Efficiency of Unicast and Broadcast Gossip Algorithms for Wireless Sensor Networks

Elma Zanaj, Marco Baldi, and Franco Chiaraluce

Original scientific paper

Abstract: **Gossip is a well-known technique for distributed computing in an arbitrarily connected network, that can be adopted effectively in wireless sensor networks. Gossip algorithms have been widely studied in previous literature, but mostly from a theoretical point of view. The aim of this paper is to verify the behavior of the gossip approach in practical scenarios, through the analysis and interpretation of simulated results. So, we investigate the impact of optimizing the neighbor selection probabilities, the effect of multiple link failures and that of limited transmission radius. The possibility to use broadcast-like algorithms to increase the rate of convergence in averaging problems is also discussed and its advantage estimated.**

Index terms: **wireless sensor networks, averaging problems, gossip, convergence rate**

I. Introduction

Wireless sensors are usually small devices, characterized by limited communications capabilities because of energy and bandwidth constraints. However, they can merge their scarce resources in networks that aim at supporting very important decisions or actions. They are individually cheap, unintelligent, imprecise and unreliable but, being part of a large network, they together may produce reliable and robust information, as the result of their physical measurements and of a collaborative process. Sensors are seen as nodes of a network that can work with the data they obtain. Typical ways for processing sensors data consist in using one of the following options:

i) elaborating the whole amount of data in a sink node;
ii) elaborating data in each node, exchanging information between them; so, at the end, each node possesses the whole information.

We adopt the second model, where nodes are supposed to implement some elementary operations like averages, sums and products. In particular, it is interesting to investigate how the nodes average their values, trying to achieve a general consensus in the shortest possible time. This way, they use a limited amount of local information to allow distributed knowledge of global network properties.

Manuscript received February, 2008 and revised June, 2008. This paper was presented in part at the International Conference on Software, Telecommunications and Computer Networks (SoftCOM) 2007.

Authors are with Università Politecnica delle Marche, Ancona, Italy (e-mail: {e.zanaj, m.baldi, f.chiaraluce}@univpm.it)

As mentioned above, nodes can elaborate together; so the result is a collective property of the network, that, this way, will have eventually uniform information. We mean the nodes talk together or, in more explicit, though pictorial, terms, they "gossip". This kind of gossip is similar to human gossip. For example, when, in a crowd, somebody has a piece of news, she/he shares it with her/his neighbors. All the people "connected" with her/him learn the news item and repeat it to their neighbors, so that the new information is spread like an epidemic illness.

A similar process exists in wireless sensor networks, when they are implemented as peer-to-peer networks formed by many small and simple devices, able to measure some quantities and to transmit their measured values to neighboring nodes [1]. When the whole network (or a subset of it) is expected to produce a unique value for a measured quantity, a group of nodes must communicate in order to merge their single contributions into a common result; in other terms, the network self-stabilizes the measured value [2].

Noting by x_i and x_j the local measures of the i-th and j-th nodes, an interaction among them produces as output $(x_i + x_j)/2$, that is acquired by both nodes and used for the subsequent interaction. Through information propagation, the object is to find, in the shortest possible time, an estimate of the average $x_{ave} = \sum_{i=1}^{N} x_i \Big/ N$, where N is the total number of nodes in the network. As known, the average provides the minimum mean squared error (MMSE) estimate of the sensed quantity.

One of the most interesting features of gossip-based networks is scalability. Each sensor node, in fact, sends a fixed number of messages (indeed only one message in the considered implementation), and this number is independent of the network size. On the other hand, provided that effective communication among nodes is achieved through suitable protocols, a fundamental aspect of gossip algorithms concerns convergence to the average [3]-[5]. This topic has been discussed extensively in previous literature (see [6]-[10], for example, where other references can be found). Most of these papers, however, are focused on theoretical issues and system modeling. In [6], 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. In this kind of problems, optimization basically consists in 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 noted by T_{ave} in the following. In [6] it was found that T_{ave} depends on the second largest eigenvalue, λ_2, of a stochastic matrix characterizing the averaging algorithm (this matrix will be defined afterward): the smaller this eigenvalue, the faster the averaging algorithm. So, the fastest averaging algorithm is obtained by minimizing the eigenvalue over the set of allowed gossip algorithms. An efficient procedure was proposed to solve the problem. Moreover, a lower and an upper bound for the averaging time, in terms of λ_2, were derived. Similar, though less explicit, bounds can be found in other papers (see [2] and [11], for example).

In spite of these in-depth analytical studies, however, the performance of the gossip algorithms has been rarely evaluated in simulation experiments. To support the study through simulations is an important issue, as it permits to validate the analytical model and to test the distance between the theoretical expectations and the true performance. The aim of this paper is to present some examples of such simulations, and discuss the corresponding results.

Referring to a specific, though arbitrarily generated, sensor network, first we discuss the impact of the optimization procedure in a fully-meshed network, showing that the convergence rate of the gossip algorithm can be only marginally improved in many practical cases (like the ones here considered). Then, we examine the effect of possible link failures that, preventing connection between nodes, necessarily increase the convergence time. Simulation permits to quantify the extent of such an increase, also in comparison with a non fully-meshed network. To model the latter, a classic procedure consists in fixing a transmission radius: two nodes can communicate only if their distance is smaller than the transmission radius. We investigate the worsening in the convergence rate that is due to a limited radius. Moreover, as a wireless network is naturally suited to apply broadcast-like algorithms [12], where the transmission from a sensor can be received by more nodes simultaneously, we simulate the performance of this variant of the gossip, too, and show it can produce a significant improvement of the convergence speed.

The organization of the paper is as follows. In Section II we introduce the notation and provide a short summary of the optimization approach presented in [6]. In Section III we describe the simulator, focusing on the relevant parameters it is based on. Section IV reports numerical results both for the ideal case of a fully-meshed network and for more practical scenarios (as in presence of link failures or limited coverage). Section V describes the broadcast-like algorithm and its benefit against the more conventional gossip. Finally, Section VI concludes the paper.

II. Notation and Previous Analytical Results

A network of N nodes can be described by a connected graph $G(V, E)$, where V is the vertex set containing the nodes and E is the edge set. Two nodes that have an edge between them are called neighbors. The class of gossip algorithms we consider is characterized by an $N\times N$ matrix $\mathbf{P} = [P_{ij}]$ of non-negative entries with the condition that $P_{ij} \neq 0$ only if $(i, j) \in E$ and $i \neq j$. We consider an asynchronous time model, in which each node has a clock which ticks at the times of a rate 1 Poisson process. Therefore, the inter-tick 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 k-th time-slot and, on average, there are N clock ticks per unit of absolute time. In [6] it is shown how to pass from quantities measured in terms of the number of clock ticks to quantities expressed in absolute time.

We assume that each node i, when its clock ticks, contacts one of its neighbors, j, by choosing it according with the selection probability P_{ij}. At each clock tick, node i tries to communicate and its attempt has always success in absence of link failures. Therefore, in such case, $\mathbf{P}$ is a stochastic matrix (i.e., each row sums to 1); consequently, its largest eigenvalue is equal to 1, while all the remaining $N - 1$ eigenvalues have magnitude strictly smaller than 1 [13].

Let us denote by $\mathbf{x}(0) = [x_1(0), x_2(0),\ldots, x_N(0)]^T$ a vector collecting the initial (sensed) values at the nodes. So, $x_{ave} = \sum_{i=1}^{N} x_i \big/ N$ is the average of the entries of $\mathbf{x}(0)$; the goal of a gossip algorithm is to compute x_{ave} in a distributed manner. Only one node is contacted at each time, and simultaneous transmissions are avoided through suitable access control protocols. After k ticks, the updated values are collected in the vector $\mathbf{x}(k) = [x_1(k), x_2(k),\ldots, x_N(k)]^T$, where superscript ^{T} denotes transposition. On the other hand, it is easy to verify that the following recursive relationship holds:

$$\mathbf{x}(k) = \mathbf{W}(k)\cdot\mathbf{x}(k-1)$$

where, with probability P_{ij}/N, the random matrix $\mathbf{W}(k)$ is:

$$\mathbf{W}(k) = \mathbf{W_{ij}} = \mathbf{I} - \frac{(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T}{2}$$

where $\mathbf{I}$ represents the identity matrix of size $N\times N$, and $\mathbf{e}_i$ is the vector with all 0's except for a 1 in the i-th coordinate. Now, for a given matrix $\mathbf{P}$, let us define the ε-averaging time of a gossip algorithm as follows [6]:

$$T_{ave}(\varepsilon, \mathbf{P}) = \sup_{\mathbf{x}(0)} T_{ave}^{\mathbf{x}(0)}(\varepsilon, \mathbf{P}) \tag{1}$$

with

$$T_{ave}^{\mathbf{x}(0)}(\varepsilon, \mathbf{P}) = \inf\left\{ k : \Pr\left(\frac{\|\mathbf{x}(k) - x_{ave}\mathbf{1}\|}{\|\mathbf{x}(0)\|} \geq \varepsilon \right) \leq \varepsilon \right\},$$

where: Pr(A) stays for the probability of A, $\|\mathbf{v}\|$ denotes the l^2-norm of vector $\mathbf{v}$, and $\mathbf{1}$ is an $N\times 1$ vector with all components equal to 1. In practice, $T_{ave}(\varepsilon, \mathbf{P})$ is the smallest time, expressed in number of clock ticks, it takes for $\mathbf{x}(k)$ to get within ε of $x_{ave}\mathbf{1}$ with probability $1 - \varepsilon$, regardless of the initial value $\mathbf{x}(0)$. If ε is small, then this probability is high.

As mentioned above, in [6], by using some results on the convergence of moments, it was proved that the ε-averaging time, for the asynchronous time model, can be bounded as follows:

$$\frac{0.5\log e^{-1}}{\log[\lambda_2(\mathbf{C})]^{-1}} \le T_{ave}(\varepsilon,\mathbf{P}) \le \frac{3\log e^{-1}}{\log[\lambda_2(\mathbf{C})]^{-1}}$$

where $\lambda_2(\mathbf{C})$ is the second largest eigenvalue of matrix

$$\mathbf{C} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} P_{ij}\mathbf{W_{ij}}.$$

It is evident, from their expressions, that the ratio between the upper bound and the lower bound is always equal to 6, and that their absolute values decrease for decreasing λ_2. This observation introduces the optimization problem, as finding the fastest averaging algorithm corresponds to finding **P** such that $\lambda_2(\mathbf{C})$ is the smallest one, while satisfying constraints on **P**. Formally, the optimization problem is as follows:

- minimize $\lambda_2(\mathbf{C})$,
- subject to $P_{ij} \ge 0$; $P_{ij} \ne 0$ if $(i, j) \in E$; $\Sigma_j P_{ij} = 1$ for any i.

In [6] a distributed subgradient method was proposed, able to solve the optimization problem over the network. In essence, the solving method can be summarized as follows.

First of all, for the sake of convenience, the non-zero entries of matrix **P** are collected in a vector **p**. Noting by m the number of edges in the graph G, a number l is assigned to each edge (i, j), with $i < j$; this way, the l-th entry of **p**, with $l = 1, 2,\ldots, m$ is $p_l = P_{ij}$. On the other hand, as **P** is stochastic (but not, necessarily, double stochastic), there is no symmetry requirement on **P**, and the entries of **p** corresponding to P_{ji} must be separately specified. This is done by setting, $p_{-l} = P_{ji}$. Globally, vector **p** has therefore $2m$ entries, that corresponds to replace each edge in the undirected graph G with two directed edges, one in each direction. Then, a classic subgradient method is applied to vector **p**, which consists in updating this vector, through an iterative procedure, as follows:

$$\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} - v_k\mathbf{g}^{(k)}$$

where $\mathbf{g}^{(k)}$ is a subgradient for vector **p** at the k-th clock tick ($\mathbf{p}^{(k)}$) and v_k is a step size (to be properly chosen). The l-th component of **g** can be obtained as

$$g_l = -\frac{1}{2N}(u_i - u_j)^2$$

where u_i is the i-th component of the unit eigenvector associated with $\lambda_2(\mathbf{C})$, i.e., a solution of the equation

$$\lambda_2(\mathbf{C}) = \mathbf{u}^T\mathbf{C}\mathbf{u}$$

Obviously, matrix **C** is updated in turn, according with its definition, because of updating of the P_{ij}'s, when the subgradient method proceeds.

On the other hand, it is not sure that the updated values of **p** are feasible; for this reason, the subgradient step is followed by a projection onto feasible set step. In practice, noting by $\tilde{p}_{ij}$ the non-zero entries in the i-th row of **P**, updated according with the subgradient $\mathbf{g}^{(k)}$, it is checked if $\sum_j \tilde{p}_{ij} \le 1, \ \forall i$; if not, a real δ_i is found such that $\sum_j (\tilde{p}_{ij} - \delta_i) = 1$ and the $\tilde{p}_{ij}$'s changed in $\tilde{p}_{ij} - \delta_i$ [6].

III. Simulation Parameters

In a first series of simulations, we have developed numerical programs in Octave and C++ language that permit:

- to simulate the performance of the gossip algorithm for a given matrix **P**;
- to optimize matrix **P**, in such a way as to find a good approximation of the fastest averaging algorithm;
- to compute lower and upper bounds for both cases.

Simulation aims at exploring where the simulated convergence time is located against the lower and upper bounds and, more important, to determine any actual (i.e., not just theoretically foreseen) improvement achievable through optimization, in terms of reduced convergence time.

The starting values P_{ij} are generated according to an assigned distribution, satisfying the property to have a stochastic matrix. For the gossip algorithm, simulation is also necessary to produce samples of the sensed quantities. For this purpose, we have used different probability distributions. As a first example, we have adopted a gaussian generator, and acted on its parameters to simulate different operational scenarios. This choice is suited to model practical cases, like the temperature measurement, where the distributed computation strategy can be efficiently applied.

Once having obtained matrix **P**, simulation proceeds as follows: a uniform random generator selects one node at a time (each node has a probability $1/N$ to be selected). Noting by i the selected node, the node j it contacts is chosen, once again, at random, according with the P_{ij}'s distribution. Changing the seed of the uniform random generator, the communication sequence is changed as well, this way obtaining different realizations of the considered random experiment.

The random variable $e(k) = \|\mathbf{x}(k) - x_{ave}\mathbf{1}\|/\|\mathbf{x}(0)\|$ is estimated in R experiments, thus obtaining a set of R curves $\varepsilon_{sim}{}^q(k)$, $q = 1\ldots R$, expressed as functions of the number of clock ticks k. This permits to draw $T_{sim}(\varepsilon, \mathbf{P})$ curves that can be compared with the bounds on $T_{ave}(\varepsilon, \mathbf{P})$, as provided by the analytical model. The simulated curves are also averaged among the R realizations, i.e.,

$$\langle \varepsilon_{sim}(k) \rangle = \frac{1}{R}\sum_{q=1}^{R} \varepsilon_{sim}^q(k), \qquad \forall k$$

thus obtaining an estimated average curve $\langle T_{sim}(\varepsilon, \mathbf{P})\rangle$, intended as a set of (ε, k) couples and referred to the specific matrix **P** and the specific (though arbitrary) initial condition $\mathbf{x}(0)$. According to the probability theory, it is:

$$\lim_{R\to\infty} \langle \varepsilon_{sim}(k) \rangle = \bar{e}(k), \qquad \forall k$$

where $\bar{e}(k)$ represents the average of $e(k)$. It is easy to prove that the variance of the input vector probability density function (p.d.f.) has no impact on $e(k)$. For this purpose, let us consider a simple network with $N = 3$ nodes, and let us suppose that, because of the assumptions on the p.d.f., the input vector is $\mathbf{x}^a(0) = [x_1(0), x_2(0), x_3(0)]^T$. If, at the first clock tick, node #1 talks with node #2, the vector at $k = 1$ results in:

$$\mathbf{x}^a(1) = \left[x_1(1) = \frac{x_1(0) + x_2(0)}{2}, x_2(1) = x_1(1), x_3(1) = x_3(0) \right]^T$$

Correspondingly, we have:

$$e^a(1) = \frac{\sqrt{\left(x_1(1) - x_{ave}^a\right)^2 + \left(x_2(1) - x_{ave}^a\right)^2 + \left(x_3(1) - x_{ave}^a\right)^2}}{\sqrt{x_1^2(0) + x_2^2(0) + x_3^2(0)}} .$$

If the variance of the p.d.f. is scaled by a factor α^2, the random extraction of the input vector would provide $\mathbf{x}^b(0) = \alpha\mathbf{x}^a(0)$. The same scale factor acts on $\mathbf{x}^b{}_{ave}$, since $x_{ave}^b = \alpha\sum_{i=1}^N x_i(0) \big/ N = \alpha x_{ave}^a$. Under the same assumption that node #1 and node #2 talk between them first, we have $\mathbf{x}^b(1) = \alpha\mathbf{x}^a(1)$; anyway, the value of $e^b(1)$ is unchanged as:

$$e^b(1) = \sqrt{\alpha^2 \sum_{i=1}^3 \left(x_i(1) - x_{ave}^a\right)^2} \Bigg/ \sqrt{\alpha^2 \sum_{i=1}^3 x_i^2(0)} = e^a(1)$$

From now on, the evolution of the two networks with different input variance proceeds exactly in the same way.

To give an idea of the dispersion of the simulated curves around the mean, the standard deviation is also computed as:

$$\sigma(k) = \sqrt{\frac{1}{R-1} \sum_{q=1}^R \left(\varepsilon_{sim}^q(k) - \langle\varepsilon_{sim}(k)\rangle\right)^2}$$

As known, division by $R - 1$, instead of R, makes impartial the estimator. Another parameter we have simulated is $T_{ave}{}^{\mathbf{x}(0),R}(\varepsilon, \mathbf{P})$, that represents an estimate of $T_{ave}{}^{\mathbf{x}(0)}(\varepsilon, \mathbf{P})$, for the specific vector of initial values $\mathbf{x}(0)$ and for a finite number (R) of realizations. In fact, though important from a theoretic viewpoint, $T_{ave}(\varepsilon, \mathbf{P})$ is impossible to evaluate in practical cases. Also $T_{ave}{}^{\mathbf{x}(0)}(\varepsilon, \mathbf{P})$, that is referred to a fixed initial vector $\mathbf{x}(0)$, should be evaluated considering, in principle, an infinite number of realizations.

In order to calculate $T_{ave}{}^{\mathbf{x}(0),R}(\varepsilon, \mathbf{P})$, the results of gossip simulations are stored in a matrix $\mathbf{S}$. The q-th row of $\mathbf{S}$ contains the value of $\varepsilon_{sim}{}^q(k)$ for the q-th realization; therefore $\mathbf{S}$ has R rows and K columns, where K is the maximum number of simulated clock ticks (assumed to be the same for all realizations), i.e., $1 \le k \le K$.

The estimate of $T_{ave}{}^{\mathbf{x}(0),R}(\varepsilon, \mathbf{P})$ is the result of a classic quantile evaluation. The value of ε is varied within an interval of interest $[\varepsilon_{\min}, \varepsilon_{\max}]$, according with a given step size. The q-th row of $\mathbf{S}$ is scanned, searching for the minimum k that gives $\varepsilon_{sim}{}^q(k) < \varepsilon$. The values of $k(q)$ so found are stored in a vector $\mathbf{t}$ whose elements are sorted in ascending order. The element in position $l \approx R(1 - \varepsilon) + 1$ is then identified and used as $T_{ave}{}^{\mathbf{x}(0),R}(\varepsilon, \mathbf{P})$. For $R \to \infty$, the curves of $\langle T_{sim}(\varepsilon, \mathbf{P})\rangle$ and $T_{ave}{}^{\mathbf{x}(0),R}(\varepsilon, \mathbf{P})$ intersect at the point corresponding to $\varepsilon = 1/2$.

IV. Numerical Simulations

A. Fully-meshed Networks

Fig. 1 shows simulation results for a fully-meshed network with $N = 50$ nodes. Explicitly, this means that $P_{ij} \neq 0$ for any (i, j), with $i \neq j$.

The values of P_{ij} are obtained through a uniform random generator, under the constraint $\Sigma_{ij}P_{ij} = 1$; suitable normalization is applied to satisfy the constraint. The assumption of a fully-meshed network is unrealistic in most operational environments, but it represents a useful benchmark. On the other hand, in Subsection IV.C, this hypothesis will be removed, and we will consider the effect of a limited transmission radius for each node.

As expected, the simulated curves place themselves within the lower and upper bounds, determined through the calculation of $\lambda_2(\mathbf{C})$. In the figure, these bounds have been denoted by $\inf\{T_{ave}\}$ and $\sup\{T_{ave}\}$, respectively. We observe that the bounds are rather loose (because of the factor 6 in their ratio). However, simulations demonstrate that the ε-averaging time is closer to the upper bound than to the lower bound. The curves of $\langle T_{sim}(\varepsilon, \mathbf{P})\rangle \pm \sigma$ have been also plotted in the figure, for the sake of clarity.

Matrix $\mathbf{P}$ can be optimized according with the subgradient method described in Section II. This corresponds to minimize the eigenvalue $\lambda_2(\mathbf{C})$, this way reducing both the upper and lower bounds to the averaging time. In Fig. 2 we compare the results already shown in Fig. 1 with those of numerical simulations for the same example, but considering the optimized version of matrix $\mathbf{P}$.

From the figure, we observe that the optimization process increases, as expected, the slope of the bound curves, thus accelerating, in principle, the averaging process. Nevertheless, the simulated realizations of the gossip algorithm seem scarcely affected by optimization: the original and optimized curves, both for $\langle T_{sim}(\varepsilon, \mathbf{P})\rangle$ and $T_{ave}{}^{\mathbf{x}(0),R}(\varepsilon, \mathbf{P})$, are in fact practically superposed.

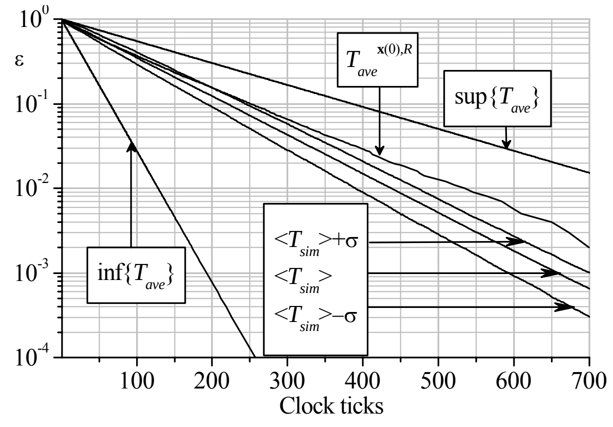

Fig. 1. Bounds and simulated results for a fully-meshed network with N = 50 nodes

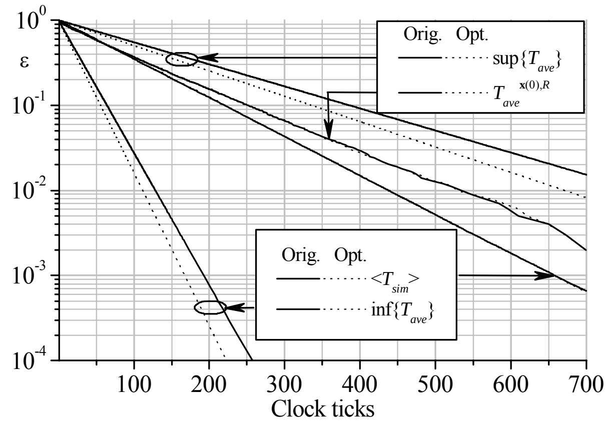

Fig. 2. Original (solid lines) and optimized (dotted lines) case for a fully-meshed network with N = 50 nodes

B. Effect of Multiple Link Failures

In Section IV.A we have assumed that all the selection probabilities P_{ij}, with $i \neq j$, are different from 0, which means that the i-th node is able to communicate with any other node of the network. Actually, because of natural, or artificial, obstacles or even node disruptions (particularly when the sensors are placed in hostile environments) some elements of matrix **P** can be forced to become zero, so matrix **P** is no longer stochastic. A relevant feature of gossip algorithms is their good tolerance to faults: the convergence of the averaging process is in fact generally ensured, although the averaging time can increase significantly. Some examples are presented next.

In the simulations, faulty links are chosen in a pseudo-random way, through a uniform generator, and their elements in matrix **P** are substituted with null entries. This corresponds to a multi-link failure model, that simulates the loss of connectivity in a subset of all the couples of linked nodes. Due to the fact that matrix **P** is no longer stochastic, the i-th node of the network has probability to communicate $\Sigma_j P_{ij} < 1$. In other terms, at each clock tick, a node could try to connect to another node without success, thus wasting the communication attempt.

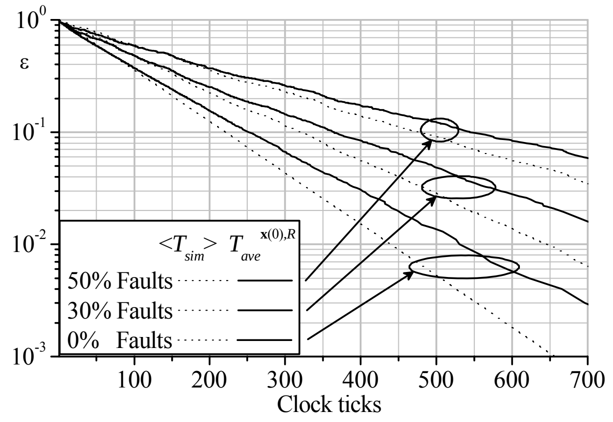

Fig. 3. Effect of link failures in a fully-meshed network with N = 50 nodes

Simulated curves for faulty networks are reported in Fig. 3 for different percentages of faults. All curves are monotonously decreasing; this implies that the averaging algorithm converges, even in the case of a high number of link failures. However, failures reflect on the averaging time. If we refer to $T_{ave}^{x(0),R}(\varepsilon, \mathbf{P})$ with the target ε = 0.1, the number of steps required in absence of link failures (k = 250) increases by about 1.5 times with 30% link failures (k = 366) and is more than doubled with 50% link failures (k = 551).

The performance of a fully-meshed network affected by link failures can be compared with that of a non fully-meshed network with the same topology.

In Section IV.C non fully-meshed networks will be modeled through the introduction of a transmission radius for each node. In this section we adopt a more approximate representation, where missing links are generated at random, according with a given percentage. Precisely, we speak of y% meshed network (y = 10 or 20, for example) if the density of non zero elements P_{ij} in matrix **P** is of the same order. It should be noted that to have an y% meshed network is not equivalent to have a fully meshed network with y% link failures. In both cases only part of the nodes can be reached directly, while communications towards the other require transit through intermediate nodes. In the case of faulty networks, however, transmissions over faulty links are lost; in non fully-meshed networks, instead, the missing links are never used (see section IV.C for details about the way it can be) and, in fact, matrix **P** is stochastic. So, in the latter case, a faster convergence is expected when the two kind of networks are compared for a given percentage of missing links.

An example of such comparison is shown in Fig. 4; the simulated non-faulty network is regular (i.e., each sensor is linked to the same number of other sensors), but irregular matrices **P**'s could be considered as well. Fig. 4 shows that non fully-meshed networks reach convergence within a smaller number of clock cycles with respect to faulty fully-meshed networks. However, the performance of a 10% meshed network is comparable with that of a fully-meshed network affected by 30% link failures (at least for the specified conditions and in the region of ε values explored).

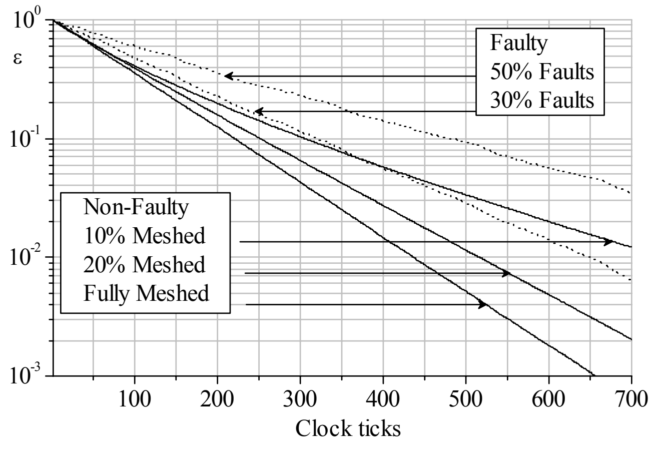

Fig. 4. Values of $\langle T_{sim}(\varepsilon, \mathbf{P})\rangle$ for networks with different mesh levels compared with a faulty fully-meshed network

TABLE I
NUMBER OF CLOCK TICKS CORRESPONDING TO $\langle T_{SIM}(\varepsilon, \mathbf{P})\rangle$ AT $\varepsilon = 0.1$

Network type	*k*
Fully-meshed network without link failures	220
Fully-meshed network with 30% link failures	320
Fully-meshed network with 50% link failures	481
20% meshed network without link failures	251
10% meshed network without link failures	305

This result is clearly shown in Table I, that reports the values of k for which the $\langle T_{sim}(\varepsilon, \mathbf{P})\rangle$ curve is at $\varepsilon = 0.1$: a 10% meshed network free of link failures employs almost the same time (number of clock ticks) as a fully-meshed network with 30% link failures. This means that, despite the gossip algorithm is fault tolerant in the sense that its absolute convergence is scarcely affected by link failures, the occurrence of faults significantly influences the convergence time.

Finally, it should be noted that the convergence time gives a measure of the complexity of the distributed algorithm and, therefore, of the power consumption at each node. In presence of faulty links, convergence is reached, but at the expense of an increase in the computational effort. However, the simplicity of the operations required by each node seems able to guarantee the feasibility of the system even in very unfavorable conditions.

C. Effect of a Limited Transmission Radius

A more realistic model of non fully-meshed network takes into account that, because of its limited energy capabilities, each sensor can communicate only with sensors within some fixed radius $r > 0$.

Let us consider a very typical situation where sensors are placed uniformly at random in an area. This corresponds to the assumption of a well-known random geometric graph [14]. Before the gossip algorithm starts, i.e., at the beginning of the procedure, each sensor informs the others it is active and ready to interact. This way, the i-th node, $i = 1, \ldots, N$, can recognize the nodes it can link, whose number will be denoted by N_i in the following, and can fix the selection rule, for example according with a uniform law. Explicitly, this means to assume:

$$P_{ij} = \begin{cases} 1/N_i, & d_{ij} \le r \\ 0, & d_{ij} > r \end{cases} \tag{2}$$

having noted by d_{ij} the Euclidean distance between the i-th and the j-th nodes. According with (2), the nodes at distance $d_{ij} \le r$ can be reached in one hop, that is, by using a single transmission. Conversely, communication with nodes at distance $d_{ij} > r$ require multiple hops, that is passing through intermediate nodes. Depending on the value of r, some nodes could be unreachable, since disjointed from the rest of the network. This situation, that prevents full connectivity, should be avoided (and, in fact, it is not considered in this paper), as it does not permit 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[1], more involved rules could be adopted, for example depending on the value of the Euclidean distances between the nodes.

On the other hand, based on the results in Subsection IV.A, we can expect that changing the selection rule, and even optimizing it, does not produce any significant improvement if the new rule maintains selection within the transmission radius of the considered node. Localization can contribute to improve significantly the convergence speed but at the expense of an increase in complexity. In [15], for example, a geographic gossip algorithm has been proposed, based on greedy routing, that is potentially able to provide remarkable gains. But 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. To simulate the effect of a limited transmission radius, we have generated a random geometric graph, for a fixed number of nodes; an example, for $N = 50$, is shown in Fig. 5; dots indicate the positions of the sensors. For the sake of convenience, we have normalized the radius to the side of the square (that is unit in the figure), and repeated the experiments for different values of r. Some results obtained for the considered graph are plotted in Fig. 6. The curves report the value of $\langle \varepsilon_{sim}(k)\rangle$; contrary to the previous figures, in this case the initial condition $\mathbf{x}(0)$ has been randomly changed at the beginning of any simulation, in such a way as to average possible good and bad conditions. It is evident, from the figure, that the convergence speed becomes slower and slower for decreasing r. This result is expected, but simulation permits to quantify an effect that cannot be described through asymptotic analytical studies.

V. BROADCAST-LIKE ALGORITHM

In a wireless network, when a node transmits some information, all the other nodes in its coverage area are able to receive the transmitted data. This suggests to implement a modified version of the gossip algorithm that, at the expense of a slight increase in complexity, permits to reduce significantly the averaging time.

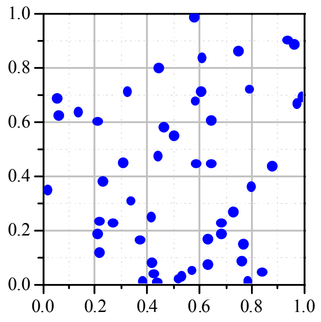

Fig. 5. Example of random geometric graph for a network with N = 50 nodes

[1] This facility is frequent in modern sensor networks but, obviously, it needs extra-processing.

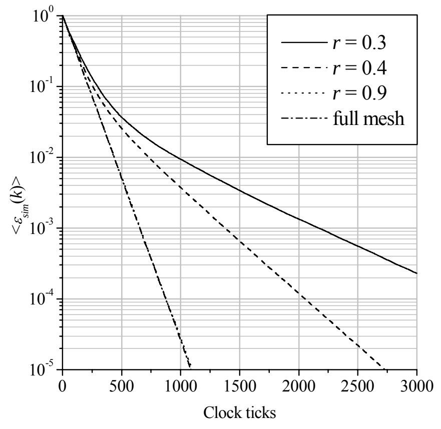

Fig. 6. Performance of the gossip algorithm for different values of the transmission radius

Contrary to the gossip described in the previous sections, this modified protocol is not bidirectional, in the sense that information flows from a transmitting node to a number of receiving nodes (depending on the transmission radius and the random nodes distribution) but not in the opposite sense. More explicitly, in this modified protocol, each node i maintains a sum, $s_i(k)$, and a weight, $w_i(k)$. When the algorithm starts, that is for $k = 0$, it is $w_i(0) = 1$ and $s_i(0) = x_i(0) = x_i$, that coincides with the initial sensed value at node i. When the i-th node's clock ticks, say at step k, the node splits its information into $N_i + 1$ parts; it keeps the first, so that $[w_i(k+1), s_i(k+1)] = \alpha_i[w_i(k), s_i(k)]$, while sends each neighbor j one of the remaining parts: $\alpha_{ij}[w_i(k), s_i(k)]$. Node j receives the transmission and updates its values by adding the received ones, so that $[w_j(k+1), s_j(k+1)] = [w_j(k), s_j(k)] + \alpha_{ij}[w_i(k), s_i(k)]$. As stated in the expressions, this sharing is ruled by the *share parameters* α_i and α_{ij}; these parameters can be collected in a matrix **A**, having $\alpha_{ii} = \alpha_i$ along the main diagonal. The elements of **A**, satisfying the condition $\alpha_i + \Sigma_j\alpha_{ij} = 1$, can be chosen at random or following some suitable deterministic rule. We prefer the second choice and adopt, in our simulations, the following two simple laws:

$$\alpha_{ij} = \begin{cases} 1/(N_i+1), & d_{ij} \le r, \forall i, j; \\ 0, & d_{ij} > r; \end{cases} \tag{3}$$

$$\alpha_{ij} = \begin{cases} \alpha_i, & j = i; \\ (1-\alpha_i)/N_i, & d_{ij} \le r, j \ne i; \\ 0, & d_{ij} > r. \end{cases} \tag{4}$$

According with (3) the information from node i is equally split among its neighbors and itself; according with (4), instead, the self-share can be different. In the latter case, the value of α_i should be optimized. An example of the impact of different α_i for a fully-meshed network is shown in Fig. 7. It is possible to verify [16] that the faster curve [corresponding to the choice (4) with $\alpha_i \approx 0$] can be also achieved by using the uniform law (3).

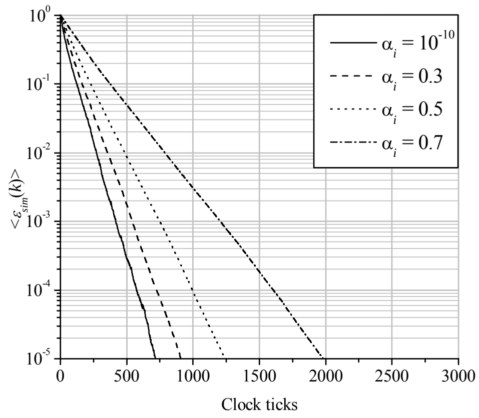

Fig. 7. Impact of different share factors when using law (4) in the broadcast-like algorithm for a fully-meshed network

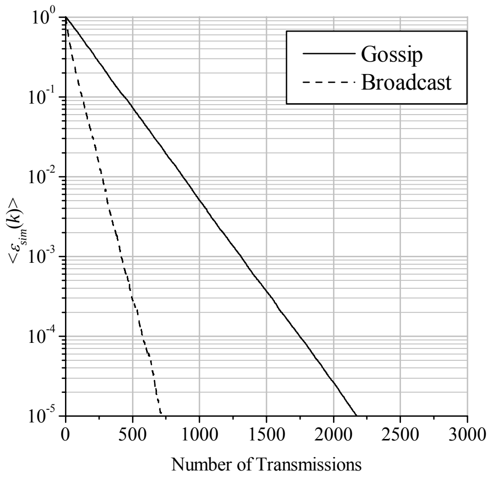

Fig. 8. Performance comparison between the broadcast-like algorithm and the gossip for a fully-meshed network

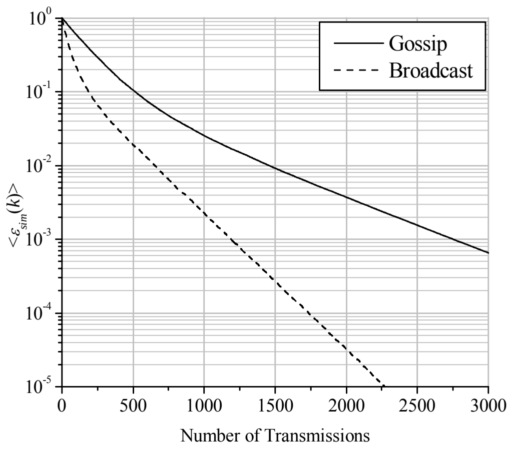

Fig. 9. Performance comparison between the broadcast-like algorithm and the gossip for a non fully-meshed network with $r = 0.4$

In Fig. 8 we compare the performance of the broadcast-like algorithm with that of the gossip algorithm, for the same graph of Fig. 5. As each clock tick in the gossip algorithm implies two transmissions, for a fair comparison, the average simulated curves have been plotted as functions of the number of transmissions, instead of the number of clock ticks. The figure confirms the advantage offered by the broadcast-like solution against the more conventional gossip one; for a given α_i , the broadcast-like algorithm requires about 1/3 of the number of transmissions needed by the gossip algorithm. In Fig. 9, the comparison is extended to the case of a non fully-meshed network with $r = 0.4$; the convergence is slower but the advantage offered by the broadcast-like algorithm is confirmed.

VI. CONCLUSIONS

Resorting to simulations is an important step to validate the conclusions of the analytical treatment and to study those situations that cannot be simply expressed in mathematical terms. In gossip algorithms, there are a lot of practical aspects that need a verification of this type; among them: the effect of optimizing the selection probabilities for each node and the impact of physical limitations as link failures or limited transmission radius. Focusing on some specific, though arbitrary, examples, we have analyzed the practical behavior of a gossip-based sensor network under different operational conditions. Based on our numerical evaluations, we can draw a number of conclusions.

First of all, we have seen that the lower and upper bounds to the averaging time are sensitive to the optimization process, as theoretically expected, and can be significantly improved through it. Nevertheless, our simulations demonstrate that there are cases of practical interest where to minimize the second eigenvalue, certainly effective as regards the bounds reduction, is not equally effective as regards lowering of the actual averaging time.

Moreover, we have verified that the gossip algorithm is fault tolerant, in the sense that convergence to the average value occurs even in the case of a high number of link failures. The averaging time, however, is significantly affected by link failures, as results from the comparison with non-fully meshed topologies. The absence of full connectivity reflects on a reduction of the rate of convergence, depending on the value of the transmission radius.

Another important issue concerns comparison between different versions of the gossip algorithm. In this paper we have considered a broadcast version of the classic unicast, though bidirectional, solution, and we have shown that it can permit significant improvements in the convergence rate.

Obviously, as based on a limited number of experiments, these conclusions cannot have a general meaning, but remain valid in many real cases, since referred to actual scenarios. Future research could adopt more complete network simulator tools in order to evaluate more complex network conditions and to consider networking issues in a thorough manner.

REFERENCES

[1] S. M. Hedetniemi, S. T. Hedetniemi, A. L. Liestman: *A Survey of Gossiping and Broadcasting in Communication Networks*, Networks, Vol. 18, pp. 319–349, 1988.

[2] D. Kempe, A. Dobra, J. Gehrke: *Gossip-Based Computation of Aggregate Information*, in Proc. IEEE Conference on Foundations of Computer Science, Cambridge, MA, Oct. 2003, pp. 482–491.

[3] J. -Y. Chen, D. X. G. Pandurangan, D. Xu: *Robust Aggregates Computation in Wireless Sensor Networks: Distributed Randomized Algorithms and Analysis*, in Proc. 2005 Fourth International Symposium on Information Processing in Sensor Networks, Los Angeles, CA, April 2005.

[4] C. C. Moallemi, B. van Roy: *Consensus Propagation*, IEEE Trans. Inf. Theory, Vol. 52, No. 11, pp. 1–13, 2006.

[5] M. Alanyali, V. Saligrama, O. Savas: *A Random-Walk Model for Distributed Computing in Energy-Limited Networks*, in Proc. First Workshop on Information Theory and its Applications, San Diego, CA, Feb. 2006.

[6] S. Boyd, A. Ghosh, B. Prabhakar, D. Shah: *Randomized Gossip Algorithms*, IEEE Trans. Inf. Theory, Vol. 52, No. 6, pp. 2508–2530, 2006.

[7] S. Boyd, A. Ghosh, B. Prabhakar, D. Shah: *Analysis and Optimization of Randomized Gossip Algorithms*, in Proc. IEEE Conf. Decision and Control, Nassau, Bahamas, Dec. 2004, pp. 5310–5315.

[8] S. Boyd, A. Ghosh, B. Prabhakar, D. Shah: *Gossip Algorithms: Design, Analysis, and Applications*, in Proc. IEEE INFOCOM, Miami, FL, Mar. 2005, Vol. 3, pp. 1653–1664.

[9] D. Kempe, J. Kleinberg, A. Demers: *Spatial Gossip and Resource Location Protocols*, in Proc. 33rd ACM Symp. Theory of Computing, 2001, Crete, Greece, July 2001, pp. 163–172.

[10] M. Mehyar, D. Spanos, J. Pongsajapan, S. H. Low, R. M. Murray: *Asynchronous Distributed Averaging on Communication Networks*, IEEE/ACM Trans. on Networking, Vol. 15, No. 3, pp. 512–520, 2007.

[11] R. Karp, C. Schindelhauer, S. Shenker, B. Vocking: *Randomized Rumor Spreading*, in Proc. IEEE Symposium on Foundations of Computer Science, Redondo Beach, CA, Nov. 2000, pp. 564–574.

[12] X. –Y. Li, K. Moaveninejad, O. Frieder: *Regional Gossip Routing for Wireless ad Hoc Networks*, in Proc. 28th Annual IEEE Int. Conf. on Local Computer Networks (LCN'03), Bonn, Germany, Oct. 2003.

[13] M. Hazewinkel: "Encyclopedia of Mathematics", Vol. 9, Springer, 1993.

[14] M. Penrose: "Random Geometric Graphs", Oxford studies in probability, Oxford: Oxford University Press, 2003.

[15] G. Dimakis, A. D. Sarwate, M. J. Wainwright: *Geographic Gossip: Efficient Averaging for Sensor Networks*, to appear in IEEE Trans. Signal Processing, available online: http://arxiv.org/abs/0709.3921, 25 Sep. 2007.

[16] M. Baldi, F. Chiaraluce. E. Zanaj: *Comparison of Averaging Algorithms for Wireless Sensor Networks*, in Proc. IEEE ICTTA'08, Damascus, Syria, April 2008.

Elma Zanaj was born in Valona (Albania) in 1980. She received her Laurea Degree in Electronics Engineering, with a specialization in Telecommunications, from the Polytechnic University of Tirana in 2003. From 2003 to 2005 she worked, as an assistant, within the group of Fundamentals in Electronics at the Polytechnic University of Tirana. Now she is a Ph. D. student at the Polytechnic University of Ancona, Italy. Her main research interests are in the field of the analysis and modeling of wireless sensor networks.

Marco Baldi was born in Macerata, Italy, in 1979. He received the "Laurea" degree (summa cum laude) in Electronics Engineering in 2003, and the Doctoral degree in Electronics, Informatics and Telecommunications Engineering in 2006 from the Polytechnic University of Marche, Ancona, Italy. At present, he is a post-doctoral researcher and contract Professor at the same university. His main research activity is in channel coding, with particular interest in linear block codes for symmetric and asymmetric channels, low-density parity-check (LDPC) codes and their application in cryptography.

Franco Chiaraluce was born in Ancona, Italy, in 1960. He received the "Laurea in Ingegneria Elettronica" (summa cum laude) from the University of Ancona in 1985. Since 1987 he joined the Department of Electronics and Automatics of the same university. At present, he is an Associate Professor at the Polytechnic University of Marche. His main research interests involve various aspects of communication systems theory and design, with special emphasis on error correcting codes, sensor networks, cryptography and multiple access techniques. He is co-author of more than 180 papers and two books. He is member of IEEE and IEICE.