Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2011, Article ID 428290, 11 pages
doi:10.1155/2011/428290

Research Article

Performance Evaluation of Some Distributed Averaging Algorithms for Sensor Networks

Marco Baldi,[1] Elma Zanaj,[2] and Franco Chiaraluce[1]

[1] *Università Politecnica delle Marche, Via Brecce Bianche, 60131 Ancona, Italy*
[2] *Universiteti Politeknik i Tiranes, Fakulteti i Teknologjise se Informacionit, Tirana, Albania*

Correspondence should be addressed to Marco Baldi, m.baldi@univpm.it

Received 1 April 2011; Revised 28 June 2011; Accepted 29 June 2011

Though capillary sensor networks have the advantage of reporting punctual estimations of their sensed quantity, it is often useful for the nodes to know the overall average value of the same quantity. This is required, for example, when the network can make autonomous decisions. Several algorithms exist for solving the averaging problem in a distributed manner. Their efficiency can be measured by the number of iterations needed to converge to the average sensed value. In this paper, we consider two point-to-point and one point-to-multipoint distributed averaging algorithms that can be seen as variants of the same averaging solution. We define a set of analytical tools to evaluate the performance of these algorithms and to optimize their parameters in such a way to accelerate convergence. We also provide a performance assessment, based on numerical simulations, aimed at verifying the results of the analytical treatment and at comparing the considered schemes.

1. Introduction

A common requirement for distributed sensor networks is to adopt very simple and cheap devices, possibly powered by batteries that should last months or even years. So, a valuable task in this context is to find energy efficient solutions and protocols to accomplish all network functions [1]. In such a scenario, a prominent role is played by the need to communicate among nodes in the network. For this purpose, a variety of protocols and transmission techniques exist, all aimed at limiting the nodes activity and, hence, the power consumption.

Existing internode communication protocols can be characterized in a number of different ways. For the purposes of the present paper, we distinguish between *gossip protocols* and *broadcast protocols*. In both cases, each node of the network initially has an atomic piece of information, and the target of the protocol is to distribute information on the overall network data to all its component nodes. By a gossip algorithm we mean an algorithm in which each node communicates with no more than one neighbor in each time slot. So, it realizes, locally, a point-to-point communication. Dually, by a broadcast algorithm, we mean an algorithm in which each node communicates with any neighbor (i.e., within its coverage radius) in each time slot. So, it realizes, locally, a point-to-multipoint communication. It must be noted that our definition of broadcasting is slightly different from that sometimes adopted in the previous literature (see, e.g., [2, 3]) where broadcasting is adopted to distribute only one piece of information, initially located at one node, to all other nodes. According to our definition, we will see, in Section 3, that broadcasting can also be seen, and analyzed, as an extension of gossiping.

Gossip algorithms permit to compute sums, averages, random samples, quantiles, and other aggregate functions, starting from an ensemble of initial sensed values [4–7]. Distributed averaging, in particular, has gained a prominent role, as it serves as an archetypal instance of distributed signal processing, in which the goal is to achieve a global objective, based on purely local computations. Apparently, distributed averaging is a very specialized problem, that is, for realizing specific tasks of interest: as an example, we can consider the need to compute the average value measured by all temperature sensors in a given area. Actually, and much more generally, it provides a useful building block for solving more complex issues. In fact, any averaging algorithm can easily be converted into a general algorithm that computes any linear projection of the sensor measurements, assuming that each sensor knows the corresponding coefficient of the projection

vector. Such algorithms have been proposed for several problems of distributed computation in sensor networks, including distributed filtering, detection, optimization, and compression; examples can be found in [8–10].

In this paper, we focus on the averaging problem. We assume that the network is formed by N sensors and that the target is to find, in the shortest possible time, an estimate of the average of their sensed values x_i, that is, $x_{\text{ave}} = \sum_{i=1}^{N} x_i/N$. It is well known that the average provides the minimum mean squared error (MMSE) estimate of the sensed quantity. More complex solutions are most suited for other contexts, including widespread communication networks, like Internet [11], or decentralized reputation systems [12]. It has been recently shown that gossip algorithms can also be effectively applied when, in a sensor network, there is a group of nodes that must acquire a function of data stored by another group of nodes [13]. A further interesting application field is gossip-based peer-to-peer video streaming [14].

In this paper, we consider unidirectional and bidirectional gossip protocols, and we investigate their efficiency for averaging purposes. The distinctive feature of the considered protocols is the possibility to see them in a unitary way (i.e., as variants of a common framework), and such property facilitates the comparison among them. The approach we adopt permits us to investigate whether margins exist for optimizing performance. This is done through theoretical arguments and numerical simulations. In particular, we extend the well-known approach based on the potential function to estimate the optimum values of some parameters. For the bidirectional algorithm, in particular, this will yield to change some common choices (i.e., the perfect symmetry of the interaction) that have been often adopted in the previous literature.

As we deal with averaging algorithms, the convergence speed to the actual mean value is considered as the main parameter for performance assessment. A relevant result of the paper is to show that the unidirectional gossip algorithm can have the same performance as the bidirectional one in the case of fully meshed networks. On the other hand, in the case of nonfully meshed networks, the bidirectional algorithm is generally faster. A broadcast averaging algorithm is also considered, and its updating rule is discussed as a special instance of the considered framework.

A first output of this paper is an ensemble of numerical results on this subject that, instead, is often faced in conceptual, but scarcely quantitative, terms. In fact, the literature lacks curves of convergence time for variable operation conditions. Our study allows to make a performance comparison among different communication strategies, that is of practical importance. The analysis of the impact of the share factors is another novelty of this paper. Combined with the quantitative analysis, it permits the designer to decide if the extra cost implied by the computation is a price that is worth paying for the performance advantages it can offer.

The remainder of the paper is organized as follows. In Section 2, we introduce the notation and the relevant parameters for performance evaluation and comparison. In Section 3, the considered algorithms are described, first in general terms and then focusing on some specific implementations. In Section 4, we recall the definition of potential function, and we use it for optimizing the share factor in the case of fully meshed and nonfully meshed networks. In Section 5, we present several simulated curves. Finally, Section 6 concludes the paper.

2. Notation

A distributed sensor network can be modeled through a connected graph $G(V, E)$, where V is the vertex set and E is the edge set. Two nodes having an edge between them are called neighbors. We consider an asynchronous time model, where each node has a clock which ticks at the times of a rate 1 Poisson process. Therefore, the intertick times at each node are rate 1 exponentials, independent across nodes, and over time. Equivalently, this corresponds to a single clock ticking according to a rate N Poisson process at times Z_k, $k \geq 1$. Time is discretized according to clock ticks, since these are the only times at which the measured values change. Therefore, the interval $[Z_k;\ Z_{k+1})$ denotes the kth time slot, and, on average, there are N clock ticks per unit of absolute time.

In a gossip algorithm, we assume that the ith node ($i = 1, \ldots, N$), when its clock ticks, contacts one of its neighbors, say j, by choosing it according to a selection probability P_{ij}. At each clock tick, node i tries to communicate, and its attempt has always success in the absence of link failures. If we define matrix $\mathbf{P} = [P_{ij}]$, it results in an $N \times N$ matrix of nonnegative entries with the condition that $P_{ij} \neq 0$ only if $(i, j) \in E$ and $i \neq j$. $\mathbf{P}$ is a stochastic matrix, so its largest eigenvalue is equal to 1, while the remaining $N - 1$ eigenvalues have magnitude strictly smaller than 1 [15]. We denote by $\mathbf{x}(0) = [x_1(0), x_2(0), \ldots, x_N(0)]^T$ the vector collecting the initial (sensed) values at all nodes (superscript T means transposition). After k ticks, the updated values are collected in the vector $\mathbf{x}(k) = [x_1(k), x_2(k), \ldots, x_N(k)]^T$. The averaging algorithms differ in the updating strategy adopted. Some examples, for the class of protocols considered in this paper, will be given in Section 3.

The performance of gossip algorithms can be measured in terms of convergence to the average [5, 16–18]. This subject has been widely investigated in the previous literature (see [16, 19–23] and the references therein), mostly under the theoretical viewpoint and by adopting abstract system models. In [16], for example, an analytical framework for the design and study of a randomized distributed averaging problem was presented, together with specific tools for network optimization. Optimizing the averaging gossip algorithm obviously coincides with minimizing the time taken for the value at each node to become sufficiently close to the average value independently of the initial condition. Such time interval is called *averaging time* and will be denoted by T_{ave} in the following.

The proximity to the average value can be measured through the following error definition:

$$e(k) = \frac{\|\mathbf{x}(k) - x_{\text{ave}}\mathbf{1}\|}{\|\mathbf{x}(0)\|}, \tag{1}$$

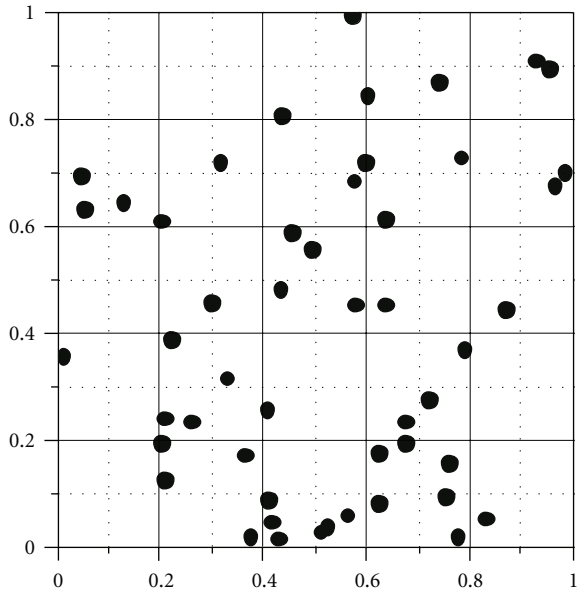

Figure 1: Example of a randomly distributed sensor network with $N = 50$ nodes.

where $\|\mathbf{v}\|$ denotes the l^2-norm of vector $\mathbf{v}$ and $\mathbf{1}$ is an $N \times 1$ vector with all components equal to 1. Because of the stochastic nature of the problem, $e(k)$ is a random variable whose value at time k can be estimated through R experiments. Each experiment (or simulation) produces a curve $e^q(k)$, $q = 1, \ldots, R$, and an estimation of the error evolution is given by the average curve after R realizations, that is,

$$\langle e(k) \rangle = \frac{1}{R} \sum_{q=1}^{R} e^q(k). \tag{2}$$

According to the probability theory, $\lim_{R \to \infty} \langle e(k) \rangle = \widehat{e(k)}$, $\forall k$, where $\widehat{e(k)}$ represents the true mean of $e(k)$.

As regards the topology, we focus attention on the so-called random geometric graph (RGG), but this is just an example, in the sense that the analysis could be repeated for different structures, like the ring one, or others. An example of RGG, with $N = 50$ nodes, is shown in Figure 1. The node positions have been randomly generated within a unit square according to a 2D homogeneous Poisson point process. The position of the nodes is one of the random variables of the problem. Further randomness derives from the selection probabilities P_{ij} and the statistics of the sensed quantity. Wishing to extract information of (sufficiently) general validity, for a given network size, all these variables should be explored and simulated.

3. The Considered Algorithms

The class of averaging algorithms considered in this paper can be described as follows.

The ith network node ($i = 1, \ldots, N$) can store two values named s_i and w_i, respectively. These values are updated at time k. The initial values are set as $s_i(0) = x_i(0)$ and $w_i(0) = 1$, for any i. Let us consider the most general case of a bidirectional algorithm. If the clock of the ith node ticks at the kth time instant, the node randomly selects one of its neighbors, say node j, and sends to it a fraction $(1 - \alpha_i)$ of its stored values, while it retains the remaining fraction α_i. Symmetrically, after having being contacted, node j sends to node i a fraction $(1 - \alpha_j)$ of its stored values, while it retains the remaining fraction α_j. Following this "double interaction", which implies, in general, two transmissions, the quantities at the ith and jth nodes are updated as follows:

$$\begin{aligned} s_i(k) &= \alpha_i s_i(k-1) + \left(1 - \alpha_j\right) s_j(k-1), \\ w_i(k) &= \alpha_i w_i(k-1) + \left(1 - \alpha_j\right) w_j(k-1), \\ s_j(k) &= \alpha_j s_j(k-1) + (1 - \alpha_i) s_i(k-1), \\ w_j(k) &= \alpha_j w_j(k-1) + (1 - \alpha_i) w_i(k-1), \end{aligned} \tag{3}$$

while the values at all the other nodes remain unchanged.

The following relationships must be verified for any k:

$$\begin{aligned} \sum_{i=1}^{N} s_i(k) &= \sum_{i=1}^{N} x_i(0) = N x_{\text{ave}}, \\ \sum_{i=1}^{N} w_i(k) &= N, \end{aligned} \tag{4}$$

that are known as "mass conservation properties". After the interaction between nodes i and j, a new estimate of the sensed quantity is acquired, by both nodes, as $x_m(k) = s_m(k)/w_m(k)$, with $m = i, j$.

Starting from the rule (3), a number of different options, bidirectional, unidirectional, or even broadcast, can be implemented and will be considered next. They are characterized by different complexity in terms of required storing and processing capacities. We expect that these differences also affect performance, that will be evaluated and discussed in the following, with special emphasis on the averaging time.

3.1. Option 1: Bidirectional Gossip (BG). A first instance of averaging algorithm can be obtained by directly implementing (3). We denote such option as bidirectional gossip (BG) algorithm. In general, in (3), the share factor α_i can be different for any node (in [24], it was even proposed that it changes with time). As the share factors represent degrees of freedom to optimize, it is intuitively evident that such choice can represent the most favorable case. As a counterpart, however, the multivariable scenario it produces is difficult to manage, both in theory and in practice. So, in the following analysis, we will focus on the choice $\alpha_i = \alpha$, $\forall i$, that simplifies the approach and translates the optimization problem into that of finding the optimum value for α. This issue will be faced in Section 4.

The assumption of identical share factors for all the nodes makes the role of w_i uninfluential, because its value remains identical to 1 (the initial condition). As a consequence, for

a fixed α, the updating rules (3) reduce to the first and third equations that become

$$\begin{aligned} s_i(k) &= \alpha s_i(k-1) + (1-\alpha)s_j(k-1), \\ s_j(k) &= \alpha s_j(k-1) + (1-\alpha)s_i(k-1). \end{aligned} \tag{5}$$

In the following, we will show that, as intuitively reasonable, the best choice for the share factor is $\alpha = 1/2$ in the case of a fully meshed network. The assumption $\alpha = 1/2$ is rather common in the previous literature [16]. For nonfully meshed networks, however, this choice is no longer optimum, and smaller values of α should be preferred. This will also be shown in Section 4.

3.2. Option 2: Unidirectional Gossip (UG). A different way for simplifying the rules (3) consists of adopting a unidirectional algorithm, which means that at the kth clock tick transmission only occurs from node i to node j. This algorithm is also known as *push sum* [24], and it can be obtained from (3) by setting $\alpha_j = 1$. If we consider $\alpha_i = \alpha, \ \forall i$, as for the BG algorithm, the updating rule becomes

$$\begin{aligned} s_i(k) &= \alpha s_i(k-1), \\ w_i(k) &= \alpha w_i(k-1), \\ s_j(k) &= s_j(k-1) + (1-\alpha)s_i(k-1), \\ w_j(k) &= w_j(k-1) + (1-\alpha)w_i(k-1). \end{aligned} \tag{6}$$

Like for BG, the value of α should be optimized; this issue will be faced, through theoretical arguments, in Section 4 and, through simulations, in Section 5.

3.3. Option 3: Broadcast (B). If we focus on distributed wireless sensor networks, we should consider that, when a node transmits some information, all its neighbors are able to receive it, this way implementing a broadcast mechanism [25]. Equations (3) can also be used to describe the B algorithm (that is, obviously, unidirectional, so that $\alpha_j = 1$), but, in this case, the fraction of s_i and w_i that node i transmits is distributed among all the receiving nodes. A possible solution (once again, the simplest to implement) consists of choosing a uniform sharing so that, for example, in the case of a fully meshed network, (3) become

$$\begin{aligned} s_i(k) &= \alpha s_i(k-1), \\ w_i(k) &= \alpha w_i(k-1), \\ s_j(k) &= s_j(k-1) + \frac{1-\alpha}{N-1} s_i(k-1), \\ w_j(k) &= w_j(k-1) + \frac{1-\alpha}{N-1} w_i(k-1). \end{aligned} \tag{7}$$

We have numerically verified that the broadcast algorithm is the most efficient one, among those considered, in terms of convergence time. For this reason, in Section 5, we will use the B algorithm as a benchmark for measuring the penalty encountered by the other solutions.

On the other hand, the B algorithm requires the adoption of omnidirectional antennas, while directional antennas, that can help reducing the power consumption, can be used only for point-to-point communications, although they must be redirected at any transmission, depending on the selected neighbor.

4. Share Factor Optimization

As outlined in the previous section, an important issue concerns optimization of the share factor α. A simple analytical approach for such optimization problem is based on the concept of "potential function".

Given a set of values $\mathbf{v} = [v_1, v_2, \ldots, v_N]^T$, where v_i is the value at node i, the potential Φ of the graph, with respect to $\mathbf{v}$, can be defined as

$$\Phi = \|\mathbf{v} - v_{\text{ave}}\mathbf{1}\|^2 = \sum_{i \in V} (v_i - v_{\text{ave}})^2, \tag{8}$$

where v_{ave} is the average value over the whole network. Evidently, Φ is a measure of the variance of the value distribution. In particular, note that $\Phi = 0$ if and only if $\mathbf{v} = v_{\text{ave}}\mathbf{1}$. In the following, we will denote by $\Phi(k)$ the potential function after the kth clock tick ($\Phi(0)$ is the initial value). Definition (8) could be applied, in principle, directly to the vector of the estimates $\mathbf{x}(k)$, this way obtaining, apart from the normalization by $\|\mathbf{x}(0)\|$, the square of the error $e(k)$. For the considered algorithms, however, a slightly more complex expression is more favorable, and it is described next.

Let us consider a vector $\mathbf{v}_i(k)$ (that does not contain any measured quantity but is only introduced for analysis purposes), whose components, $v_{ij}(k)$, are such that

$$s_i(k) = \sum_{j=1}^{N} v_{ij}(k) x_j(0). \tag{9}$$

The following condition is satisfied:

$$w_i(k) = \sum_{j=1}^{N} v_{ij}(k). \tag{10}$$

It is clear that, if $\mathbf{v}_i(k)$ is nearly proportional to the all-one vector, then $x_i(k) = s_i(k)/w_i(k)$ is close to the true average. The potential function is then defined as follows [24]:

$$\Phi(k) = \sum_{i=1}^{N} \sum_{j=1}^{N} \left[v_{ij}(k) - \frac{w_i(k)}{N} \right]^2. \tag{11}$$

So, in the limit case of all nodes perfectly aware of the true average, the potential function is null. Based on this evidence, evaluation of the mean potential function, for any k, should permit to estimate the convergence speed of the algorithm.

In Section 4.1, we determine the mean potential function for the three options described in Section 3 by considering the case of a fully meshed network. In Section 4.2, the analysis is repeated for the case of nonfully meshed networks though limited, for the sake of brevity, to the UG algorithm.

Focusing attention on the most general case of nonfully meshed networks (that, obviously, includes the fully meshed as a particular option), the theoretical analysis refers to the hypothesis that each node has the same probability of contacting each one of its N_i neighbors within its coverage radius r_i, that is

$$P_{ij} = \begin{cases} \dfrac{1}{N_i} & d_{ij} \le r_i, \\ 0 & d_{ij} > r_i, \end{cases} \tag{12}$$

where d_{ij} is the Euclidean distance between nodes i and j. The same assumption will be adopted in Section 5, where several simulation results will be presented and discussed.

We also observe that, depending on the values of r_i ($i = 1,\ldots,N$), some nodes could be unreachable, since they are disjointed from the rest of the network. This situation should be avoided (and, in fact, it is not considered in this paper), as, in general, it does not allow convergence to the true average value. If the starting procedure includes a localization phase, which means that each node knows its own geographic location and can learn those of its one-hop neighbors, suitable routing rules could be adopted [26], for example, depending on the value of the Euclidean distance between the nodes.

Localization can contribute to improve the convergence speed but at the expense of an increase in complexity. In [6], for example, a geographic gossip algorithm has been proposed, based on greedy routing, that is potentially able to provide remarkable gains. But the applicability of this kind of protocols, where each node must compute and compare a large number of distances from a prefixed target, seems difficult. For this reason, we have not included these gossip versions in our study.

4.1. Optimization for Fully Meshed Networks. Let us focus attention on the UG algorithm. Assuming that at the time instant k, node l is selected as the transmitter and node m as the receiver, the following difference between the potential functions at instants $k-1$ and k can easily be derived:

$$\begin{aligned} \delta\Phi &= \Phi(k-1) - \Phi(k) \\ &= 2\alpha(1-\alpha)\sum_{j=1}^{N}\left[v_{lj}(k-1) - \frac{w_l(k-1)}{N}\right]^2 \\ &\quad - 2(1-\alpha)\sum_{j=1}^{N}\left[v_{lj}(k-1) - \frac{w_l(k-1)}{N}\right] \\ &\quad \cdot \left[v_{mj}(k-1) - \frac{w_m(k-1)}{N}\right]. \end{aligned} \tag{13}$$

For simplifying the notation, in the following, we will omit to indicate that the quantities at the right side are computed at $k-1$. We wish to compute the average of (13) over all possible choices of the transmitting and receiving nodes. Taking into account that both the choices of transmitter and receiver are made following a uniform law, and using definition (11), we find

$$\begin{aligned} \langle\delta\Phi\rangle &= \frac{2\alpha(1-\alpha)}{N}\Phi + \frac{2(1-\alpha)}{N(N-1)}\Phi \\ &\quad - \frac{2(1-\alpha)}{(N-1)N}\sum_{j=1}^{N}\sum_{l=1}^{N}\sum_{m\in C_l}\left(v_{lj} - \frac{w_l}{N}\right)\left(v_{mj} - \frac{w_m}{N}\right), \end{aligned} \tag{14}$$

where $\Phi = \Phi(k-1)$, and C_l is the subset of nodes that includes node l and the nodes it is linked to. In the case of a fully meshed network, however, C_l is the entire network, and it is easy to verify that the last term at the right side of (14) is null because of the mass conservation properties (4). This yields the simple expression for $\langle\delta\Phi\rangle/\Phi$ that is reported in Table 1. The last column of the table also shows the optimum value of α, that is, the one which maximizes $\langle\delta\Phi\rangle/\Phi$. A similar analysis can be developed for the other algorithms and results in the expressions of $\langle\delta\Phi\rangle/\Phi$ and the value of α_{opt} also reported in Table 1. Additionally, for any algorithm, $\alpha < 1$ implies $\langle\delta\Phi\rangle/\Phi > 0$, and this result can be used to demonstrate convergence.

Table 1: Expressions of $\langle\delta\Phi\rangle/\Phi$ for the considered algorithms, in the case of fully meshed networks, and corresponding optimum α.

Algorithm	$\langle\delta\Phi\rangle/\Phi$	α_{opt}
UG	$\dfrac{2(1-\alpha)}{N}\left(\alpha + \dfrac{1}{N-1}\right)$	$\dfrac{N-2}{2(N-1)}$
BG	$\dfrac{4\alpha(1-\alpha)}{N}$	$\dfrac{1}{2}$
B	$\dfrac{1-\alpha^2}{N-1}$	0

Maximizing $\langle\delta\Phi\rangle/\Phi$ seems a good criterion for optimizing the degree of freedom offered by the share factor. The reliability of the analytical results obtained will be tested in Section 5, where a number of simulation outputs will be presented and discussed.

By inserting α_{opt} in $\langle\delta\Phi\rangle/\Phi$, we see that $\langle\delta\Phi\rangle/\Phi|_{\text{UG}} \approx 1/2N$, while $\langle\delta\Phi\rangle/\Phi|_{\text{BG}} = 1/N \approx \langle\delta\Phi\rangle/\Phi|_{\text{B}}$. This gives preliminary indications about the convergence speed of the various algorithms. Apparently, the BG algorithm reduces the potential function faster than the UG algorithm. Actually, we must consider that the former implies two transmissions at each clock tick while the latter only performs one transmission. If comparison is done on the basis of the number of transmissions (as it is more correct, for the sake of fairness), the expected behavior is then equivalent. Similarly, the $\langle\delta\Phi\rangle/\Phi$ for the B algorithm decreases with the same speed as in the BG algorithm; however, the latter requires two transmissions at each clock tick instead of one. We will show in Section 5 that the advantage in terms of $\langle e(k)\rangle$ is even more remarkable.

4.2. Optimization for Nonfully Meshed Networks. The assumption of a fully meshed network is unrealistic in most operation environments. As a matter of fact, network

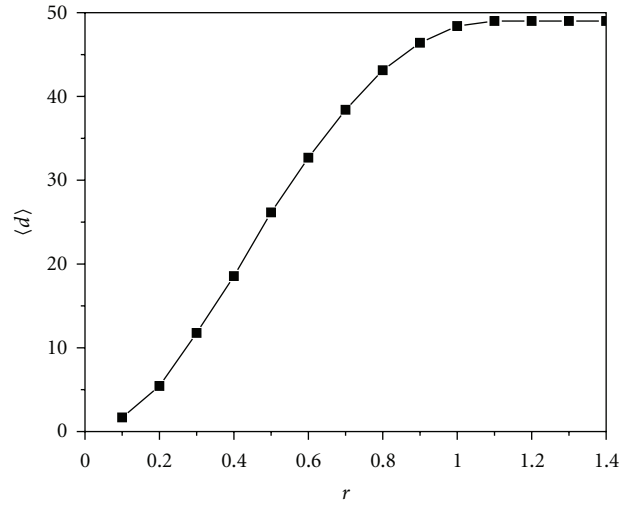

Figure 2: Average value of d as a function of r, computed over 100 RGG realizations with $N = 50$.

connectivity can be reduced by (i) limited coverage or (ii) link failures. Depending on its power availability (but also on constraints on the transmission task), the ith node can reach only the nodes that are within a coverage radius r_i. Correspondingly, its nodal degree d_i, defined as the number of neighbors at distance not greater than r_i, is also limited (we observe that the nodal degree coincides with the number of reachable neighbors, N_i, used in (12)).

The nodal degree has an important role in establishing the averaging time for the case of nonfully meshed networks. For a random topology, like the RGG, even assuming that $r_i = r$, $\forall i$, the nodal degree is generally not unique. Moreover, for each value of r, the connectivity level can vary from graph to graph. So, an average nodal degree, $\langle d \rangle$, must be computed for our analysis purposes. Figure 2 reports the $\langle d \rangle$ values, as a function of r, that we have found by averaging over 100 network realizations randomly generated through a 2D homogeneous Poisson process. The impact of different r on the averaging time will be discussed in Section 5.

Link failures can appear because of a number of electronic or environmental troubles [27]. It is reasonable to assume they are randomly distributed, and, under this hypothesis, the approach based on the potential function can be extended to face analytically the problem of the share factor optimization under such more complex conditions.

Let us focus on the UG algorithm. In the case of nonfully meshed networks, (14) must be modified as follows:

$$\langle \delta\Phi \rangle = \frac{2\alpha(1-\alpha)}{N}\Phi + \frac{2(1-\alpha)}{N}\sum_{j=1}^{N}\sum_{l=1}^{N}\frac{1}{d_l}\left(v_{lj} - \frac{w_l}{N}\right)^2 - \frac{2(1-\alpha)}{N}\sum_{j=1}^{N}\sum_{l=1}^{N}\frac{1}{d_l}\sum_{m\in C_l}\left(v_{lj} - \frac{w_l}{N}\right)\left(v_{mj} - \frac{w_m}{N}\right). \tag{15}$$

In this expression, contrary to the fully meshed case, the last term is not zero, and it provides an additional contribution to $\langle \delta\Phi \rangle / \Phi$ that we must take into account.

For simplifying the analysis, we replace $1/d_l$ with its average value $\langle 1/d \rangle$. This is, obviously, an approximation, that yields

$$\langle \delta\Phi \rangle = \frac{2}{N}(1-\alpha)\left(\alpha + \left\langle \frac{1}{d} \right\rangle\right)\Phi - \frac{2(1-\alpha)}{N}\left\langle \frac{1}{d} \right\rangle \cdot \sum_{j=1}^{N}\sum_{l=1}^{N}\sum_{m\in C_l}\left(v_{lj} - \frac{w_l}{N}\right)\left(v_{mj} - \frac{w_m}{N}\right). \tag{16}$$

Further analytical elaboration is possible by exploiting the definition of Laplacian matrix [28]. In the most general case, the Laplacian matrix $\mathbf{Q}(G)$ of the graph $G(V, E)$ is an $N \times N$ matrix whose elements are defined as follows:

$$Q_{ij} = \begin{cases} d_i & \text{if } i = j, \\ -1 & \text{if } i \neq j,\ (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

The Laplacian matrix can also be written as $\mathbf{Q} = \mathbf{\Delta} - \mathbf{A}$, where $\mathbf{\Delta}$ is the diagonal matrix with elements $\Delta_{ii} = d_i$ and $\mathbf{A}$ is the adjacency matrix of the considered graph. The eigenvalues of $\mathbf{Q}$ are called the Laplacian eigenvalues. They are all real and nonnegative and satisfy the condition $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$. The second smallest eigenvalue, λ_2, is also known as the algebraic connectivity, and it is particularly important; λ_2 is equal to zero only if G is disconnected. Other properties of matrix $\mathbf{Q}$ and its eigenvalues can be found in the literature (see, e.g., [29]). Let $y_{ij} = v_{ij} - w_i/N$, $i = 1, \ldots, N$, be the components of a vector $\mathbf{y}_j$. Through simple algebra, using (11) and (17), (16) can be rewritten as follows:

$$\langle \delta\Phi \rangle = -\frac{2(1-\alpha)^2}{N}\Phi + \frac{2(1-\alpha)}{N}\left\langle \frac{1}{d} \right\rangle \sum_{j=1}^{N}\mathbf{y}_j^T\mathbf{Q}\mathbf{y}_j. \tag{18}$$

Let us define another vector $\mathbf{z} = (\mathbf{y}_1^T, \mathbf{y}_2^T, \ldots, \mathbf{y}_N^T)^T$ having N^2 components; it is evident that $\mathbf{z}^T\mathbf{z} = \Phi$. Moreover, let us consider a block matrix $\mathbf{L}$, with size $N^2 \times N^2$, having N repetitions of $\mathbf{Q}$ along the main diagonal and all the other blocks equal to the null matrix. Also $\mathbf{L}$ can be interpreted as a Laplacian matrix, whose eigenvalues coincide with those of $\mathbf{Q}$, but each appears with multiplicity N. Using these further definitions, through simple algebra, (18) can be rewritten as

$$\langle \delta\Phi \rangle = \frac{2(1-\alpha)}{N}\left[-(1-\alpha) + \left\langle \frac{1}{d} \right\rangle \frac{\mathbf{z}^T\mathbf{L}\mathbf{z}}{\mathbf{z}^T\mathbf{z}}\right]\Phi = \frac{2(1-\alpha)}{N}\left[-(1-\alpha) + \left\langle \frac{1}{d} \right\rangle \mathrm{RQ}\right]\Phi, \tag{19}$$

having denoted by $\mathrm{RQ} = \mathbf{z}^T\mathbf{L}\mathbf{z}/\mathbf{z}^T\mathbf{z}$ the so-called Rayleigh quotient. It follows from the Courant-Fischer minimax theorem [30] that $\lambda_2 \leq \mathrm{RQ} \leq \lambda_N$. This implies that

$$\frac{2(1-\alpha)}{N}\left[-(1-\alpha) + \left\langle \frac{1}{d} \right\rangle \lambda_2\right] \leq \frac{\langle \delta\Phi \rangle}{\Phi} \leq \frac{2(1-\alpha)}{N}\left[-(1-\alpha) + \left\langle \frac{1}{d} \right\rangle \lambda_N\right]. \tag{20}$$

In practice, the above analysis permits us to find a lower bound (lb) and an upper bound (ub) for the mean variation of the potential function conditioned on a starting value Φ. Similar, though less explicit, bounds can be found in the literature (see, e.g., [31]).

If the network is fully meshed, we have lb = ub, as all eigenvalues, except $\lambda_1 = 0$, are coincident. For nonfully meshed networks, instead, lb becomes lower and lower when reducing the average nodal degree, while ub has an opposite behavior [32]. As, depending on the value of α, the lb can be smaller than zero, convergence of the algorithm cannot be established a priori simply by looking at the lower bound. On the other hand, looking at the upper bound could yield too optimistic conclusions (in regard to the convergence and its speed).

Using (19), the value of α that maximizes $\langle \delta\Phi \rangle / \Phi$ can be derived as

$$\alpha_{\text{opt}} = 1 - \left\langle \frac{1}{d} \right\rangle \frac{\text{RQ}}{2}. \tag{21}$$

So, a correct estimate of α_{opt} is conditioned on a proper evaluation of RQ. An in-depth analysis should weight the contributions of λ_2 and λ_N and, even more important, should take into account the role of the other eigenvalues, λ_i, with $i \in [3, N-1]$. With the goal to develop further the theoretical analysis, a first approximation can consist in considering the average of the eigenvalues, that is, approximating RQ as

$$\text{RQ} = \frac{\sum_{i=2}^{N} \lambda_i}{N-1}. \tag{22}$$

In spite of the approximations, the agreement between the analytical result (21), with RQ computed as in (22), and the optimum value determined through simulations is good. An example is shown in Figure 3, for the RGG with $N = 50$ of Figure 1.

5. Simulation Results

In this section, we present a number of simulation results for the algorithms discussed in the previous sections. This will permit us to validate the theoretical analysis, to provide examples of the actual averaging performance, and to compare the various protocols.

For each considered algorithm, we can obtain an estimate of the optimal share factor value, through what we define as an *experiment*, as follows. We fix an RGG with $N = 50$ nodes (like that reported in Figure 1), a value of coverage radius r, and a set of 200 randomly chosen vector pairs. Each of such vector pairs is formed by (i) a real vector of initial sensed values at the N nodes and (ii) an integer vector of at most 3000 node indexes, corresponding to the sequence of nodes selected for transmission within the first 3000 clock ticks. Then, we choose a set of values of α (with a suitable granularity) and, for each of them, we calculate 200 $e(k)$ curves, each corresponding to one of the 200 pairs of initial conditions and sequences of transmitting nodes. So, for each value of α, we can obtain the curve $\langle e(k) \rangle$, averaged over 200 realizations. By comparing the $\langle e(k) \rangle$ curves, each experiment finally results in an estimate of the optimal value of α.

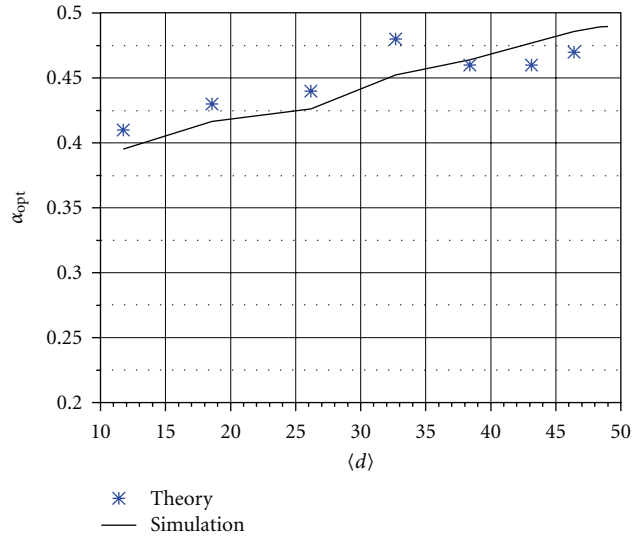

FIGURE 3: Example of α_{opt} for the UG algorithm under reduced connectivity due to randomly distributed (with uniform law) failure links.

5.1. UG Algorithm. By focusing on the UG algorithm, Figures 4(a) and 4(b) report the $\langle e(k) \rangle$ obtained in a single experiment, as a function of the share factor α (see (6)), for a fixed fully meshed and nonfully meshed ($r = 0.4$) network, respectively. In the latter case, reduced connectivity is due to limited coverage radius. The same will be, for the simulations regarding nonfully meshed networks, in the next subsections. The optimum share factor for the considered fully meshed network is $\alpha_{\text{opt}} = 0.48$, close to the value theoretically predicted in Table 1. In the following, we will justify such residual difference, as due to the random nature of the parameters involved. On the contrary, for the case of nonfully meshed networks, the optimum share factor is generally different: in Figure 4(b), for example, we see that $\alpha_{\text{opt}} = 0.26$, and a significant amount of time can be saved by adopting this optimum value in place of $\alpha = 0.5$: as an example, $e(k) = 10^{-3}$ requires 2556 clock ticks in the case of $\alpha = 0.5$ and "only" 2078 clock ticks in the case of $\alpha = 0.26$ (i.e., a reduction of about 19% in the convergence time). The gap is even more evident with $\alpha > 0.5$.

Obviously, to consider a single experiment gives an idea of the framework, but its significance is strongly limited by the random nature of the quantities involved. More significantly from a statistical viewpoint, and coherent with (2), for the same RGG and coverage radius, but changing the vectors of the initial conditions and transmission sequence, we can repeat the experiment R times (with R sufficiently high). This way, we find, at any attempt, an optimum value

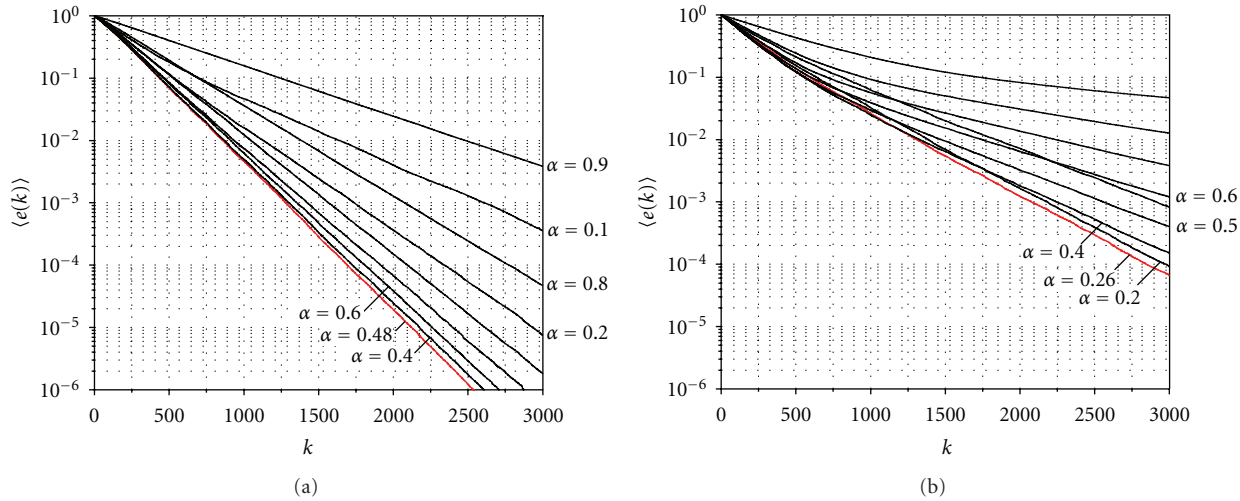

FIGURE 4: Simulated error of the UG algorithm for (a) a fully meshed and (b) a nonfully meshed ($r = 0.4$) RGG with $N = 50$ nodes and different values of α.

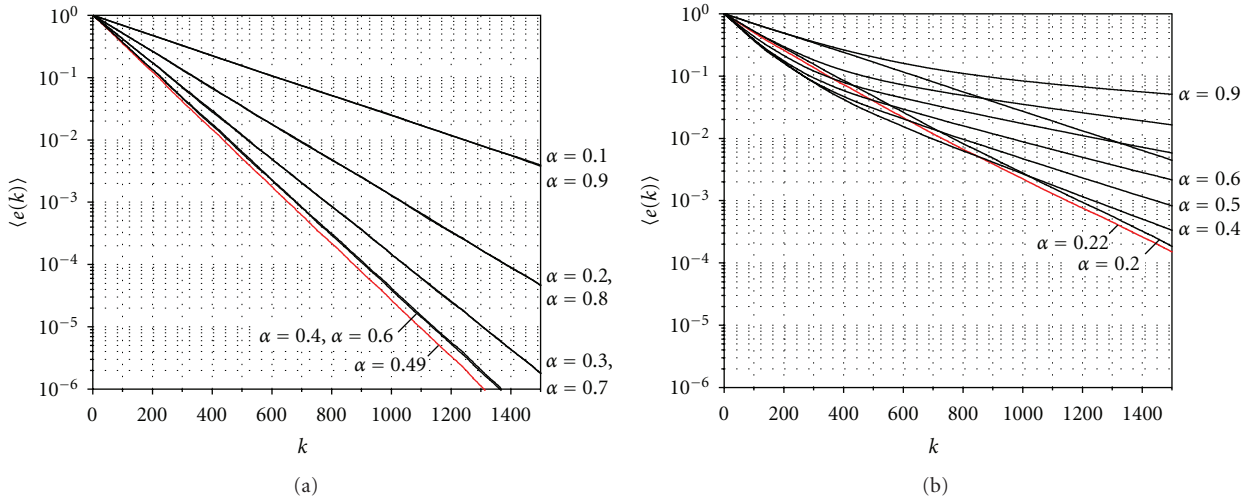

FIGURE 5: Simulated error of the BG algorithm for (a) a fully meshed and (b) a nonfully meshed ($r = 0.4$) RGG with $N = 50$ nodes and different values of α.

α_{opt}^m, with $m = 1, \ldots, R$, and then we compute the average optimum share factor as

$$\left\langle \alpha_{\text{opt}} \right\rangle = \frac{1}{R} \sum_{m=1}^{R} \alpha_{\text{opt}}^m. \tag{23}$$

Alternatively, we can consider more than one RGG, in order to better investigate the statistical nature of the problem. An example of the results obtained is shown in Table 2. Simulations have been carried out by considering 10 different realizations of an RGG with $N = 50$ nodes, and, for each RGG realization, $R = 100$ experiments have been performed (we remind that each experiment involves 200 randomly chosen pairs of initial conditions and transmitting node sequence). The table reports the $\langle \alpha_{\text{opt}} \rangle$ value and its standard deviation ($\sigma_{\alpha_{\text{opt}}}$), so found, for different values of the coverage radius.

Only values of $r \geq 0.3$ have been considered, since for $r < 0.3$ it is highly probable that the considered RGGs, randomly generated, are no longer connected, that is, some nodes remain isolated from the rest of the network. We observe that, even taking into account the several random aspects of the problem, we still obtain estimated values of α_{opt} that show a small dispersion around their mean. As an example, from Table 2, we see that the average optimum share factor for the case $r = 0.4$ is $\langle \alpha_{\text{opt}} \rangle|_{r=0.4} \approx 0.27$, while for the experiment reported in Figure 4(b) the optimum was $\alpha = 0.26$.

5.2. BG Algorithm. An analysis similar to that developed in Figure 4 is repeated in Figure 5 by considering the BG algorithm. In this case, the optimum share factor value is $\alpha_{\text{opt}} = 0.49$ for the fully meshed network and $\alpha_{\text{opt}} = 0.22$ for the nonfully meshed network. In the latter case, reaching

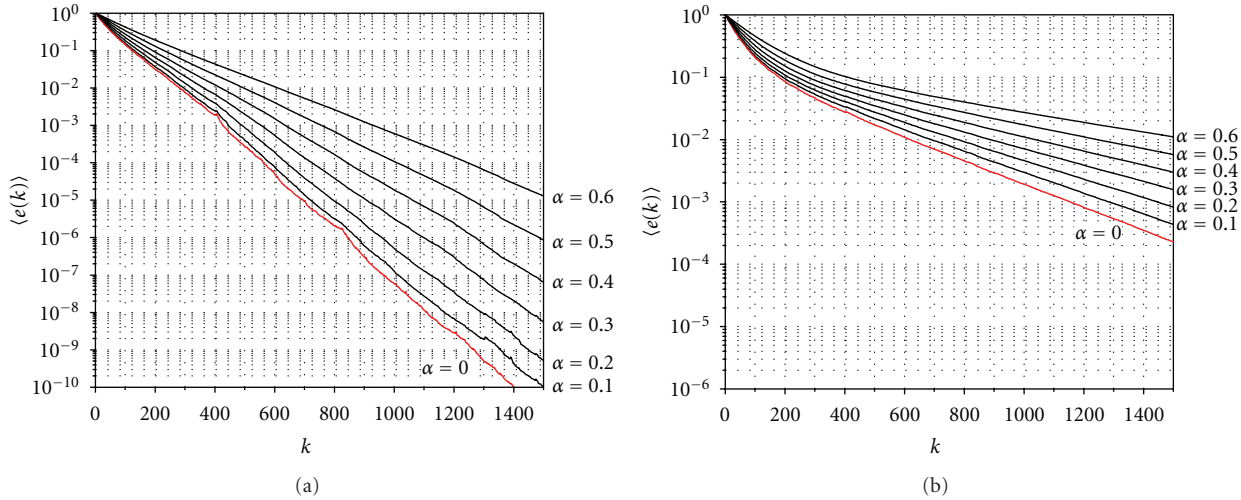

FIGURE 6: Simulated error of the B algorithm for (a) a fully meshed and (b) a nonfully meshed ($r = 0.4$) RGG with $N = 50$ nodes and different values of α.

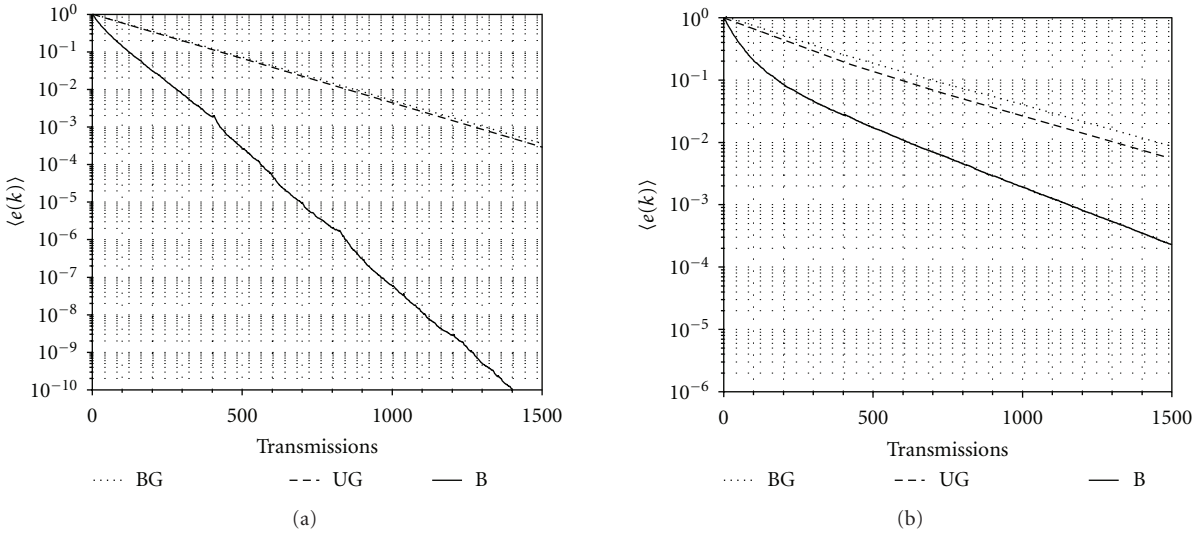

FIGURE 7: Performance comparison between the considered algorithms for (a) a fully meshed and (b) a nonfully meshed ($r = 0.4$) RGG with $N = 50$ nodes.

$e(k) = 10^{-3}$ requires 1148 clock ticks versus 1446 clock ticks with the more conventional choice $\alpha = 0.5$. So, the reduction in the convergence time is in the order of 21%.

5.3. B Algorithm. When considering the broadcast (B) algorithm, the uniform sharing assumption adopted in (7) can be generalized, for the case of nonfully meshed network, as follows:

$$\alpha_{ij} = \begin{cases} \alpha & j = i, \\ \dfrac{(1-\alpha)}{N_i} & d_{ij} \le r,\ j \ne i, \\ 0 & d_{ij} > r, \end{cases} \tag{24}$$

where α_{ij} is the share factor from the ith node to the jth node and N_i is the number of neighbors of the ith node within its coverage area. In Section 4, we have shown that, in the case of fully meshed networks, the optimum value of α is 0.

This is confirmed by numerical simulations, as we can notice in the example reported in Figure 6(a). As shown in Figure 6(b), for $r = 0.4$, the value of α_{opt} is 0 even in the case of a nonfully meshed network.

5.4. Comparison of the Three Algorithms. A comparison can be done among the averaging time performance of the considered algorithms. Two examples are shown in Figure 7, for the case of a fully meshed and a nonfully meshed network with $r = 0.4$. According to the considerations made in Section 4, to make the comparison fair, the averaging time has been determined as a function of the number of transmissions instead of the number of clock ticks. For all cases, the value of α has been optimized.

Table 2: Simulated mean and standard deviation of α_{opt} over 10 RGG realizations and 100 experiments for each RGG realization.

r	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1	1.1	1.2	1.3	1.4	1.5
$\langle\alpha_{opt}\rangle$	0.225	0.2707	0.3511	0.424	0.4624	0.4723	0.4767	0.4772	0.4773	0.4786	0.4782	0.4778	0.4771
$\sigma_{\alpha_{opt}}$	0.0481	0.0242	0.0213	0.0218	0.0231	0.0224	0.0219	0.0206	0.0206	0.0214	0.0204	0.0201	0.0208

The simulation results confirm the advantage offered by the B algorithm against the gossip algorithms: in the case of a fully meshed network, the former requires about 1/3 of the number of transmissions needed by the latter. In the case of a nonfully meshed network with $r = 0.4$, the convergence is slower, but the advantage offered by the B algorithm is still evident as expected. The curves of the UG and BG algorithms are practically superposed for fully meshed networks; on the contrary, for nonfully meshed networks, convergence of the unidirectional algorithm is slightly faster than that of the bidirectional algorithm. This result gives the designer precise indications about the best solution to adopt when a short convergence time is the main target to reach.

6. Conclusion

Despite the amount of literature already available on gossip-based algorithms for solving averaging problems, several aspects are still open and deserve further investigation. In this paper, we have considered a class of gossip algorithms that, through different choices of the updating rules, can originate unidirectional or bidirectional protocols. By extending the communication strategy, the same class can also define a broadcast algorithm. Because of the averaging target, the performance of these algorithms has been compared looking at the convergence time required for making the average error sufficiently small. Fixing attention on the RGG topology, we have verified, through theoretical arguments and simulations, that the performance of the unidirectional and bidirectional algorithms is equivalent in fully meshed networks, while it can be different in nonfully meshed networks.

To reach fast convergence, the share factor, which is a relevant parameter of the updating rule, should be optimized. In the case of fully meshed networks, such task is rather easy and yields explicit results. In the case of nonfully meshed networks, instead, two different situations must be considered. If reduced connectivity is due to link failures, randomly distributed, the optimum share factor can be predicted theoretically, and it does not differ significantly from 1/2. On the contrary, when the main effect is the limited coverage radius of the nodes, significant variations in the optimum share factor may occur depending on the resulting nodal degree. Calculation of the optimum share factor in the latter case could be made offline, through the approach presented in this paper. An online computation would be even better, as it could take into account changes in the network, like the appearance of link failures, but the increase in complexity it requires is sustainable only in few applications.

References

[1] H. Sabbineni and K. Chakrabarty, "An energy-efficient data delivery scheme for delay-sensitive traffic in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2010, Article ID 792068, 14 pages, 2010.

[2] S. M. Hedetniemi, S. T. Hedetniemi, and A. L. Liestman, "A survey of gossiping and broadcasting in communication networks," *Networks*, vol. 18, pp. 319–349, 1988.

[3] M. Dietzfelbinger, "Gossiping and broadcasting versus computing functions in networks," *Discrete Applied Mathematics*, vol. 137, no. 2, pp. 127–153, 2004.

[4] F. Bénézit, A. G. Dimakis, P. Thiran, and M. Vetterli, "Gossip along the way: order-optimal consensus through randomized path averaging," in *Proceedings of the 45th Annual Allerton Conference*, pp. 1226–1233, Allerton House, Ill, USA, September, 2007.

[5] J. Y. Chen, G. Pandurangan, and D. Xu, "Robust computation of aggregates in wireless sensor networks: distributed randomized algorithms and analysis," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 9, pp. 987–1000, 2006.

[6] A. D. G. Dimakis, A. D. Sarwate, and M. J. Wainwright, "Geographic gossip: efficient averaging for sensor networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 3, pp. 1205–1216, 2008.

[7] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-based ad hoc routing," in *Proceedings of the IEEE INFOCOM*, pp. 1707–1716, New York, NY, USA, June 2002.

[8] M. Rabbat, J. Haupt, A. Singh, and R. Nowak, "Decentralized compression and predistribution via randomized gossiping," in *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN '06)*, pp. 51–59, Nashville, Tenn, USA, April 2006.

[9] V. Saligrama, M. Alanyali, and O. Savas, "Distributed detection in sensor networks with packet losses and finite capacity links," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4118–4132, 2006.

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

[11] M. Mehyar, D. Spanos, J. Pongsajapan, S. H. Low, and R. M. Murray, "Asynchronous distributed averaging on communication networks," *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 512–520, 2007.

[12] Y. Bachrach, A. Parnes, A. D. Procaccia, and J. S. Rosenschein, "Gossip-based aggregation of trust in decentralized reputation systems," *Autonomous Agents and Multi-Agent Systems*, vol. 19, no. 2, pp. 153–172, 2009.

[13] M. E. Yildiz and A. Scaglione, "Computing along routes via gossiping," *IEEE Transactions on Signal Processing*, vol. 58, no. 6, pp. 3313–3327, 2010.

[14] S. Agarwal, J. P. Singh, and S. Dube, "Analysis and implementation of gossip-based P2P streaming with distributed incentive mechanisms for peer cooperation," *Advances in Multimedia*, vol. 2007, Article ID 84150, 12 pages, 2007.

[15] M. Hazewinkel, *Encyclopedia of Mathematics*, vol. 9, Springer, 1993.

[16] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[17] C. C. Moallemi and B. Van Roy, "Consensus propagation," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 4753–4766, 2006.

[18] M. Alanyali, V. Saligrama, and O. Savas, "A random-walk model for distributed computing in energy-limited networks," in *Proceedings of the 1st Workshop on Information Theory and Its Applications*, San Diego, Calif, USA, February 2006.

[19] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Analysis and optimization of randomized gossip algorithms," in *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC '04)*, pp. 5310–5315, Nassau, Bahamas, December 2004.

[20] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Gossip algorithms: design, analysis and applications," in *Proceedings of the IEEE INFOCOM*, pp. 1653–1664, Miami, Fla, USA, March 2005.

[21] D. Kempe, J. Kleinberg, and A. Demers, "Spatial gossip and resource location protocols," in *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pp. 163–172, Crete, Greece, July 2001.

[22] A. Olshevsky and J. N. Tsitsiklis, "Convergence speed in distributed consensus and averaging," *SIAM Journal on Control and Optimization*, vol. 48, no. 1, pp. 33–55, 2009.

[23] M. Baldi, F. Chiaraluce, and E. Zanaj, "Comparison of averaging algorithms for wireless sensor networks," in *Proceedings of the 3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA '08)*, Damascus, Syria, April 2008.

[24] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS '03)*, pp. 482–491, Cambridge, Mass, USA, October 2003.

[25] X.-Y. Li, K. Moaveninejad, and O. Frieder, "Regional gossip routing for wireless ad hoc networks," in *Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks (LCN '03)*, Bonn, Germany, October 2003.

[26] L. J. García Villalba, A. L. Sandoval Orozco, A. Triviño Cabrera, and C. J. Barenco Abbas, "Routing protocols in wireless sensor networks," *Sensors*, vol. 9, no. 11, pp. 8399–8421, 2009.

[27] M. Baldi, F. Chiaraluce, and E. Zanaj, "Performance of gossip algorithms in wireless sensor networks," in *Solutions on Embedded Systems*, vol. 81 of *Lecture Notes in Electrical Engineering*, pp. 3–16, Springer, 2011.

[28] R. Merris, "A survey of graph Laplacians," *Linear and Multilinear Algebra*, vol. 39, no. 1-2, pp. 19–31, 1995.

[29] B. Mohar, "The Laplacian spectrum of graphs," in *Graph Theory, Combinatorics, and Applications*, Y. Alavi, G. Chartrand, O. R. Oellermann, and A. J. Schwenk, Eds., pp. 871–898, Wiley, 1991.

[30] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.

[31] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking, "Randomized rumor spreading," in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pp. 564–574, Redondo Beach, Calif, USA, November 2000.

[32] E. Zanaj, M. Baldi, and F. Chiaraluce, "Share factors optimization in the push-sum algorithm for sensor networks," in *Proceedings of the 16th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2008)*, Dubrovnik, Croatia, November 2008.