Fast Decentralized Averaging via Multi-Scale Gossip

Konstantinos I. Tsianos[1] and Michael G. Rabbat[2]

[1] McGill University, Department of Electrical and Computer Engineering
Montreal, QC, Canada, konstantinos.tsianos@gmail.com
[2] McGill University, Department of Electrical and Computer Engineering
Montreal, QC, Canada, michael.rabbat@mcgill.ca

Abstract. We are interested in the problem of computing the average consensus in a distributed fashion on random geometric graphs. We describe a new algorithm called Multi-scale Gossip which employs a hierarchical decomposition of the graph to partition the computation into tractable sub-problems. Using only pairwise messages of fixed size that travel at most $O(n^{\frac{1}{3}})$ hops, our algorithm is robust and has communication cost of $O(n \log\log n \log \epsilon^{-1})$ transmissions, which is order-optimal up to the logarithmic factor in n. Simulated experiments verify the good expected performance on graphs of many thousands of nodes.

1 Introduction

Applications in sensor networks often demand that nodes cooperatively accomplish a task without centralized coordination. Autonomy is often equated with robustness and scalability in large-scale networked systems. This is especially true in wireless networks, where fundamental limits on spatial and temporal channel reuse limit the amount of communication possible at any instant in time. Moreover, when nodes are battery-powered—a typical design element of wireless sensor networks—each transmission consumes valuable energy resources. This has stimulated research into resource-efficient algorithms for distributed computing and coordination. Gossip algorithms are an attractive paradigm for decentralized, autonomous computation. A classic and well-studied example is gossiping to compute the average consensus: in a graph $G = (V, E)$ with $|V| = n$ nodes, where each node initially has a value $x_i(0)$, the goal is to compute an estimate of the average $x_{\text{ave}} = \frac{1}{n}\sum_{i=1}^{n} x_i(0)$ at all nodes. At each gossip iteration, a random connected subset $S(t) \subset V$ of nodes exchange their current estimates, $x_i(t)$, and locally compute the update $x_i(t+1) = \frac{1}{|S(t)|}\sum_{j\in S(t)} x_j(t)$. In the original algorithm, described in [1] and revisited in [2] and [3], $|S(t)| = 2$ at every iteration, and the pair of nodes that update are neighbors in G.

Gossip algorithms have the attractive properties that they run asynchronously, do not require any specialized routing protocols, and therefore do not create bottlenecks or single points of failure. In the original gossip algorithm, pairs of nodes that communicate directly exchange information at each iteration. Consequently,

no special routing is needed, and the algorithm has been shown to be robust to fluctuating availability of links, as well as other changes in topology.

Of course, robustness and autonomy come at a price. Standard pair-wise randomized gossip is inefficient on topologies commonly used to model connectivity in wireless networks, such as grids and random geometric graphs; the number of messages per node scales linearly with the size of the network. In contrast, one can imagine numerous other approaches to computing a linear combination of the values at each node, assuming the existence of some specialized routing such as a Hamilton cycle or spanning tree, in which the number of messages per node remains constant as the network size tends to infinity. Although the overhead involved in finding and maintaining these routes may be prohibitive, requiring more centralized control and creating bottlenecks and single points of failure, these observations have motivated substantial research to close the gap between the two scaling regimes.

Motivated by the robust scaling properties exhibited by other hierarchical systems, this paper describes a gossip algorithm that achieves nearly-optimal performance in wireless networks. We partition the network computation hierarchically in k different scales. At each scale, nodes gossip within their partition until convergence; then one representative is elected within each partition, and the representatives gossip. This is repeated, with representatives gossiping at each higher scale, until the representatives at the coarsest scale have computed x_{ave}. At that point the average is disseminated throughout the network.

The main contribution of this work is a new multi-scale gossip algorithm for which we prove that the average number of messages per node needed to reach average consensus in a grid or random geometric graph on n nodes, scales as $(k+1)n^{\delta(k)}$, where $\delta(k) \to 0$ as $k \to \infty$. Since larger graphs facilitate deeper hierarchies, we show that we can take $k = O(\log\log n)$ to obtain a scheme that asymptotically requires a number of messages per node proportional to $\log\log n$. In simulations of networks with thousands of nodes, two or three levels of hierarchy suffice to achieve state-of-the-art performance. This is comparable to existing state-of-the art gossip algorithms. Similar to the existing state-of-the-art, we assume the network implements geographic routing. This facilitates gossip iterations between pairs of representatives that may not communicate directly at higher levels of the hierarchy. However, unlike other fast gossip algorithms, we do not require many gossip exchanges among nodes. For example, the path averaging algorithm described in [4] gossips on average among $|S(t)| \propto \sqrt{n}$ nodes along a path at each iteration. In contrast, all gossip iterations in multi-scale gossip are between pairs of nodes, so if a message is lost in transit through the network, the amount of information lost is minimal. Moreover, the longest distance individual messages must travel is on the order of $n^{1/3}$ hops.

2 Previous Work and Known Results

Our primary measure of performance is *communication cost*—the number of messages (transmissions over a single hop) required to compute an estimate to

ϵ accuracy—which is also considered in [4, 5]. In the analysis of scaling laws for gossip algorithms, a commonly studied measure of convergence rate is the ϵ averaging time, denoted $T_\epsilon(n)$, which is the number of iterations required to reach an estimate with ϵ accuracy with high probability[3]. $T_\epsilon(n)$ reflects the idea that the complexity of gossiping on a particular class of network topologies should depend both on the final accuracy and the network size. When only neighbouring nodes communicate at each iteration, $T_\epsilon(n)$ and communication cost are identical up to a constant factor. Otherwise, communication cost can generally be bounded by the product of $T_\epsilon(n)$ and a bound on the number of messages required per iteration.

Kempe, Dobra, and Gehrke [2] initiated the study of scaling laws for gossip algorithms and showed that gossip requires $\Theta(n \log \epsilon^{-1})$ total messages to converge on complete graphs. Note that at least n messages are required to compute a function of n distributed data values in any network topology.

In wireless sensor network applications, *random geometric graphs* are a typical model for connectivity since communication is restricted to nearby nodes. In a 2-dimensional random geometric graph, n nodes are randomly assigned coordinates uniformly in unit square, and two nodes are connected with an edge when their Euclidean distance is less than or equal to a connectivity radius, r [6, 7]. In [6] it is shown that if the connectivity radius scales as $r_{\text{con}}(n) = \Theta(\sqrt{\frac{\log n}{n}})$ then the network is connected with high probability. Throughout this paper when we refer to a random geometric graph, we mean one with the connectivity $r_{\text{con}}(n)$. Boyd, Ghosh, Prabhakar, and Shah [3] studied scaling laws for pairwise randomized gossip on random geometric graphs and found that communication cost scales as $\Theta(\frac{n^2}{\log n} \log \epsilon^{-1})$ messages even if the algorithm is optimized with respect to the topology.

One approach for improving convergence rates is to introduce memory at each node, creating higher-order updates [8, 9]. However, the only known scaling laws for this approach are for a deterministic, synchronous variant of gossip [10], leading to $\Theta(\frac{n^{1.5}}{\sqrt{\log n}} \log \epsilon^{-1})$ communication cost. Gossip algorithms based on lifted Markov chains have been proposed that achieve similar scaling laws [11, 12].

A variant called *geographic gossip*, proposed by Dimakis, Sarwate, and Wainwright [5], achieves a similar communication cost of $\Theta(\frac{n^{1.5}}{\sqrt{\log n}} \log \epsilon^{-1})$ by allowing distant (non-neighbouring) pairs of nodes to gossip at each iteration. Assuming that each node knows its own coordinates and the coordinates of its neighbours in the unit square, communication between arbitrary pairs of nodes is made possible using *greedy geographic routing*. Rather than addressing nodes directly, a message is sent to a randomly chosen target (x, y)-location, and the recipient of the message is the node closest to that target. To reach the target, a message is forwarded from a node to its neighbour who is closest to the target. If a node is closer to the target than all of its neighbours, this is the final message recipient. It is shown in [5] that for random geometric graphs with connectivity radius

[3] A more rigorous definition is provided in the next section.

$r(n) = r_{\text{con}}(n)$, greedy geographic routing succeeds with high probability. For an alternative form of greedy geographic routing, which may be useful in implementations, see [13]. The main contribution of [5] is to illustrate that allowing nodes to gossip over multiple hops can lead to significant improvements in communication cost. In follow-up work, Benezit, Dimakis, Thiran, and Vetterli [4] showed that a modified version of geographic gossip, called *path averaging*, can achieve $\Theta(n \log \epsilon^{-1})$ communication cost on random geometric graphs. To do this, all nodes along the path from the source to the target participate in a gossip iteration. If geographic routing finds a path of nodes $S = \{x_i, \ldots, x_j\}$ to deliver a message from x_i to x_j, on the way to x_j values of nodes in S are accumulated. Then x_j computes the average of all S values and sends the average back down the same path towards x_i. All nodes along the way update their values.

The multi-scale approach considered in this paper also assumes that the network is capable of geographic routing in order to gossip among representative nodes at each scale. Below, we show that asymptotically, the communication complexity of multi-scale gossip is $O(n \log \log n \log \epsilon^{-1})$ messages, which is equivalent to that of path averaging up to a logarithmic factor. However, in multi-scale gossip, information is only exchanged between pairs of nodes, and there is no averaging along paths. We believe that this makes our algorithm more fault tolerant, since each message only carries the information for a pair of nodes. If an adversary wishes to disrupt gossip computation by forcing the network to drop a particular message or by deactivating a node in the middle of an iteration, a substantial amount of information can be lost in path averaging since each iteration involves $O(\sqrt{\frac{n}{\log n}})$ nodes on average. In addition, the longest distance a message travels in our multi-scale approach is $O(n^{1/3})$ hops in comparison to $O(n^{1/2})$ hops for geographic gossip or path averaging.

Finally, we note that we are not the first to propose gossiping in a multi-scale or hierarchical manner. Sarkar et al. [14] describe a hierarchical approach for computing aggregates, including the average. However, because their algorithm uses order and duplicate insensitive synopses to estimate the desired aggregate, the size of each message exchanged between a pair of nodes must scale with the size of the network. Other hierarchical distributed averaging schemes that have been proposed in the literature focus on the synchronous form of gossip, and they do not prove scaling laws for communication cost neither do they provide rules for forming the hierarchy (i.e. assume the hierarchical decomposition is given) [15–17].

3 Network Model and Problem Definition

Let $G = (V, E)$ be a random geometric graph [7] of n nodes with connectivity radius $r_{con}(n)$. Each node in G holds an initial real value $x_i(0)$. Vector $x(t) = [x_1(t)x_2(t)\ldots x_n(t)]^{\mathrm{T}}$ describes the values on the network at time t. Our goal is for nodes to communicate over graph edges to compute the average $x_{ave} = \frac{1}{n}\sum_{i=1}^{n} x_i(0)$. In the end, all nodes should have knowledge of x_{ave}. To measure performance of an averaging algorithm we use the following definition.

Definition 1. *ϵ averaging time $T_\epsilon(n)$. Given any desired accuracy $\epsilon > 0$, the ϵ averaging time is the earliest time at which vector $x(t)$ is ϵ close to the normalized true average with probability greater than $1 - \epsilon$:*

$$T_\epsilon(n) = \sup_{x(0)} \inf_{t=0,1,2,\ldots} \left\{ \mathbb{P}\left(\frac{||x(t) - x_{ave}||}{||x(0)||} \geq \epsilon \right) \leq \epsilon \right\} \quad (1)$$

Convergence rate generally depends on the initial values $x(0)$ and the definition assumes the worst possible starting point. In practice, to eliminate the possibility of favourable initial conditions when evaluating our algorithm, as initial condition we assign to each node the sum of its geographic coordinates.

Notice that time is measured in discrete time moments until convergence. As explained in [3] this facilitates the analysis of gossip algorithms but does not force the actual implementation to be sequential. Multiple communication events happen in parallel. Finally, for multi-scale gossip we use $T_\epsilon(n)$ since the consensus time characterization of convergence rate defined in [18] is not applicable. Specifically, our averaging scheme proceeds in phases and changes over time in a non-ergodic manner. Moreover, by definition our scheme stops after a finite number of iterations. The approach in [18] is only defined asymptotically as $t \to \infty$ and for averaging schemes that are ergodic.

4 Multi-scale Gossip

Multi-scale gossip performs averaging in a hierarchical manner. At each moment only nodes in the same level of hierarchy are doing computations at a local scale and computation at one level begins after the previous level has finished. By hierarchically decomposing the initial graph into subgraphs, we impose an order in the computation. As shown in the next section, for a specific decomposition it is possible to divide the overall work into a small number of linear sub-problems and thus obtain very close to linear complexity in the size of the network overall.

Assume a random geometric graph $G = (V, E)$ where each node knows its own coordinates in the unit square and the locations of its immediate neighbours. Each node also knows the total number of nodes in the network n, and k, the desired number of hierarchy levels[4]. Figure 1 illustrates an example with $k = 3$. At level 1 the unit square is split into m_1 small cells – denote them C_1 cells. The subgraphs G_1 of G involving nodes inside a single C_1 cell run standard randomized gossip until convergence. Then, each C_1 cell elects a representative node[5] L_1. The representative selection can be randomized or deterministic as explained in Section 7. Generally, not all G_1 graphs have the same number of nodes. For this reason, the value of each representative has to be reweighed proportionally to its graph size. At level 2, the unit square is split into C_2 cells.

[4] As explained in Section 6, given n, k can be computed automatically.

[5] Note that in level 1 as well as any other level d there are many C_d cells but for simplicity we avoid denoting them $C_{d,i}$ with i running from 1 to maximum number of cells. Similarly for representatives L_d.

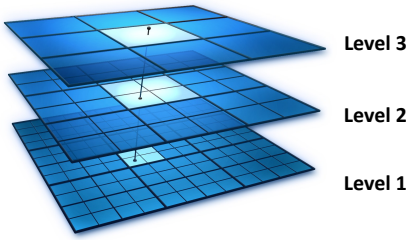

Fig. 1. Hierarchical multiscale subdivision of the unit square. At each level, each cell is split into equal numbers of smaller cells. Before the representatives of the cells can gossip on a grid graph, we run gossip on each cell.

Each C_2 contains the same number of C_1 cells. The representatives of the C_1 cells form grid graphs G_2 with two representatives $L_{1,i}$ and $L_{1,j}$ sharing an edge in a G_2 if cells $C_{1,i}$ and $C_{1,j}$ are adjacent and contained in the same C_2 cell. Note that representatives can determine which cells they are adjacent to given the current level of hierarchy and n. Next, we run randomized gossip simultaneously on all G_2 grid graphs. Finally, we select a representative node L_2 out of each G_2 and continue the next hierarchy level. The process repeats until we reach level k at which point we have only one grid graph G_k contained in the single cell C_k which coincides with the unit square. Once gossip on G_k is over, each representative L_{k-1} disseminates his final value to all the nodes in its cell.

Algorithm 1 describes multi-scale gossip in a recursive manner. The initial call to the algorithm has as arguments, the vector of initial node values (x_{init}), the unit square ($C = [0,1] \times [0,1]$), the network size n, the desired number of hierarchy levels k and the desired error tolerance tol. In a down-pass the unit square is split into smaller and smaller cells all the way to the C_1 cells. After gossiping in the G_1 graphs in *Line 15*, the representatives adjust their values (*Line 16*). As explained in the next section, if k is large enough, the G_1 are complete graphs. Since each node knows the locations of its immediate neighbours (needed for geographic routing), at level 1 it is easy to also compute the size of each G_1 which is needed for the reweighting. The up-pass begins with the L_1 representatives forming the G_2 grid graphs (*Line 8*) and then running gossip in all of them in parallel.. We use a parameter $a = \frac{2}{3}$ to decide how many C_{d-1} cells fit in each C_d cell. The motivation for this parameter and its specific value is explained in the following section. Notice the pseudocode mimics a sequential single processor execution which is in line with the analysis that follows in Section 6. However, it should be emphasized that the algorithm is intended for and can be implemented in a distributed fashion. The notation $x_{init}(C)$ or $x_{init}(L)$

Algorithm 1 MultiscaleGossip(x_{init}, C, n, k, tol)

1: $a = \frac{2}{3}$
2: **if** $k > 1$ **then**
3: Split C into $m_{k-1} = n^{1-a}$ cells: $C_{k-1,1}, \ldots, C_{k-1,m_{k-1}}$
4: Select a representative node $L_{k-1,i}$ for each cell $C_{k-1,i}, i \in \{1, \ldots, m_{k-1}\}$
5: **for all** cells $C_{k-1,i}$ **do**
6: **call** *HierarchicalGossip*($x_{init}(C_{k-1,i}), C_{k-1,i}, n^a, k-1, tol$)
7: **end for**
8: Form grid graph G_{k-1} of representatives $L_{k-1,i}$
9: **call** *RandomizedGossip*($x_{init}(L_{k-1,1:m_{k-1}}), G_{k-1}, tol$)
10: **if** at top level **then**
11: Spread value of $L_{k,i}$ to all nodes in $C_{k,i}$
12: **end if**
13: **else**
14: Form graph G_1 only of nodes in $V(G)$ contained in C
15: **call** *RandomizedGossip*(x_{init}, G_1, tol)
16: Reweight representative values as : $x(L_{1,i}) = x(L_{1,i}) \frac{|V(G_1)| \cdot m_2}{|V(G)|}$
17: **end if**

indicates that we only select the entries of x_{init} corresponding to nodes in cell C or representatives L.

The ideal scenario for multi-scale gossip is if computation inside each cell stops automatically when the desired accuracy is reached. This way no messages are wasted. However in practice cells at the same level we may need to gossip on graphs of different sizes that take different numbers of messages to converge. This creates a need for node synchronization so that all computation in one level is finished before the next level can begin. To overcome the need for synchronization, we can fix the number of randomized gossip iterations per level. Given that nodes are deployed uniformly at random in the unit square, we can make a worst case estimate of how many nodes are expected to be in a cell of a certain area. Since by construction all cells at the same level have equal area, we gossip on all graphs at that level for a fixed number of iterations. Usually, at level 1, we have less nodes than expected so we end up wasting messages running gossip for longer than necessary.

5 Evaluation of Multi-scale Gossip

Before proceeding with the formal analysis of the algorithm complexity, we show in this section that Multi-scale Gossip performs very well against Path Averaging [4], a recent state-of-the-art gossip algorithm that requires linear number messages in the size of the network to converge to the average with ϵ accuracy. Figure 2 (left) shows the number of messages needed to converge within $\epsilon = 0.0001$ error for graphs of sizes 500 to 8000. The bottom curve tagged *MultiscaleGossip* shows the ideal case where computation inside each cell stops automatically when the desired accuracy is reached. The curve tagged *MultiscaleGossipFI* was gener-

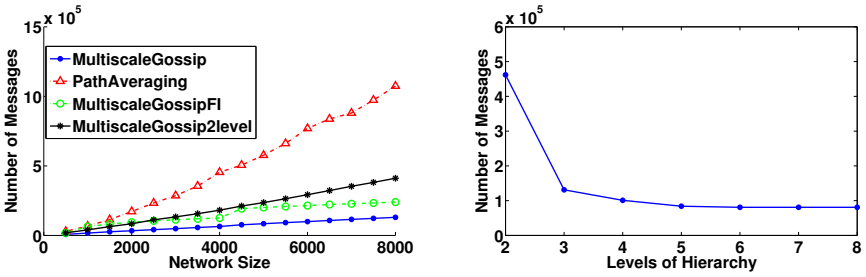

Fig. 2. (left) Comparison of MultiscaleGossip to PathAveraging. Total number of messages to converge with $\epsilon = 0.0001$ accuracy on random geometric graphs of increasing sizes. Results are averages over 20 runs. MultiscaleGossip used with 5 levels of hierarchy. MultiscaleGossipFI is the version using a fixed number of iterations for gossiping at a specific level. MultiscaleGossip2level is a version using only two levels of Hierarchy and is explained in Section 7. (right) Increasing the levels of Hierarchy yields a diminishing reward. Results are averages over 10 random geometric graphs with 5000 nodes and final desired accuracy $\epsilon = 0.0001$. All graphs are created with radius $r = \sqrt{\frac{3 \log n}{n}}$.

ated using fixed number of iterations per level based on worst case graph sizes as explained in Section 4. One reason why path averaging seems to be slower is because we use a smaller connectivity radius for our graphs ($r = \sqrt{\frac{3 \log n}{n}}$ instead of $r = \sqrt{\frac{10 \log n}{n}}$ which is described in [4]).

Multi-scale Gossip has several advantages over Path Averaging. All the information relies on pairwise messages. In contrast, averaging over paths of length more than two has two main disadvantages. First, if a message is lost, a large number of nodes (potentially $O(\sqrt{\frac{n}{\log n}})$) are affected by the information loss. Second, when messages are sent to a remote location over many hops, they increase in size as the message body accumulates the information of all the intermediate nodes. Besides being variable, the message size now depends on the length of the path and ultimately on the network size. Our messages are always of constant size and independent of the hop distance or network size. Moreover, as will be shown in the next section, the maximum number of hops any message has to travel is $O(n^{\frac{1}{3}})$ at worst. This should be compared to distance $O(\sqrt{n})$ which is necessary for Path Averaging to achieve linear scaling. Finally, Multi-scale Gossip is relatively easy to analyze and implement using standard randomized gossip as a building block for the averaging computations.

6 Analysis of Multi-scale Gossip

The motivation behind multi-scale gossip is to divide the computation into stages each of which takes no more than linear number of messages in the size of

the network. This allows the overall algorithm to scale very close to linear as established by the following theorem:

Theorem 1. *Let a random geometric graph G of size n and an $\epsilon > 0$ be given. As the graph size $n \to \infty$, the communication cost of the multi-scale gossip scheme described above with scaling constant $\alpha = \frac{2}{3}$ behaves as follows:*

1. *If the number of hierarchy levels k remains fixed as $n \to \infty$, then the communication cost of multi-scale gossip is $O\big((kn + n^{1+(\frac{2}{3})^k}) \log \epsilon^{-1}\big)$ messages.*
2. *If $k = \Theta(\log \log n)$, then the communication cost of multi-scale gossip is $O(n \log \log m \log \epsilon^{-1})$ messages.*

Proof. Suppose we run Multi-scale Gossip (Algorithm 1) on a random geometric graph $G = (V, E)$ with $|V| = n$ and connecting radius is $r(n) = \sqrt{\frac{c \log n}{n}}$. Call the unit square cell C_k for a total of k hierarchy levels. At the highest level, we split the unit square into $m_{k-1} = n^{1-a}$ cells $C_{k-1,i}$ each of dimensions $n^{\frac{a-1}{2}} \times n^{\frac{a-1}{2}}$ where $a = \frac{2}{3}$ as explained below. On a grid of p nodes, randomized gossip requires $O(p^2)$ or by including the dependence on final accuracy, $O(p^2 \log \epsilon^{-1})$ messages to converge (e.g. see [3]). The grid graph G_{k-1} formed by the representatives of the $C_{k-1,i}$ cells has n^{1-a} nodes. Moreover, for appropriately large c (e.g. $c = 3$), graph G is *geo-dense* [19] and a patch of area n^{a-1} is expected to have $\Theta(n^{a-1}n) = \Theta(n^a)$ nodes in it. The maximum distance between two representatives in G_{k-1} will be $\sqrt{5}n^{\frac{a-1}{2}} = O(n^{\frac{a-1}{2}})$. If we divide by $r(n)$, we get a worst case estimate of the cost for multi-hop messages between representatives: $MsgCost_{k-1} = O(n^{\frac{a}{2}})$. Now the total number of pairwise messages on G_{k-1} will be $O((n^{1-a})^2 \log \epsilon^{-1}) \cdot O(n^{\frac{a}{2}})$. This number is $O(n)$ if $a = \frac{2}{3}$.

At the next level of hierarchy, we subdivide each $C_{k-1,i}$ cell of $q = \Theta(n^a)$ nodes into $q^{1-a'}$ cells in a recursive manner. Each $C_{k-2,i}$ cell will have dimensions $q^{\frac{a'-1}{2}} \times q^{\frac{a'-1}{2}}$. Using the exact same analysis as above, we will have n^{1-a} grid graphs GG_{k-2} and each has $q^{1-a'}$ (representative) nodes. The communication cost between $L_{k-2,i}$ representatives is $MsgCost_{k-2} = O(q^{\frac{a'}{2}})$. To make the total number of messages at level $k-2$ linear, we get $n^{1-a} \cdot O((q^{1-a'})^2 \log \epsilon^{-1}) \cdot O(q^{\frac{a'}{2}}) = n \Rightarrow a' = \frac{2}{3}$ as well. A simple induction proves that in general a subdivision of each cell of size q into q^{1-a} subcells yields linear performance for that level if $a = \frac{2}{3}$. Finally, after k levels, the algorithm runs gossip on each subgraph of G with nodes contained inside each of the C_1 cells. We have $O(n^{1-(\frac{2}{3})^k})$ C_1 cells, each containing $n^{(\frac{2}{3})^k}$ nodes. Since we run randomized gossip on each subgraph, the total number of messages at the last level is $O(n^{1+(\frac{2}{3})^k})$. Summing up all levels, plus n messages to spread the final result back to all nodes,the total number of messages for Multi-scale Gossip is $O\big((kn + n^{1+(\frac{2}{3})^k}) \log \epsilon^{-1}) + n\big) = O\big((kn + n^{1+(\frac{2}{3})^k}) \log \epsilon^{-1})\big)$.

For the second part of the theorem, observe that at level 1 each cell contains a subgraph of $n^{(\frac{2}{3})^k}$ nodes in expectation. We can choose k so that each cell contains at least $m \geq 2$ nodes for randomized gossip to be non-trivial and no

more than $M \geq m$ nodes so that the cost per cell is bounded by $M^2 \log \epsilon^{-1}$. In other words, choose k such that:

$$m \leq n^{(\frac{2}{3})^k} \leq M \Rightarrow \frac{\log\log M - \log\log n}{\log \frac{2}{3}} \leq k \leq \frac{\log\log m - \log\log n}{\log \frac{2}{3}}$$

Since the cost per level 1 cell is now bounded by a constant for $k = \Theta(\log\log n)$, the total level 1 cost is $O(n^{1-(\frac{2}{3})^k} \log \epsilon^{-1})$ and the overall cost is $O\big((kn + n^{1-(\frac{2}{3})^k}) \log \epsilon^{-1}) + n\big) = O(n \log\log n \epsilon^{-1})$ □

In practice we only need a few levels of hierarchy. Figure 2 (right) investigates the effect of increasing the levels of hierarchy. The figure shows the number of messages until convergence within 0.0001 error, averaged over ten graphs of 5000 nodes. More levels yield a diminishing reward and we don't need more than 4 or 5 levels. As discussed in Section 7 these observation lead us to try a scheme with only two levels of hierarchy which still produces an efficient algorithm.

7 Practical Considerations

There is a number of practical considerations that we would like to bring to the reader's attention. We list them in the form of questions below:

Does Multi-scale gossip computation scheme affect the final error? This is a valid concern since our algorithm essentially uses randomized gossip as a lossy averaging operator over subsets of the network nodes. At each level the representatives trust the ϵ approximate values of the previous level. Fortunately the error deteriorates only linearly with the number of levels. If $ave(\cdot)$ is an ϵ accuracy averaging operator, $ave(a_1, \ldots . a_m) = \overline{a} \pm \epsilon$. At the next level, $ave(\overline{a}_1 \pm \epsilon, \ldots, \overline{a}_m \pm \epsilon) = \overline{a} \pm \epsilon \pm \epsilon$. After k levels the final error will be at worst $\pm k\epsilon$.

How can we detect convergence in a subgraph or cluster? Do the nodes need to be synchronized? At each hierarchy level, representatives know how big the grid that they are gossiping over is (function of n and k only). Moreover, all grids at the same level are of the same size and we have tight bounds on the number of messages needed to obtain ϵ accuracy on grids w.h.p. We can thus gossip on all grids for a fixed number or rounds and synchronization is implicit. At level 1 however, in general we need to gossip on random geometric subgraphs which are not of exactly the same size. As n gets large though, random geometric graphs tend to become regular and uniformly spaced on the unit square. Therefore, the subgraphs contained in cells at level 1 all have sizes very close to the expected value of $n^{(\frac{2}{3})^k}$. Thus, we run gossip for a fixed number of rounds using the theoretical bound for graphs of the size $n^{(\frac{2}{3})^k}$. As discussed in Section 4, fixing the number of iterations leads to redundant transmissions, however the algorithm is still very efficient.

What happens with disconnected subgraphs or grids due to empty grid cells? Technically this is possible since the division of the unit square into grid cells does not mean that each cell is guaranteed to contain any nodes of the initial graph. Representatives use multi-hop communication and connected grids can always be constructed as long as the initial random geometric graph is connected. At level 1 the subgraphs of the initial graph contained in each cell could still be disconnected if edges that go outside the cell are not allowed. However, as explained in Section 6 we can use enough hierarchy levels so that each C_1 cell is a complete graph and the probability of getting disconnected C_1 cells tends to zero.

How can we select representatives in a natural way? The easiest solution is to pick the point p_c that is geographically at the centre of each cell. Again, knowledge of n, k uniquely identifies the position of each cell and also p_c. By sending all messages to p_c, geographic routing will deliver them to the unique node that is closest to that location w.h.p. To change representatives, we can deterministically pick a location $p_c + u$ which will cause a new node to be the closest to that location. A more sophisticated solution would be to employ a randomized auction mechanism. Each node in a cell generates a random number and the largest number is the representative. Once a new message enters a cell, the nodes knowing their neighbours values, route the message to the cell representative. Notice that determining cell leaders this way does not incur more than linear cost.

Are representatives bottlenecks and single points of failure? This is not an issue. There might be a small imbalance in the amount of work done by each node, but it can be alleviated by selecting different representatives at each hierarchy level. Moreover, for increased robustness, at a linear cost we can disseminate the representative's values to all the nodes in its cell. This way if a representative dies, another node in the cell can take its place. The new representative will have a value very similar (within ϵ) to that of the initial representative at the beginning of the computation at the current level. Thus node failure is expected to only cause small delay in convergence at that level. We should emphasize however that the effect of node failures has received little attention so far and still asks for a more systematic investigation.

How much extra energy do the representatives need to spend? This question is hard to answer analytically. We use simulation to get a feel for it. Figure 3 shows the number of messages sent by each of the 5000 nodes in a random geometric graph. For this case we used five levels of hierarchy. The first 3200 nodes were representatives at some point in the computation and the rest only participated in gossiping at level 1. As we go down the hierarchy the cells get smaller and the options for representatives are less so it is expected that some nodes will be doing more computation. Here the average number of messages per representative is about 16 with a standard deviation of 14. However, only less than 10% of the representative nodes send more than 30 messages. For

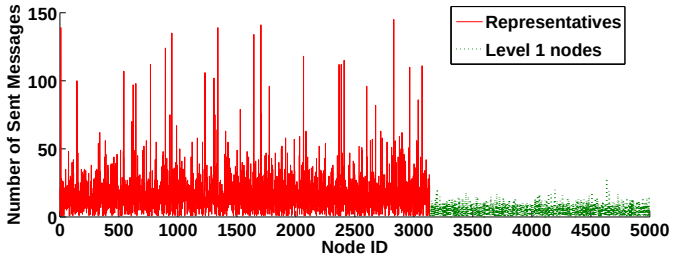

Fig. 3. Node utilization on a random geometric graphs with 5000 nodes and final desired accuracy $\epsilon = 0.0001$. For each node plot the total number of sent messages. Nodes 0 to 3200 have had at least one representative node role and the rest only participated at level 1. Both types of nodes may have participated as indermediates in multi-hop communication as well.

the level 1, nodes have an average of 5 messages with standard deviation of 3.

Don't many levels of hierarchy make it harder to implement and keep the nodes synchronized? This is a valid concern. Given our observations in Section 6 we tried an algorithm with just 2 levels of hierarchy. In this case, for graphs of size a few thousand nodes, splitting the unit square into n^{1-a} cells with $a = \frac{2}{3}$ is not a good choice as it produces a very small grid of representatives and quite large level 1 cells. To achieve better load balancing between the two levels, we use $a = \frac{1}{2}$. This choice has the advantage that the maximum number of hops any message has to travel is $O(n^{\frac{1}{4}})$. To see this, observe that each cell C_1 has area $\frac{1}{n^a} = n^{-\frac{1}{2}} = n^{-\frac{1}{4}} \times n^{-\frac{1}{4}}$. Thus the maximum distance between representatves is $O(n^{-\frac{1}{4}})$. If we divide by the connecting radius $r(n) = \sqrt{\frac{c \log n}{n}}$ we get the result. Another interesting finding is that for moderate sized graphs, using cells of area $n^{-\frac{1}{2}}$ produces subgraphs which are very well connected. Since nodes are deployed uniformly at random, an area $n^{-\frac{1}{2}}$ is expected to contain $n^{\frac{1}{2}}$ nodes. A subgraph inside a C_1 cell is still a random geometric graph with $t = n^{\frac{1}{2}}$ nodes, but for which the radius used to connect nodes is not $\sqrt{\frac{c \log t}{t}}$. It is $\sqrt{\frac{c \log n}{n}}$. This is equivalent to creating a random geometric graph of t nodes in the unit square but with a scaled up radius of $r_t = \sqrt{\frac{c \log n}{t}}$. From [19] we know that a random geometric graph of t nodes is rapidly mixing (i.e. linear number of messages for convergence) if the connecting radius is $r_{rapid} = \frac{1}{poly(\log t)}$. Now, e.g. for $c = 3$ and $n \leq 9 * 10^6$, we get $r_t \geq \frac{1}{\log t} \geq r_{rapid}$ for $t = \sqrt{n} \leq 3000$. Consequently, the C_1 cells are rapidly mixing for networks of less than a few millions of sensors. In Figure 2 verifies this analysis. For graphs from 500 to 8000 nodes and final error 0.0001, we see that *MultiscaleGossip2level* performs

very close to Multi-scale Gossip with more levels of hierarchy and better than path averaging.

8 Discussion and Future Work

We have presented a new algorithm for distributed averaging exploiting hierarchical computation. Multi-scale gossip separates the computation in linear phases and achieves very close to linear complexity overall ($O((k+1)n^{1+\delta(k)})$). Moreover, the maximum distance any message has to travel is $O(n^{\frac{1}{3}})$ as opposed to $O(\sqrt{n})$ needed by path averaging which is the other existing gossip algorithm with linear complexity. Finally, multi-scale gossip uses fixed size messages independent of the graph size and does not rely on longer that pairwise averaging operations. There is a number of interesting future directions that we see. In our present description, computation happens on grids which are known to require quadratic number of messages. Since these grids use multi-hop communication anyway, it might be possible to further increase performance by devising other overlay graphs between representatives with better convergence properties, i.e. expander graphs [20]. Moreover, the subdivision of the unit square into grid cells is not necessarily natural with respect to the topology of the graph. One could use other methods for clustering. So far, we have some preliminary results with spectral clustering which seem promising in simulation. It is however not clear how to do spectral clustering in a distributed way and in linear number of messages. Another idea is to combine the multi-scale approach with the use of more memory at each node to get faster mixing rates. Notice however that how to use memory to provably accelerate asynchronous gossip is still an open question. Current results only look at synchronous algorithms [10]. Finally, an important advantage of gossip algorithms in general is their robustness. Intuitively this is expected. However the question of modelling and reacting to node failures has not been formally investigated in the literature. It would be very interesting to introduce failures and see the effect on performance for different gossip algorithms.

Acknoledgements

We would like to thank Marius Şucan for providing the multiscale grid figure in Section 4.

References

1. Tsitsiklis, J.: Problems in Decentralized Decision Making and Computation. PhD thesis, Massachusetts Institute of Tech. (Nov. 1984)
2. Kempe, D., Dobra, A., Gehrke, J.: Computing aggregate information using gossip. In: Proc. Foundations of Computer Science, Cambridge, MA (Oct. 2003)
3. Boyd, S., Ghosh, A., Prabhakar, B., Shah, D.: Randomized gossip algorithms. IEEE Trans. Inf. Theory **52**(6) (Jun. 2006) 2508–2530

4. Benezit, F., Dimakis, A., Thiran, P., Vetterli, M.: Gossip along the way: Order-optimal consensus through randomized path averaging. In: Proc. Allerton Conf. on Comm., Control, and Comp., Urbana-Champaign, IL (Sep. 2007)
5. Dimakis, A., Sarwate, A., Wainwright, M.: Geographic gossip: Efficient averaging for sensor networks. IEEE Trans. Signal Processing **56**(3) (Mar. 2008) 1205–1216
6. Gupta, P., Kumar, P.R.: Critical power for asymptotic connectivity in wireless networks. In: Stochastic Analysis, Control, Optimization, and Applications, Boston (1998) 1106–1110
7. Penrose, M.: Random Geometric Graphs. Oxford University Press (2003)
8. Cao, M., Spielman, D.A., Yeh, E.M.: Accelerated gossip algorithms for distributed computation. In: Proc. 44th Annual Allerton Conf. Comm., Control, and Comp., Monticello, IL (Sep. 2006)
9. Kokiopoulou, E., Frossard, P.: Polynomial filtering for fast convergence in distributed consensus. IEEE Trans. Signal Processing **57**(1) (Jan. 2009) 342–354
10. Oreshkin, B., Coates, M., Rabbat, M.: Optimization and analysis of distributed averaging with short node memory. To appear *IEEE Trans. Signal Processing* (2010)
11. Li, W., Dai, H.: Location-aided fast distributed consensus. In: IEEE Transactions on Information Theory, submitted. (2008)
12. Jung, K., Shah, D., Shin, J.: Fast gossip through lifted Markov chains. In: Proc. Allerton Conf. on Comm., Control, and Comp., Urbana-Champaign, IL (Sep. 2007)
13. Sarkar, R., Yin, X., Gao, J., Luo, F., Gu, X.D.: Greedy routing with guaranteed delivery using ricci flows. In: Proc. Information Processing in Sensor Networks, San Francisco (April 2009)
14. Sarkar, R., Zhu, X., Gao, J.: Hierarchical spatial gossip for multi-resolution representations in sensor networks. In: Proc. of the International Conference on Information Processing in Sensor Networks (IPSN'07). (April 2007) 420–429
15. Kim, J.H., West, M., Lall, S., Scholte, E., Banaszuk, A.: Stochastic multiscale approaches to consensus problems. In: Proc. IEEE Conf. on Decision and Control, Cancun (Dec. 2008)
16. Epstein, M., Lynch, K., Johansson, K., Murray, R.: Using hierarchical decomposition to speed up average consensus. In: Proc. IFAC World Congress, Seoul (Jul. 2008)
17. Cattivelli, F., Sayed, A.: Hierarchical diffusion algorithms for distributed estimation. In: Proc. IEEE Workshop on Statistical Signal Processing, Wales (Aug. 2009)
18. Denantes, P., Benezit, F., Thiran, P., Vetterli, M.: Which distributed averaging algorithm should i choose for my sensor network? In: Proc. IEEE Infocom, Phoenix (Apr. 2008)
19. Avin, C., Ercal, G.: On the cover time and mixing time of random geometric graphs. Theoretical Computer Science **380** (June 2007) 2–22
20. Margulis, G.: Explicit group-theoretical constructions of combinatorial schemes and their application to the design of expanders and concentrators. J. Probl. Inf. Transm. **24**(1) (1988) 39–46