon) \approx \log(\frac{1}{\epsilon}) \left(\frac{\lambda_{n-1}(\mathbf{L})}{d_{\max}}\right)^{-1}$. The above result implies that the cluster-based algorithm converges about 6 times faster for any n .

Remark 3: It appears that clustering does not improve the time-complexity of the linear iteration in the order sense, as clustering only effectively increases the transmission radius by a factor of about 2 to 3. However, the number of messages transmitted in each slot is reduced from $2|E|$ to $\Theta(|\tilde{E}|)$, where $|\tilde{E}|$ is the number of active links in \hat{G} . On the geometric random graph with $r(n) = \Theta(\sqrt{\frac{\log n}{n}})$, we have $|E| = \Theta(n \log n)$. On the other hand, $|\tilde{E}| = \Theta(K) = \Theta(\frac{n}{\log n})$, since by construction, the number of neighbors of a cluster is bounded with high probability. This demonstrates a $(\log n)^2$ gain in the message complexity.

3.2. Random Gossiping

Let \mathbf{P} be a $K \times K$ stochastic matrix with the condition that $P_{ij} > 0$ if and only if cluster i and cluster j are adjacent. The cluster-head of cluster k has a clock that ticks at times of a rate 1 Poisson process. Suppose the t -th tick belongs to the cluster-head of cluster i . It randomly picks one of its neighboring cluster j with probability P_{ij} , and sends a signal to the gateway node in cluster i that has an active link to a gateway node in cluster j . Both gateway nodes exchange their values, and forward the received value to their respective cluster head. The cluster head of i and j update their values with

$$y_i(t+1) = y_j(t+1) = \frac{n_i y_i(t) + n_j y_j(t)}{n_i + n_j}.\quad (8)$$

The new value is then broadcasted, and nodes in cluster i and j update their values with $\hat{x}_v(t) = y_{c(v)}(t)$. The sum within the network is retained. The number of message transmissions per round is at most 6 (which may be smaller if the gateway node is also the cluster-head). Therefore,

$$\hat{\mathbf{x}}(t+1) = \hat{\mathbf{W}}(t)\hat{\mathbf{x}}(t) = \hat{\mathbf{W}}_{i,j}\hat{\mathbf{x}}(t),\quad (9)$$

where

$$\hat{W}_{i,j}(u, v) = \begin{cases} \frac{1}{n_i + n_j} & c(u), c(v) \in \{i, j\} \\ 1 & u = v \text{ and } c(u) \notin \{i, j\} \\ 0 & \text{otherwise.} \end{cases}\quad (10)$$

Consider $\hat{\mathbf{W}} = E[\hat{\mathbf{W}}(t)] = \frac{1}{K} \sum_{i,j} P_{ij} \hat{\mathbf{W}}_{i,j}$. Let \mathbf{D} be an $n \times n$ diagonal matrix with entries

$$D_u = \sum_{j=1}^K (P_{c(u),j} + P_{j,c(u)}),\quad (11)$$

and \mathbf{Q} be an $n \times n$ matrix with entries

$$Q(u, v) = \begin{cases} \frac{1}{2} \frac{P_{c(u),c(v)} + P_{c(v),c(u)}}{n_{c(u)} + n_{c(v)}} & c(u) \neq c(v) \\ \frac{1}{2} \sum_{j=1}^K \frac{P_{c(u),j} + P_{j,c(u)}}{n_{c(u)} + n_j} & c(u) = c(v). \end{cases}\quad (12)$$

Then $\hat{\mathbf{W}}$ can be written as

$$\hat{\mathbf{W}} = \mathbf{I} - \frac{\mathbf{D}}{K} + \frac{2\mathbf{Q}}{K}. \quad (13)$$

If \mathbf{P} is doubly-stochastic, then $D_u = 2$, \mathbf{Q} is a stochastic matrix, and $\hat{\mathbf{W}} = (1 - \frac{2}{K})\mathbf{I} + \frac{2\mathbf{Q}}{K}$.

Definition 2. The ϵ -averaging time $T_{ave}(\epsilon, \mathbf{P})$ for cluster-based gossip algorithm is defined as the earliest time at which $\hat{\mathbf{x}}(t)$ is ϵ close to the true average with probability $1 - \epsilon$:

$$T_{ave}(\epsilon, \mathbf{P}) = \sup_{\hat{\mathbf{x}}(0)} \inf \left\{ t : \Pr \left(\frac{\|\hat{\mathbf{x}}(t) - \bar{x}\mathbf{1}\|_2}{\|\hat{\mathbf{x}}(0)\|_2} \geq \epsilon \right) \leq \epsilon \right\}. \quad (14)$$

Note that the relative error is measured with respect to the initial value $\hat{\mathbf{x}}(0)$ instead of $\mathbf{x}(0)$. It can be verified that for each t , $\hat{\mathbf{W}}(t)$ is a doubly stochastic, symmetric projection matrix, i.e., $\hat{\mathbf{W}}(t)^T \hat{\mathbf{W}}(t) = \hat{\mathbf{W}}(t)^2 = \hat{\mathbf{W}}(t)$. Thus, $\hat{\mathbf{W}}$ is symmetric, positive-definite and doubly-stochastic. By similar arguments in [2], the following theorem can be proved.

Theorem 2. The ϵ -averaging time of the cluster-based gossip algorithm is

$$T_{ave}(\epsilon, \mathbf{P}) = \frac{C \log \epsilon^{-1}}{\log \lambda_2(\hat{\mathbf{W}})^{-1}}, \quad (15)$$

where $C \in [0.5, 3]$ is a positive constant. Moreover, for large K and doubly-stochastic \mathbf{P} ,

$$T_{ave}(\epsilon, \mathbf{P}) = \frac{CK \log \epsilon^{-1}}{1 - \lambda_2(\mathbf{Q})}, \quad (16)$$

where $C \in [0.25, 1.5]$ is a positive constant.

Example: We assume that for gossip algorithms, each node i chooses one of its neighbors with equal probability $1/d_i$. Similarly define matrix \mathbf{P} for cluster-based gossip algorithm. Fig. 1 shows the relative error $\frac{\|\hat{\mathbf{x}}(t) - \bar{x}\mathbf{1}\|_2}{\|\hat{\mathbf{x}}(0) - \bar{x}\mathbf{1}\|_2}$ vs. the number of iterations for conventional gossip algorithm and the cluster-based gossip algorithm, where the result is averaged over 1000 realizations of $G(100, r(100))$ and initial values. It is evident that clustering significantly speeds up the convergence.

Remark 1: Consider the doubly-stochastic \mathbf{P} . From Theorem 2, the rapid convergence of the cluster-based gossip algorithm w.r.t. the conventional gossip can be attributed to two facts. Firstly, because the number of clusters is much less than the number of nodes, the value of every node is updated more often under cluster-based gossip (as manifested by the substitution of n by K). Secondly, since the graph is better connected after clustering, the random walk on the graph mixes faster, i.e., the mixing time $\frac{1}{1 - \lambda_2(\mathbf{Q})}$ becomes smaller than the corresponding walk on the original graph.

Remark 2: In the above, the ϵ -averaging time is measured in the relative sense, as opposed to the absolute time considered in [2]. The message complexity of asynchronous gossip algorithms is of the same order as the relative averaging time, since the number of transmissions per time instant is bounded and independent of n .

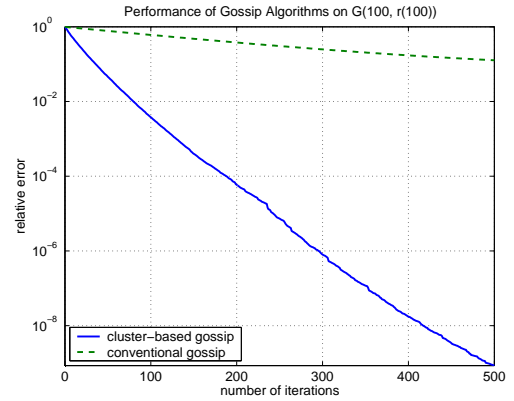


Fig. 1. Decay of relative norm-2 error for conventional and cluster-based gossip algorithms on $G(100, r(100))$

4. CONCLUSION

Cluster-based distributed averaging algorithms in forms of fixed linear iteration and random gossiping are proposed. Both are shown to provide much faster convergence and reduced communication and computation complexity than their non-cluster-based variants.

5. REFERENCES

- [1] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," in *IEEE Conf. on Decision and Control*, Maui, Hawaii, Dec. 2003.
- [2] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2506–2530, 2006.
- [3] 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.
- [4] A. Giridhar and P.R. Kumar, "Computing and communicating functions over sensor networks," *IEEE Journal on Selected Areas of Communication*, vol. 23, no. 4, pp. 755–764, Apr. 2005.
- [5] M. R. Garey and D. S. Johnson, *A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, 1979.
- [6] F. Kuhn and R. Wattenhofer, "Constant-time distributed dominating set approximation," in *22nd ACM Int. Symposium on the Principles of Distributed Computing (PODC)*, 2003, pp. 25–32.
- [7] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.
- [8] H. E. Bell, "Gerschgorin's theorem and the zeros of polynomials," *Amer. Math. Monthly*, vol. 72, pp. 292–295, 1965.
- [9] N. Khude, A. Kumar, and A. Karnik, "Time and energy complexity of distributed computation in wireless sensor networks," in *IEEE Infocom 2005*, Miami, FL, Mar. 2005, pp. 2625–2637.