Distributed protocol for determining when averaging consensus is reached

Vikas Yadav, and Murti V Salapaka

***Abstract*— Distributed averaging over a large network is a well studied problem that converges asymptotically; however, existing protocols does not provide a way for each node to distributively detect the occurrence of convergence. In this paper a method is developed to distributively determine when the consensus has reached within a given error margin. In absence of such a method all nodes in the network keep running the required computation and communication even if the consensus value are within acceptable tolerance, which is not preferable as in large-scale distributed networks resources like power are limited. Furthermore, this extra communication can cause signal interference with other critical information. This distributed detection takes finite time and occurs at each node simultaneously.**

I. Introduction

Consensus or agreement in a large network of agents refers to the event in which each agent has same information. It is assumed that each node is sharing information with its neighbors. Averaging consensus is a special case where the each node starts with some initial node-value and as a result of agreement it obtains a value which is an average of initial node-values of all the nodes in the network. Averaging consensus protocol refers to the action to be performed on the received information. In this paper, the focus is on the linear averaging protocol presented in [10] where each node takes an average of the information received from neighboring nodes. [10] provides a necessary and sufficient condition that underlying network is strongly connected and balanced which leads to an asymptotic convergence in absence of any malicious user. It is shown in [12] that the requirement for balanced graph can be dropped by using weighted integrators in the protocol. A faster linear averaging protocol similar to [10] is proposed in [13]. In [2] a condition on functions that can be computed distributively is provided. A good survey of consensus problems is provided in [11], [9]. In [8] authors provide convergence analysis for the angular interaction among agents using a switched linear model. This model also assumes that over every finite period of time the particles are jointly connected for the length of the entire interval. Similar agrement problem over random graphs is addressed in [7] for graphs having binomial distribution. Most of these works assume some kind of connectivity in the network. In [3] work is done towards maintaining the connectivity of network by controlling the algebraic connectivity (also known as the second smallest eigenvalue of Laplacian of graph) of the network. In [6] authors address agreement problem over geometric random graphs with noisy communication. They showed the convergence in presence of a modified update rule where the nearest neighbor value is scaled by a special time varying step size. A malicious or faulty node is one which is not following the consensus protocol. In presence of such nodes, the averaging protocol becomes unstable, i.e. it fails to converge. In [1] authors provide results on stabilizing consensus protocol in presence of faults by assuming that node-values are binary.

In large sensor networks, each node has limited power for its computational and communication need. In all consensus protocols, convergence takes place in asymptotic sense and there is no distributed way for each individual node to know if the convergence has reached within desired error margin. If each node can detect the consensus occurrence, then they can stop doing computation and communication required by the consensus protocol, and thus saving on the limited power supply. In this paper, a distributed algorithm is presented which facilitates each node to detect the occurrence of consensus within desired bounds in finite time. This algorithm requires implementation of maximum and minimum consensus protocols, which have finite convergence time bounded by the diameter of the network.

The paper is organized as follows: In Section 2 a mathematical setup for the problem is presented with an overview of different protocols. In Section 3 the main scheme to achieve finite time convergence is presented. In Section 4 some examples are presented. Finally conclusion of the paper and discussion on future research directions is presented in Section 5.

II. Problem setup

Consider a system of N nodes or agents connected with each other in an arbitrary manner via communication links. Each node is sensing its local information e.g. local temperature or chemical concentration and is trying to compute the average of that local information over the whole network. The system is modelled as a graph $G := (V, E)$ consisting of a set $V := \{1, 2, ..., N\}$ of elements called vertices or nodes or agents, and a set E of node pairs called edges, with $E \subseteq E_c := \{(i, j)|i, j \in V\}$. If $E = E_c$ i.e. each node is connected to rest of $n-1$ nodes, it is called a complete graph. A graph is called undirected if for every pair of distinct nodes i and j both (i, j) and (j, i) are in E. Otherwise, it is called a directed graph or a digraph. A simple graph is a graph with no self loops, i.e. $(i, j) \notin E$ if $i = j$. A graph is connected if it has a path between each pair of distinct nodes i and j, where by a path between nodes i and j we mean a sequence of distinct edges of G of the

Email: {vikasy, murti}@iastate.edu

form $(i, k_1), (k_1, k_2), \ldots, (k_m, j) \in E$. A digraph is called "strongly connected" if there is a directed path between each pair of distinct nodes. Diameter D of a graph is the longest shortest path between any two pair of nodes. Fixed graphs are graphs in which the edge set E does not change with time. In this paper, fixed graphs are considered.

Radius r of node pair (i, j) implies the minimum path length, i.e. the minimum number of edges connecting i to j is equal to r. The neighborhood N_i of i^{th} node is a set consisting of all nodes within radius 1 not including the i^{th} node itself. The degree or out-degree of an i^{th} node is $|N_i|$, where $|N_i|$ denotes the number of elements in N_i. The maximum degree of the graph is denoted by Δ and the minimum degree of the graph is denoted by δ. The adjacency matrix $A = \{a_{ij}\}$ of a graph G is an $N \times N$ matrix. $a_{i,j} > 0$ only if the node pair $(i, j) \in E$ and is equal to zero otherwise. The graph G is assumed to be simple, which implies that $a_{i,i} = 0$ for all $i = 1, 2, \cdots N$. The diagonal matrix Φ is an $N \times N$ diagonal matrix with each diagonal entry $d_{ii} = \sum_{j=1}^{N} a_{ij}$. For undirected graph the graph Laplacian matrix L is defined as $\Phi - A$.

The graph Laplacian matrix L is an important function of the graph G. Eigenvalues of L have direct relation to the connectivity of the graph. Let, $\lambda_1 \leq \lambda_2 \leq \cdots \lambda_N$ be N eigenvalues of L. Since L has row sum equal to zero (such matrices are called row stochastic), $\lambda_1 = 0$ is a trivial eigenvalue of L with $\bar{1} := [1, \cdots, 1]^T$ as the corresponding eigenvector i.e. $L\bar{1} = 0$. A graph is connected if and only if the second smallest eigenvalue of Laplacian is non-zero i.e. $\lambda_2 > 0$ [4], and larger the λ_2 better is the connectivity of the graph and faster is the convergence of the distributed consensus protocol. The second smallest eigenvalue λ_2 is also called the algebraic connectivity of the graph. It is assumed that the communication among nodes is noiseless.

A. Average consensus protocol

The state vector of node-values for average consensus protocol is defined by column vector $x(k) = (x_1(k)\ x_2(k) \cdots x_N(k))^T$. The average consensus protocol denoted by AP distributively computes the average of a given initial node-values $x(0) = (x_1(0)\ x_2(0) \cdots x_N(0))^T$. It takes $x(0)$ as an input and generates a sequence of node-values $x(k)x(k) = (x_1(k)\ x_2(k) \cdots x_N(k))^T$ such that $\{x(k)\}_{k=1}^{\infty} = AP(x(0))$ based on the following nearest-neighborhood update rule:

$$x_i(k+1) = x_i(k) + \epsilon \sum_{j \in N_i} a_{ij}(x_j(k) - x_i(k)) \quad (1)$$

for all $i = 1, 2 \cdots N$. This implies

$$x(k+1) = Px(k) \quad (2)$$

where $P = I - \epsilon L$. Since $I - \epsilon L \approx exp(-\epsilon L)$, discrete time average consensus can be seen as the first order approximation of continuous time average consensus problem which is given by $\dot{x} = -\epsilon Lx$. It is known that P with $0 < \epsilon < \frac{1}{d_{max}}$, where $d_{max} = \max d_{ii}$ satisfies following properties [9]:

1) P is row-stochastic non-negative matrix with a trivial eigenvalue of 1,
2) All eigenvalues of P are inside the unit circle,
3) If G is strongly connected then P is a primitive matrix,
4) If G is a balanced graph ($\bar{1}^T L = 0$) then P is column-stochastic ($\bar{1}^T P = \bar{1}$). Note that every undirected graph is a balanced graph.

We will make following assumptions throughout the paper:

Assumption 2.1: (a) $0 < \epsilon < \frac{1}{d_{max}}$, (b) the graph G is connected, and (c) if the graph G is directed graph, then it is "strongly connected" and balanced.

The average consensus protocol for the graph G given by (1) converges asymptotically to average of the initial condition $x(0)$ denoted by $\alpha := \frac{1}{N}\sum_{i=1}^{N} x_i(0)$ [9]. The average value $\alpha\bar{1}$ is an invariant quantity of the dynamics given by (1) i.e. $P(\alpha\bar{1}) = \alpha\bar{1}$. Further, this convergence is reached exponentially with exponent bounded above by μ_2, which is the second largest eigenvalue of P ($\mu_2 < 1$). Following property of P which relies on the fact that $0 < \epsilon < \frac{1}{d_{max}}$ is needed the rest of the development.

Proposition 2.1: Let p_{ij} be $(i, j)^{th}$ element of P. Then, $0 \leq p_{ij} < 1$ for all $i, j = 1, 2 \cdots N$. Moreover, $p_{ii} > 0$ for all i.

Proof: Since P is a non-negative matrix, it implies that $p_{ij} \geq 0$. Since, $0 < \epsilon < \frac{1}{d_{max}}$, $0 < p_{ii} = 1 - \epsilon d_{ii} < 1$; and for $i \neq j$, $p_{ij} = \epsilon a_{ij} < \frac{a_{ij}}{d_{max}} \leq 1$. Thus, $p_{ij} < 1$ for all $i, j = 1, 2 \cdots N$ with $p_{ii} > 0$ for all i. Also, for all $j \in N_i$, $p_{ij} = \epsilon a_{ij} > 0$. ■

The average protocol update rule can be rewritten as:

$$x_i(k+1) = \sum_{j=1}^{N} p_{ij} x_j(k) \quad (3)$$

at i^{th} node. Thus, each updated node-value is a weighted average of its neighboring node-values such that weights are non-negative and strictly less than one with $\sum_{j=1}^{N} p_{ij} = 1$ for all i. This leads to the following conclusion:

Proposition 2.2:

$$x_i(k+1) \leq \max_j x_j(k) \quad \text{for all } i = 1, 2 \cdots, N; \quad (4)$$

$$x_i(k+1) \geq \min_j x_j(k) \quad \text{for all } i = 1, 2 \cdots, N. \quad (5)$$

Equalities hold in above equations if and only if $x_i(k) = x_j(k)$ for all $i = 1, 2 \cdots, N$. ■

By taking maximum over all nodes in (4) (and minimum over all nodes in (5)) it can be shown that:

$$\max x(k+1) := \max_j x_j(k+1) \leq \max_j x_j(k) \quad (6)$$

$$\min x(k+1) := \min_j x_j(k+1) \geq \min_j x_j(k) \quad (7)$$

where equalities hold in both cases if and only if $x_i(k) = x_j(k)$ for all $i, j = 1, 2 \cdots, N$. Combining this with Proposition 2.2, it can be shown that node-value $x_i(k)$ at any time k is bounded from above by the maximum value in network in the past and below by the minimum value in the network in the past. This can be expressed as follows:

$$\min x(k') \leq x_i(k) \leq \max x(k') \quad \text{for all } i = 1, 2 \cdots, N. \quad (8)$$

and for all $k \geq k'$.

The following lemma states that if a node reaches an average consensus protocol node-value that is strictly less than the maximum over the network at some past time instant k' then the node-value at that node at any future time instant $k > k'$ remains strictly less than the maximum over the network at the past time instant k'.

Lemma 2.1: Consider a graph G (undirected or directed "strongly connected", balanced graph) running an average consensus protocol AP given by (1) with an initial condition $x(k')$. Let, i and i' be nodes such that $x_i(k) < \max x(k')$ and $x_{i'}(k) > \min x(k')$, respectively for some time instant $k \geq k'$. Then for all $k'' \geq k$:

$$x_i(k'') < \max x(k')$$
$$x_{i'}(k'') > \min x(k')$$

Proof: It is given that for node i, $x_i(k) < \max x(k')$ for some time instant $k \geq k'$. It follows that:

$$\begin{aligned} x_i(k+1) &= \sum_j p_{ij} x_j(k) \\ &= p_{ii} x_i(k) + \sum_{j \neq i} p_{ij} x_j(k) \\ &\leq p_{ii} x_i(k) + \sum_{j \neq i} p_{ij} \max x(k) \\ &\leq p_{ii} x_i(k) + \sum_{j \neq i} p_{ij} \max x(k') \text{ [From Prop. 2.2]} \\ &= p_{ii} x_i(k) - p_{ii} \max x(k') + \sum_j p_{ij} \max x(k') \\ &= p_{ii} x_i(k) + (1 - p_{ii}) \max x(k') \\ &< p_{ii} \max x(k') + (1 - p_{ii}) \max x(k') \ [\because p_{ii} > 0] \\ &= \max x(k'). \end{aligned}$$

Thus, $x_i(k+1) < \max x(k')$. It follows that $x_i(k+J) < \max x(k')$ for all $J \geq 1$. Therefore, if node i assumes a node-value $x_i(k) < \max x(k')$, then it remains strictly less than $\max x(k')$ for all future time instances. Similar proof holds for the minimum value case. ■

Next lemma shows that after D time steps the maximum value has to strictly decrease and the minimum value has to strictly increase.

Lemma 2.2: Consider a graph G (undirected or directed "strongly connected", balanced graph) running an average consensus protocol AP given by (1) with an initial condition $x(k')$ such that $\max x(k') > \min x(k')$. Then for all $k \geq k' + D$:

$$\max x(k) < \max x(k'), \text{ and} \quad (9)$$
$$\min x(k) > \min x(k'). \quad (10)$$

Proof: Consider any particular node j. There exists a node i such that $x_i(k') < \max x(k')$ as $\min x(k') < \max x(k')$. The shortest distance between node i and j, denoted by d, is less than or equal to the diameter D of the graph. Let the path connecting i and j be $(i, m_1), (m_1, m_2), \ldots, (m_{d-1}, j)$. Because of weighted averaging, at time $k = k' + 1, x_{m_1}$ will become strictly less than $\max x(k')$ as shown below:

$$\begin{aligned} x_{m_1}(k'+1) &= \sum_{n=1}^{N} p_{m_1 n} x_n(k') \\ &\leq p_{m_1 i} x_i(k') + \sum_{n \neq i} p_{m_1 n} \max x(k') \\ &< p_{m_1 i} \max x(k') + (1 - p_{ii}) \max x(k') = \max x(k'). \end{aligned}$$

Thus, $x_{m_1}(k'+1) < \max x(k')$. Therefore, from Lemma 2.1 for all $k'' \geq k' + 1$, $x_{m_1}(k'') < \max x(k')$. It follows that for all $k'' \geq k' + 2$, $x_{m_2}(k'') < \max x(k')$ and that for all $k'' \geq k' + d_{ij} - 1$, $x_j(k'') < \max x(k')$. Note that $k' + D \geq k' + d_{ij} - 1$ ($D \geq d_{ij}$), therefore for all $k'' \geq k' + D$, $x_j(k'') < \max x(k')$. ■

Thus from lemma 2.2, after a finite time given by the diameter D of the graph, all node-values under averaging consensus protocol become strictly less than the maximum value in network in the past and strictly greater than the minimum value in the network in the past, which in turn means that after a finite time the maximum value in the network decreases and the minimum value in the network increases.

B. Maximum consensus protocol

The maximum consensus protocol denoted by MXP distributively computes the maximum of a given initial node-values $z(0) = (z_1(0) z_2(0) \cdots z_N(0))^T$. It takes $z(0)$ as an input and generates a sequence of node-values $z(k)$ i.e. $\{z(k)\}_{k=1}^{\infty} = MXP(z(0))$ based on the following update rule:

$$z_i(k+1) = \max_{j \in N_i} z_j(k), \quad (11)$$

where $z_i(k)$ is the node-value of i^{th} node for maximum consensus protocol. Each node updates its value to the present maximum value in its neighborhood. The overall state vector for maximum protocol is defined by the column vector $z(k) = (z_1(k) z_2(k) \cdots z_N(k))^T$. Note that $z_i(k)$ is a non-decreasing function with time k.

Proposition 2.3: Maximum consensus protocol MXP given by (11) converges to $\max z(0)$ in finite time $T \leq D$.

Proof: Let m be a node with node-value at $z_m(0) = \max z(0)$. Due to connectedness of graph G, each node in graph is connected to node m. Let, $\tilde{D}$ be the maximum distance between m and any other node, then $\tilde{D} \leq D$. At time $k = 1$ all nodes connected to m at one unit distance (one hop) will have the maximum value, at time $k = 2$ all nodes connected to m at two unit distance (two hops) will have the maximum value, and so on. Thus, by time $T = \tilde{D}$ all the nodes will have maximum value. ■

C. Minimum consensus protocol

The minimum consensus protocol denoted by MNP distributively computes the minimum of a given initial node-values $y(0) = (y_1(0)y_2(0)\cdots y_N(0))^T$. It takes $y(0)$ as an input and generates a sequence of node-values $y(k)$ i.e. $\{y(k)\}_{k=1}^{\infty} = MXP(y(0))$ based on the following update rule:

$$y_i(k+1) = \min_{j \in N_i} y_j(k), \tag{12}$$

where $y_i(k)$ is the node-value of i^{th} for minimum consensus protocol. Each node updates its value to the present minimum value in its neighborhood. The overall state vector for minimum protocol is defined by the column vector $y(k) = (y_1(k)y_2(k)\cdots y_N(k))^T$. Further, $y_i(k)$ is a non-increasing function with time k.

Proposition 2.4: Minimum consensus protocol given by (12) converges to $\max y(0)$ in finite time $T \leq D$.

Proof: Similar to the proof of Proposition 2.3. ∎

III. Finite time convergence within a given error margin

Consider the graph $G = (V, E)$ with N nodes as defined above, each node running a distributed average consensus protocol AP given by (1). In this section, a distributed algorithm is provided which enables each node to detect the occurrence of the convergence in the network within a given error margin in finite time. To achieve this each node runs two more protocols, a maximum consensus protocol MXP and a minimum consensus protocol MNP given by (11) and (12) , respectively with $z(k_0) = y(k_0) = x(k_0)$, where k_0 is the time when maximum and minimum protocols are started. By finite time convergence we imply that for any given $\rho > 0$, all agents can simultaneously reach to a decision in some finite time T_c that their node-values are ρ close to the desired average value i.e. they are in the interval $[\alpha-\rho, \alpha+\rho]$. From Proposition 2.2 and 2.3, we have that after time $k = k_0 + D, z(k) = \max x(k_0)\bar{1}$ and $y(k) = \min x(k_0)\bar{1}$. Thus, at $k = k_0 + D$ the difference $z_i(k) - y_i(k)$ will be same at each node.

Define $T(j) = (j-1)D$ for $j = 1, 2, \cdots$, as the set of time instants when MXP and MNP are reset. This is done at $k = T(j)$ by setting their initial conditions $z(T(j))$ and $y(T(j))$ equal to the current node-values $x(T(j))$ from AP. Thus, at every time instant $k = T(j+1)$, MXP running at each node with initial value $z(T(j))$ will output $\max z(T(j))$ and MNP running at each node with initial value $y(T(j))$ will output $\min y(T(j))$. Define these outputs of MXP and MNP as $\bar{\alpha}(j) = \max z(T(j))$, $\underline{\alpha}(j) = \min y(T(j))$, respectively and the difference between these two outputs as $\beta(j) = \bar{\alpha}(j) - \underline{\alpha}(j)$. At $k = T(j+1)$ each node will have the same value $\beta(j)$. Following corollary shows that $\bar{\alpha}(j)$ and $\underline{\alpha}(j)$ both converge to α, which in turn implies that $\beta(j)$ converges to 0.

Lemma 3.1: The sequences $\bar{\alpha}(j)$ and $\underline{\alpha}(j)$ converge to α as $j \to \infty$. Further, the sequence $\beta(j)$ converges to 0 as $j \to \infty$.

Proof: From [9] it is given $\{x(k)\}_{k=1}^{\infty}$ converges to α i.e.

$$\lim_{k \to \infty} x_i(k) = \alpha$$

for all $i = 1, 2, \cdots N$. Thus, for any $\epsilon > 0$ there exists K such that for all $k \geq K$ implies:

$$\begin{aligned}
&|x_i(k) - \alpha| < \epsilon \text{ for all } i = 1, 2, \cdots N \\
&\Rightarrow -\epsilon < x_i(k) - \alpha < \epsilon \text{ for all } i = 1, 2, \cdots N \\
&\Rightarrow -\epsilon < \max x(k) - \alpha < \epsilon \\
&\Rightarrow |\max x(k) - \alpha| < \epsilon \\
&\Rightarrow \lim_{k \to \infty} \max x(k) = \alpha \\
&\text{Similarly, } \lim_{k \to \infty} \min x(k) = \alpha
\end{aligned}$$

Now, $\bar{\alpha(j)} = \max x_i(jD)$ and $\underline{\alpha}(j) = \min x_i(jD)$. So, they are subsequences of convergent sequences converging to same limit α, thus both $\bar{\alpha}(j)$ and $\underline{\alpha}(j)$ converge to α as $\bar{j} \to \infty$. Further, note that $\beta(j) = \bar{\alpha}(j) - \underline{\alpha}(j)$, therefore $\beta(j)$ converges to 0 as $j \to \infty$. ∎

This leads to the following distributed algorithm, which is the main result of the paper. It helps each node in deducing the occurrence of convergence in the network in finite time within desired error margin ρ.

Algorithm I:

Initialization: Given initial condition $x(0)$, set $z(0) = x(0)$ and $y(0) = x(0)$. Start $AP(x(0))$, $MXP(z(0))$ and $MNP(y(0))$. Set $j = 1$.

Step 1: At $k = T(j) + D$, let $\bar{\alpha}(j) = MXP(z(T(j))), \underline{\alpha}(j) = MNP(y(T(j)))$ and $\beta(j) = \bar{\alpha}(j) - \underline{\alpha}(j)$. Check at each node if $\beta(j) < \rho$; If yes then Stop, else set $j = j + 1$.

Step 2: At $k = T(j)$, set $z(T(j)) = x(T(j))$ and $y(T(j)) = x(T(j))$. Go to Step 1.

Next theorem with help of Lemma 3.1 shows that the Algorithm I terminates in finite time.

Theorem 3.1: Algorithm I terminates in some finite time $T_c < \infty$.

Proof: $\beta(j)$ converges to 0 as $j \to \infty$ (from Lemma 3.1). Thus, for any given $\rho > 0$, there exists an integer j_0 such that $\beta(j) < \rho$ for all $j \geq j_0$. This implies that the Algorithm I converges in finite time $T_c = T(j_0)$. ∎

The finite time T_c is not known beforehand because the size of $\beta(j)$'s which is proportional to the algebraic connectivity of the graph is not known to each node beforehand. The significant achievement is that all nodes can deduce in some finite time that the consensus in the network has reached and this happens at the same time at each node without help of any centralized information or source.

Remark 1: In above algorithm, the maximum and minimum protocols are getting reset after every D time. The value of j at the termination of above algorithm gives number of times maximum and minimum protocols are executed. This number can be reduced at the cost of delaying the detection of convergence by choosing $T(j) = (j-1)D + \Delta T_j$ where $\Delta T_j \geq 0$ for all $j = 1, 2, \cdots$. One heuristic way to choose ΔT_j is by estimating the rate of decrease in the difference between maximum and minimum of node-values and setting ΔT_j equal to that estimated rate. In fact, above algorithm should work for all the graphs with diameter bounded by D_{max} at the expense of delaying the detection of occurrence of convergence by time bounded by the difference between D_{max} and actual diameter D. In other words, for this scheme to work it is not required for each node to know the actual diameter of the graph instead all it needs is some upper bound value on the diameter. In [5] a distributed method for computing the diameter of a graph is presented which uses a maximum of $2N^2$ messages. Each node can first run this protocol to determine D in a distributed manner. Diameter D is the only parameter of the network graph required by each node.

Remark 2: This detection of convergence technique can be generalized to distributed protocols such that $x(k)$ satisfies Lemma 2.2, i.e. the maximum and minimum of $x(k)$ over all nodes is strictly decreasing and increasing after every finite time D.

IV. Examples: Finite time convergence

In this section, we present three scenarios of averaging protocol, and show how the algorithm presented in previous section facilitates distributed detection of occurrence of consensus in the network.

Scenario A is of an undirected graph G_1 with 25 nodes. The diameter of graph is 4, and the algebraic connectivity of the graph is 1.79. It has maximum degree of 12 and minimum degree of 2. The initial condition $x(0)$ is chosen from a uniform distribution between $+10$ and -10. The average value $\alpha = 0.95$. Each node comes to know when the consensus has reached within an error margin of $\rho = 0.02$. The simulation result shown in Figure 1 demonstrates that after $k = 45$, each node correctly concludes that the convergence has occurred in the network within an error margin of 0.02.

Scenario B is another undirected graph G_2 with 25 nodes, diameter of the graph is 4, the algebraic connectivity of the graph is 1.48. It has maximum degree of 11 and minimum degree of 2. The average value $\alpha = -0.84$. In this case, after consensus has reached and detected, there is some change in one of node-values at $k = 60$, so that the new average value becomes -0.65. The algorithm presented in previous section starts automatically to find another occurrence of convergence due to this change. The simulation result shown in Figure 2 demonstrates that each node comes to know when the consensus has reached within an error margin of 0.02 at $k = 85$.

Scenario C is of a directed, strongly connected and balanced graph G_3 with 25 nodes. The diameter of graph is 11 , and the algebraic connectivity of the graph is 0.17. It has maximum degree of 5 and minimum degree of 1. The initial condition $x(0)$ is chosen from a uniform distribution between $+10$ and -10. The average value $\alpha = 1.07$. The

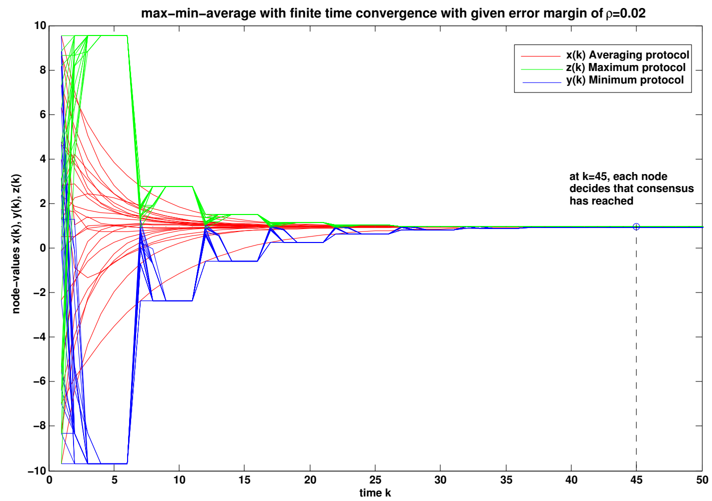

Fig. 1. A: Maximum-minimum protocol running in parallel with averaging protocol helps individual agents to make a decision about the occurrence of agreement in the network

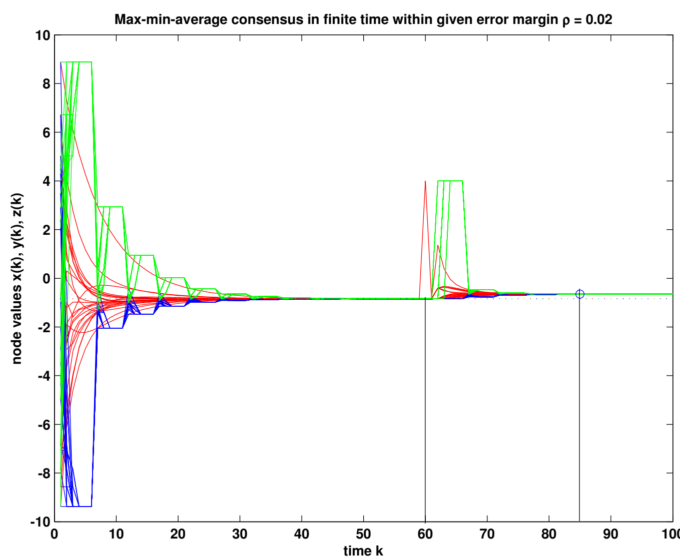

Fig. 2. B: One node changes it value at $k = 60$ after the agreement has reached in the network. Maximum-minimum protocol and averaging protocol restart to detect next occurrence of agreement in the network

simulation result shown in Figure 3 demonstrates that each node comes to know when the consensus has reached within an error margin of $\rho = 0.02$ at $k = 96$.

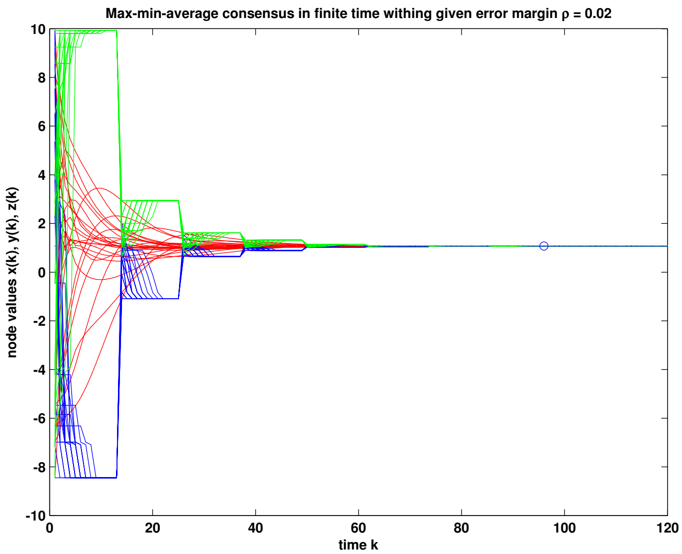

Fig. 3. C: A case of directed graph. Maximum-minimum protocol running in parallel with averaging protocol helps individual agents to make a decision about the occurrence of agreement in the network

V. Conclusions and Future Work

In this paper, a methodology is provided for detection of occurrence of consensus in the network running a distributed averaging protocol such that after finite time each node comes to know that the consensus has reached within given error bounds. This method requires two more protocols viz. maximum and minimum consensus protocols to run along with the averaging protocol at each sensor. The maximum and minimum protocols are reset after every D time which is the diameter of the graph. This is a conservative wait time, as mostly these protocols converge faster than D time. Thus, as part of future work this wait time can be further refined to a lower value such as period the graph.

References

[1] D. Angluin, M.J. Fischer, and H. Jiang. Stabilizing Consensus in Mobile Networks. *Proc. International Conference on Distributed Computing in Sensor Systems (DCOSS06), June*, 2006.

[2] D. Bauso, L. Giarre, and R. Pesenti. Distributed Consensus in Networks of Dynamic Agents. *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 7054–7059, 2005.

[3] M. C. DeGennaro and A. Jadbabaie. Decentralized control of connectivity for multi-agent systems. *IEEE Conference on Decision and Control, San Diego, CA. December*, 2006.

[4] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98):298–305, 1973.

[5] S. Haldar. An All Pairs Shortest Paths Distributed Algorithm Using 2n2Messages. *Journal of Algorithms*, 24(1):20–36, 1997.

[6] Y. Hatano, A. K. Das, and M. Mesbahi. Agreement in presence of noise: Pseudogradients on random geometric networks. *Proceedings of the 44th IEE Conference on Decision and Control*, pages 6382–6387, Dec. 2005.

[7] Y. Hatano and M. Mesbahi. Agreement over random networks. *IEEE Trans. Autom. Contr.*, 50(11):1867–1872, Nov. 2005.

[8] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Trans. Automat. Contr.*, 48(6):988–1001, 2003.

[9] R. Olfati-Saber, J.A. Fax, and R.M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[10] R. Olfati-Saber and R. M. Murray. Consensus protocols for network of dynamic agents. *IEEE Trans. on Automatic Control*, 49(9):1520–1533, Sept. 2004.

[11] W. Ren, RW Beard, and EM Atkins. A survey of consensus problems in multi-agent coordination. *American Control Conference, 2005. Proceedings of the 2005*, pages 1859–1864, 2005.

[12] C.W. Wu. Agreement and consensus problems in groups of autonomous agents with linear dynamics. *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pages 292–295, 2005.

[13] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, 5, 2003.