PRAMANA
— journal of
physics

Vol. 79, No. 3
September 2012
pp. 493–499

A simple consensus algorithm for distributed averaging in random geographical networks

MAHDI JALILI
Department of Computer Engineering, Sharif University of Technology, Azadi Avenue, Tehran, Iran
E-mail: mjalili@sharif.edu

MS received 6 February 2012; revised 5 April 2012; accepted 18 April 2012

Abstract. Random geographical networks are realistic models for wireless sensor networks which are used in many applications. Achieving average consensus is very important in sensor networks and the faster the consensus is, the durable the sensors' life, and thus, the better the performance of the network. In this paper we compared the performance of a number of linear consensus algorithms with application to distributed averaging in random geographical networks. Interestingly, the simplest algorithm – where only the degree of receiving nodes is needed for the averaging – had the best performance in terms of the consensus time. Furthermore, we proved that the network has guaranteed convergence with this simple algorithm.

Keywords. Sensor networks; random geographical networks; distributed averaging; consensus algorithms.

PACS Nos 89.75.Hc; 89.75.Fb; 89.20.Ff

1. Introduction

Wireless sensor networks are increasingly used in many applications ranging from environmental to military cases [1–3]. In such networks, the sensors (or nodes) sense, communicate and make decision on some parameters (e.g. temperature or humidity) in a distributed manner without traditional infrastructures (e.g. routers). The nodes repeatedly exchange their information with their neighbouring nodes to come to a conclusion about the environmental phenomenon, e.g. temperature, in the area covered by the network. If the network is carefully designed, this way of distributed monitoring is advantageous over the case where each sensor sends its information to the base station. Especially, the network will have better energy consumption and robustness properties [1–3].

Often, in a sensor network, the nodes need to reach consensus on the sensing parameters [4,5]. Therefore, a consensus (or synchronization) algorithm, i.e. a communication

DOI: 10.1007/s12043-012-0335-0; ***e*Publication: 31 August 2012**

protocol between the nodes, should be used [4–7]. Since the nodes in a sensor network are cheap, unreliable, with limited computational power and limited battery life, the consensus algorithm should be as simple as possible. Also, the algorithm should be designed in such a way to force the nodes to reach consensus in a short time, as the shorter the consensus time is, the less the energy consumption, and thus, the more durable the network.

Synchronization is the most common form of collective behaviour observed in dynamical networks [8,9], which can be enhanced through structural alteration in the network [10,11]. The consensus (or synchronization) can be achieved through linear, nonlinear, adaptive, local, distributed, or time-varying coupling between the nodes [4,12–14]. It is also possible to consider communication delay in the network and seek for consensus rules [15]. Performance of a consensus algorithm, i.e. consensus time, can be linked to the characteristics of the connection graph, e.g., the second smallest eigenvalue of the Laplacian matrix [16,17]. Ability of the network for providing better consensus can be improved by changing its topology, e.g., adding new edges to the network [18]. Having the global structure of the connection graph, the optimal communication protocol, i.e. coupling weighting between the nodes, can be designed. However, this needs global information on the network structure that cannot be attained in many cases. Alternatively, a number of protocols that only need local information can be used to do the job [17]. In this paper we show that a simple local communication protocol can lead to the consensus in a reasonably short time.

2. Consensus in discreet-time systems

The consensus can be studied in continuous-time or discrete-time dynamical systems. In this work, we considered discrete-time systems; however, the results are also valid for continuous-time systems. Let us suppose that N nodes measure the quantities $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ (e.g. temperature, humidity, or time) and the network has to make a decision on the observed phenomenon by computing a function $f(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ of the measurements. The aim of a consensus algorithm is to make the $\mathbf{x}_i$'s the same through an iterated process. We considered a simple linear coupling, i.e., communication protocol, where the nodes linearly influence each other without any communication delay. More precisely, the difference equations for the network read as

$$\mathbf{x}_i(k+1) = \mathbf{x}_i(k) + \sum_{j=1}^{N} \sigma_{ij} a_{ij} (\mathbf{x}_j(k) - \mathbf{x}_i(k)), \qquad i = 1, 2, ..., N, \tag{1}$$

where $\mathbf{x}_i \in \mathbb{R}^d$ are d-dimensional state vectors (measured quantities), $\sigma = \sigma_{ij} > 0$ is the matrix of the step sizes and $A = a_{ij}$ is the binary adjacency matrix. The above equations can be rewritten as

$$\mathbf{x}_i(k+1) = P\mathbf{x}_i(k), \qquad i = 1, 2, ..., N, \tag{2}$$

where $P = I - \sigma \times L$ is called the Perron matrix ($\times$ is the array multiplication operator), $L = D - A$ is the Laplacian matrix (D is a diagonal matrix with degree of the nodes in the diagonal entries) and I is the identity matrix.

It can be simply proved that for the values of the uniform step size σ in the range $(0,1/k_{\max}]$, with $k_{\max}$ being the maximum degree of the graph, the above system is asymptotically globally convergent to [17]

$$\forall i; \qquad \lim_{k\to\infty} \mathbf{x}_i(k) = \boldsymbol{\alpha} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i(0), \tag{3}$$

which is indeed an average of the initial measurements. With this configuration, the consensus time is at least as fast as $1-\mu_2$, where μ_2 is the second smallest eigenvalue of $\sigma \times L$ [4]. It is worth mentioning that when the step size is larger the consensus can be achieved faster (provided that the convergence is guaranteed).

3. Speeding up the average consensus

In order to minimize the consensus time, one can optimize the step size σ_{ij} for any link e_{ij} between the nodes i and j, i.e., optimizing the link weights. To this end, a number of methods have been proposed such as a convex optimization-based method [19], and the ones based on the betweenness centrality measures [5,20]. However, these methods require complete knowledge of the topological properties of the networks, which cannot be attained for many cases. We consider the following effective algorithms that need only local information of the networks.

Algo1. *Uniform step size as* $\sigma = 1/k_{\max}$. As we mentioned, with this choice the average consensus is guaranteed in finite steps. However, since computing $k_{\max}$ needs global information on the network topology, this might not be applicable in some applications.

Algo2. *Uniform step size as* $\sigma = 2/(\lambda_2 + \lambda_N)$. λ_2 and λ_N are the second smallest and the largest eigenvalues of L, respectively. This choice for the uniform step size has been shown to be sub-optimal for consensus time [17]; however, it needs computation of the eigenvalues, which indeed requires global information on the network.

Algo3. *Edge-specific step size as* $\sigma_{ij} = 1/\max(k_i, k_j)$. k_i is the degree of node i. This algorithm – known as Metropolis–Hasting algorithm – has been shown to have guaranteed convergence and have been frequently used in real networks [17]. The advantage of this algorithm over Algo1 and Algo2 is that it needs local information (degree) of the adjacent nodes.

Algo4. *Edge-specific step size as* $\sigma_{ij} = 1/k_i k_j$. It has been shown that the link weights in many real-world networks have high correlation with the product of the degree of its end nodes [21]. Therefore, this quantity can be an estimate of the load of the links.

Algo5. *Edge-specific step size as* $\sigma_{ij} = 1/k_i$. This way the weighted connection graph becomes directed; σ_{ij} is the weight of the communication from the node j to the node i. This is the simplest one among these five algorithms, since when node i receives a

signal $\mathbf{x}_j$ from node j, j does not need to transmit its degree to i in order to let i compute the update for its quantity $\mathbf{x}_i$ (eq. (1)). The only parameter required for each node to update its quantity is its own degree. With this choice for the step size, the convergence is guaranteed.

Lemma. Considering the system expressed by eq. (1) or (2) and with the choice of the step sizes as Algo5, the consensus is guaranteed in finite iterations.

Proof. In order to prove the convergence, one has to show that the Perron matrix is primitive with all eigenvalues inside the unit circle [4]. With this choice of step sizes, the Perron matrix can be written as $P = I - D^{-1}L$. Since the connection graph is connected, P is irreducible. Also, all its raw-sums are equal to zero. Therefore, P is a primitive matrix. It is easy to show that the eigenvalues of P are $1 - \delta_i$, where δ_i is the ith eigenvalue of $D^{-1}L$. The diagonal elements of $D^{-1}L$ are 1. Although $D^{-1}L$ is a non-symmetric matrix, due to $\det(D^{-1}L - \lambda \mathrm{I}) = \det(D^{-1/2}LD^{-1/2} - \lambda \mathrm{I})$ for any λ ('det' denotes determinant), the eigenvalues of $D^{-1}L$ are equal to those of $D^{-1/2}LD^{-1/2}$, i.e. real and non-negative with smallest eigenvalue as $\delta_1 = 0$. For this case, Gerschgorin circle theorem [22] guarantees that $0 < \delta_2 \leq \cdots \leq \delta_N < 2$. Thus, all the eigenvalues of P are inside the unit circle, and the proof is complete.

Considering that $x(k) = P^k x(0)$, an average consensus is obtained if the limit $\lim_{k\to\infty} P^k$ exists. Due to Perron–Frobenius theorem, if P is a primitive non-negative matrix with right and left eigenvectors v and w, respectively, satisfying, $w^T P = w^T$, $Pv = v$ and $v^T w = 1$, then we have, $\lim_{k\to\infty} P^k = vw^T$ [23]. Since P is primitive and non-negative, the limit exists and average consensus is achieved.

4. Simulation results

In this section, we compared the performance of the above algorithms through numerical simulations on random geographical networks that are realistic models for sensor networks. The random geographical networks were constructed as follows [5,24]: A number of nodes were randomly distributed in a two-dimensional field with normalized length and width of 1. The coordinates of the nodes were picked up from a uniform distribution in the range [0,1]. Each sensor can communicate with those in its neighbourhood defined by a circle with radius R centred at the coordinates of the sensor. Furthermore, even though two nodes are in the communication range, the communication between them might fail with probability P_f.

We numerically computed the consensus time T. One can determine T by directly monitoring the average consensus error and computing the time necessary for making the error small enough. When consensus is achieved in a network, the error converges distinctly to zero. Thus, by putting a threshold on the consensus error and some proper stopping conditions, one can obtain the consensus time. Consider the dynamical network (1). The average consensus error of the network at step k is defined as

$$E(k) = \frac{2}{N(N-1)} \sum_{i<j} \|\mathbf{x}_i(k) - \mathbf{x}_j(k)\|^2. \tag{4}$$

We chose randomly an initial state of the network with unit average error, which can be done by normalizing the average error to the initial conditions, i.e. dividing the average error by the error at $k = 0$. In this case, when $E(0) = 1$, the error will be insensitive to the initial conditions. Then, through numerical simulations we determined the time the network needs until the average error reaches a threshold ε, $\varepsilon = 10^{-4}$ in this work, and stays below this value thereafter. Indeed, the time T, where $E(T) = \varepsilon$ and $E(k) < \varepsilon$ for $k > T$, is interpreted as the consensus time T for the dynamical network. If the network includes disconnected components, general consensus (or synchronization) cannot be attained, regardless of how much strong the coupling is. Therefore, it is not meaningful to talk about a single consensus time in such networks. In disconnected networks, each connected component will have a specific consensus time.

We performed numerical simulation on networks with different structural properties. The sensors were located on a 2D grid with unit dimensions. Figures 1–3 show the consensus time T, i.e. iteration steps, as a function of the network size N, the communication failure probability P_f and the communication radius R, respectively. Whenever necessary, we fixed the parameters as $N = 1600$, $R = 0.15$ and $P_f = 0.4$. The parameters were considered in such a way that the networks are connected; otherwise, a general consensus cannot be obtained regardless of how strong the coupling is.

As can be seen, although Algo5 is the simplest communication protocol among these five algorithms, it has the best performance in terms of consensus time. For example, in networks with $N = 3600$ nodes, the consensus time of Algo5 is half of Algo1 and 30% less than Algo2 (figure 1). Interestingly, its performance is also better than Algo3, which gives the optimal uniform step size through computationally expensive spectral analysis. Note that Algo5 (as well as Algo3 and Algo4) gives non-uniform, i.e. edge-specific, step sizes.

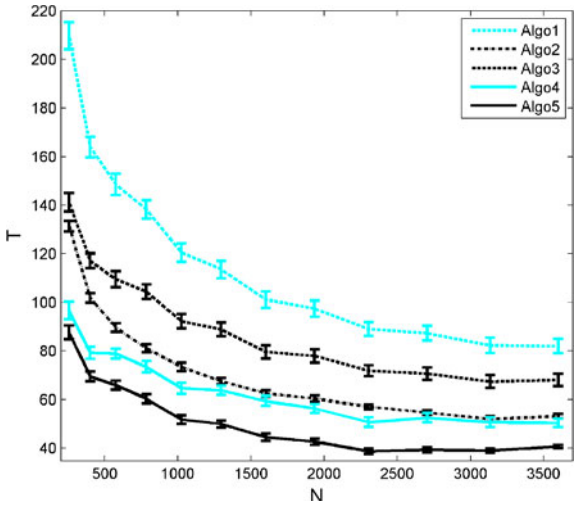

Figure 1. Consensus time T (i.e. iteration steps) as a function of network size N ($P_f = 0.4$ and $R = 0.15$) for different consensus algorithms (Algo1–Algo5). The graphs show the mean values with standard errors over 50 realizations.

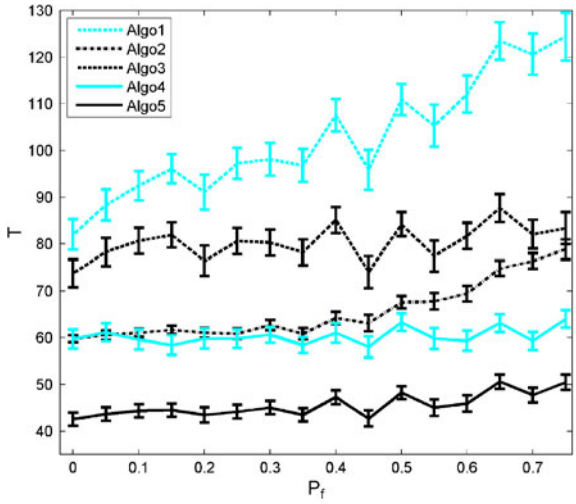

Figure 2. T as a function of communication failure probability P_f ($N = 1600$ and $R = 0.15$). The graphs show the mean values with standard errors over 50 realizations.

For fixed communication range and failure probability ($R = 0.15$ and $P_f = 0.4$), as the number of nodes increases, the consensus time exponentially decreases (figure 1). This is mainly due to the increase of the density of the sensor in the grid. As failure probability increases, the network become sparser, and thus, it takes longer time for the nodes to achieve consensus (figure 2). However, Algo5 showed the least increase in the consensus time. The network become denser as the communication range R increases, and not surprisingly, the consensus time T decreases (figure 3). T shows an exponential decrease

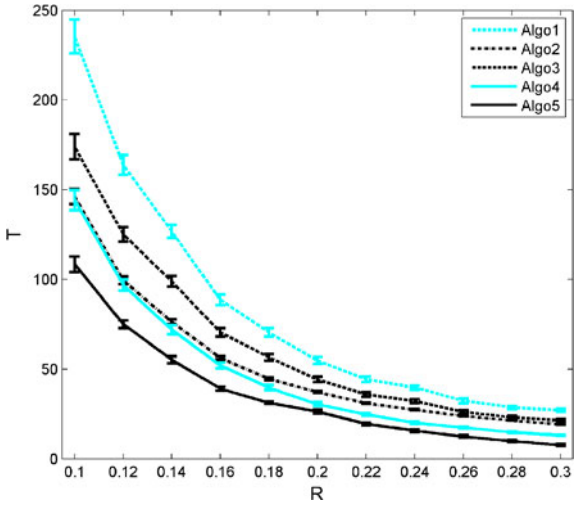

Figure 3. T as a function of communication radius R ($N = 1600$ and $P_f = 0.4$). The graphs show the mean values with standard errors over 50 realizations.

by increasing R for all algorithms. However, Algo5 showed the best performance among them.

5. Conclusions

In this paper, we proposed a simple consensus algorithm with guaranteed convergence for random geographical networks – realistic models for wireless sensor networks. Through numerical simulations on random geographical networks, we showed that this algorithm – which only needs the degree of receiving nodes for implementation and is the simplest one among a number of linear algorithms – has the best performance, i.e. less consensus time. This is important since the energy consumption is one of the major challenges in the design of sensor networks. Faster consensus time can save energy in the network, reduce the costs and increase the life of the sensors.

References

[1] I F Akyildiz *et al*, *Computer Networks* **38**, 393 (2002)
[2] A Milenkovic, C Ottoa and E Jovanova, *Comput. Commun.* **29**, 2521 (2006)
[3] P Baronti *et al*, *Comput. Commun.* **30**, 1655 (2007)
[4] R Olfati-Saber, J A Fax and R M Murray, *Proceedings of the IEEE* **95**, 215 (2007)
[5] M Jalili and A Mazloomian, *Phys. Lett.* **A374**, 3920 (2010)
[6] S Barbarossa and G Scutari, *IEEE Signal Processing Magazine* **24**, 26 (2007)
[7] G Scutari, S Barbarossa and L Pescosolido, *IEEE Transactions on Signal Processing* **56**, 1667 (2008)
[8] M Barahona and L M Pecora, *Phys. Rev. Lett.* **89**, 054101 (2002)
[9] L M Pecora and T L Carroll, *Phys. Rev. Lett.* **80**, 2109 (1998)
[10] A Ajdari Rad, M Jalili and M Hasler, *Chaos* **18**, 037104 (2008)
[11] M Jalili, A Ajdari Rad and M Hasler, *Phys. Rev.* **E78**, 016105 (2008)
[12] L Xiao, S Boyd and S-J Kim, *J. Parallel and Distributed Computing* **67**, 33 (2007)
[13] B Kozma and A Barrat, *Phys. Rev.* **E77**, 016102 (2008)
[14] P-A Bliman and G Ferrari-Trecate, *Automatica* **44**, 1985 (2008)
[15] J Lu, D W C Ho and J Kurths, *Phys. Rev.* **E80**, 066121 (2009)
[16] S Boyd, P Diaconis and L Xiao, *SIAM Rev.* **46**, 667 (2004)
[17] L Xiao and S Boyd, *Systems & Control Lett.* **53**, 65 (2004)
[18] D Xu, Y Li and T Wu, *Phys. A: Stat. Mech. Appl.* **382**, 722 (2007)
[19] S Boyd, *Proceedings of the International Congress of Mathematicians* **3**, 1311 (2006)
[20] M Jalili, A Ajdari Rad and M Hasler, *IEEE International Symposium on Circuits and Systems* (2008) p. 2522
[21] A Barrat *et al*, *Proceedings of the National Academy of Science of the United States of America* **101**, 3747 (2004)
[22] S A Gerschgorin, *Izv. Akad. Nauk. USSR Otd. Fiz.-Mat. Nauk* **7**, 749 (1931)
[23] R A Horn and C R Johnson, *Matrix analysis* (Cambridge University Press, Cambridge, UK, 1987)
[24] A Khadivi and M Hasler, *International Conference on Sensor Networks Applications, Experimentation and Logistics* (Athens, Greece, 2009) p. 16