Randomized Sequential Algorithms for Data Aggregation in Sensor Networks

Onur Savas, Murat Alanyali, Venkatesh Saligrama
Department of Electrical and Computer Engineering
Boston University, Boston, MA 02215, USA
Email: {savas,alanyali,srv}@bu.edu

***Abstract*— We consider distributed algorithms for data aggregation in sensor networks. The algorithms are admitted via message passing hence through pairwise computations. Under these algorithms a transmitting node becomes inactive until it is reactivated, yielding to substantial energy gains. We study the per-node message and time complexities of the algorithms in explicit graphs as a function of the network size. The obtained complexities are compared to those of so-called gossip algorithms. The results suggest trade-offs between the number of messages transmitted and the computation times.**

I. Introduction

We consider distributed algorithms to compute a class of functions that admit flexible decomposition in terms of pairwise computations. These functions include in particular maximum, product and weighted averages, which arise in sensor applications. Rather than optimal centralized algorithms we consider robust online algorithms that require only local information at each sensor. The main theme of this work is to quantify the number of message transmissions as proxy to energy consumption, as well as the time required to compute a given function.

Similar aspects of optimal centralized algorithms were considered by [6], [7]. Decentralized local message-passing algorithms have been previously proposed to compute several functions considered in the present paper [1], [2], [4], [8], [12], [13]. These algorithms are based on parallel asynchronous transmissions of local estimates that lead to improved estimates at recipient nodes. Although they possess desirable operational features, convergence of these algorithms relies in essence on the mixing properties of the connectivity matrix of communication graph G and thereby on substantial number of message transmissions.

We consider two randomized sequential algorithms that lead to striking gains in energy consumption while enjoying similar operational advantages as previously studied algorithms. Under the algorithms studied here, a transmitting node becomes inactive and does not transmit further messages until it is reactivated by a message reception from another node. An active node generates messages at constant rate and sends each message to a randomly selected neighbor. Algorithm SIMPLE-WALK maintains a single active node, whose trajectory is a random walk on the communication graph. Local processing at each active node exploits the decomposability of the considered functions to guarantee that the value of interest is computed when each node becomes active at least once. Both the time and the message complexities of this algorithm are determined by the cover time [10] of the random walk. Under algorithm COALESCENT all nodes are initially active and the computation is completed when a single active remains in the network. This latter algorithm is closely related with coalescing random walk [5]. It is arguable that COALESCENT expedites the computation while SIMPLE-WALK terminates with fewer message transmissions. We quantify this tradeoff on several topologies and provide a comparison with gossip algorithms.

Section II gives the problem definition and Section III formalizes the studied algorithms. Section IV studies the time and message complexities of these algorithms on explicit graphs and these results are compared to those of gossip algorithms in Section V. The numerical results are presented in Section VI with emphasis on geometric random graphs. The paper concludes with final remarks in Section VII.

II. Problem Definition

Let Λ be a set and let $f : \Lambda^2 \mapsto \Lambda$ be an operation on Λ. We shall assume that f is commutative and associative, and that Λ has an identity element with respect to f. That is, there exists $e \in \Lambda$ such that $f(\lambda, e) = \lambda$ for $\lambda \in \Lambda$. Let $F_2 = f$. For $n \geq 3$ and given a permutation of $\{1, 2, \cdots, n\}$ by $\pi = \{\pi_1, \pi_2, \cdots, \pi_n\}$ define the mapping $F_n : \Lambda^n \mapsto \Lambda$ recursively by

$$F_n(\lambda_1, \lambda_2, \cdots, \lambda_n) = f(\lambda_{\pi_1}, F_{n-1}(\lambda_{\pi_2}, \lambda_{\pi_3}, \cdots, \lambda_{\pi_n})), \tag{1}$$

for $\lambda_1, \lambda_2, \cdots, \lambda_n \in \Lambda$. Note that the mapping F_n does not depend on π owing to the properties of f.

Example 2.1: Let $k \geq 1$ be an integer and let $M^{k\times k}$ be the set of k-dimensional positive definite matrices. For $i = 1, 2, \cdots, n$ let $x_i \in \mathbb{R}^k$ and let $w_i \in M^{k\times k}$. The weighted average $(\sum_{i=1}^n w_i)^{-1} \sum_{i=1}^n w_i x_i$ can be expressed in the form (1) by choosing $\Lambda = \{\mathbb{R}^k \times M^{k\times k}\} \cup \{e\}$ with $\lambda_i = (x_i, w_i)$, and by setting

$$f(\lambda, \lambda') = ((w + w')^{-1}(wx + w'x') \ , \ w + w'),$$

for $\lambda = (x, w), \lambda' = (x', w') \in \Lambda - \{e\}$.

Consider a sensor network of n nodes modelled by a graph $G = (V, E)$ where each $v \in V$ is a sensor node (and $|V| = n$) and each $(i, j) \in E$ represents a bidirectional

communication link between sensor nodes. We are concerned with the distributed computation of $F_n(\lambda_1, \lambda_2, \cdots, \lambda_n)$ where each λ_i is known to distinct sensor nodes. We assume G is connected in order to avoid trivialities.

III. Algorithms

We consider two different algorithms, namely `SIMPLE-WALK` and `COALESCENT` based on well-studied simple and coalescing random walks, respectively. A pseudo-code for the algorithms is given in Figure 1. Under each algorithm, each node maintains two variables `value` and `status`. Content of `value` takes values in the set Λ and `status` is either *'active'* or *'idle'*. We identify these variables by $v_i(t) \in \Lambda$ and $\xi_i(t) \in \{0, 1\}$ at node $i \in V$ at time $t \geq 0$; in particular

$$v_i(t) = \text{the content of } \texttt{value} \text{ of node } i \text{ at time } t,$$
$$\xi_i(t) = \begin{cases} 1 & \text{if } \texttt{status} \text{ of node } i \text{ is 'active' at time } t, \\ 0 & \text{else.} \end{cases}$$

Initially $v_i(0) = \lambda_i$ for each node i. The initial value $\xi_i(0)$ (i.e. of `status`) depends on the particular algorithm in the following fashion: Under `SIMPLE-WALK` $\xi_i(0) = 1$ for exactly one node, say node i_o, whereas $\xi_i(0) = 1$ for all nodes i under `COALESCENT`. Variables `value, status` evolve according to the same rules under both algorithms: Namely, each node has an independent Poisson clock that ticks at unit rate. When the local clock of node i ticks, say at time t_o, the node does not take any action unless it is active (i.e. $\xi_i(t_o^-) = 1$). Otherwise, if $\xi_i(t_o^-) = 1$, the node chooses a neighbor at random, sends its current value $v_i(t_o^-)$ to that neighbor, and sets $(v_i(t_o), \xi_i(t_o)) = (e, 0)$. In particular it becomes idle and ceases message transmission until it becomes active again. The selected neighbor, say node j, sets $v_j(t_o) = f(v_j(t_o^-), v_i(t_o^-))$ and becomes active by setting $\xi_j(t_o) = 1$.

In both algorithms an active node can be interpreted as holding a token and this node passes it to a randomly chosen neighbor at its clock tick changing. Under `COALESCENT` two tokens coalesce into one when they occupy the same vertex and act as one thereafter. Under `SINGLE-WALK` there is only one token and it passes through one node to another. An illustration of the evolution of tokens of `COALESCENT` is given in Figure 2.

Both algorithms compute the value of $F_n(\lambda_1, \lambda_2, \cdots, \lambda_n)$ correctly regardless of the paths chosen until time $t > 0$ as shown in the next proposition.

Proposition 3.1: Under both `SIMPLE-WALK` and `COALESCENT`,

$$F_n(v_1(t), v_2(t), \cdots, v_n(t)) = F_n(\lambda_1, \lambda_2, \cdots, \lambda_n), \quad t > 0.$$

Let τ_S and τ_C be random times that are defined as follows:

$$\tau_S = \inf\{t : \text{ each node becomes active by time } t\ \}$$
$$\tau_C = \inf\{t : \sum_i \xi_i(t) = 1\}.$$

Variables: `status, value`.
Initialize: `value` $\leftarrow \lambda_i$;

$$\texttt{SIMPLE}-\texttt{WALK}: \quad \texttt{status} \leftarrow \begin{cases} \text{'active'} & \text{if } i = i_o; \\ \text{'idle'} & \text{else.} \end{cases}$$
$$\texttt{COALESCENT}: \quad \texttt{status} \leftarrow \text{'active'}.$$

Procedure **Send**()
 if(`status` == 'active') {
  choose neighbor;
  send(neighbor , `value`);
  `value` $\leftarrow e$;
  `status` $\leftarrow$ 'idle';
  }
Procedure **Receive**(message) {
 `value` $\leftarrow f($`value`, message$)$;
 `status` $\leftarrow$ 'active';
 }

Fig. 1. Pseudo-code for algorithms `SIMPLE-WALK` and `COALESCENT` at node i. Send() is activated by the local Poisson clock at the node, and Receive() is activated by message reception from another node. The two algorithms differ in the initialization of the variable `status`. That is, only one node starts out active in `SIMPLE-WALK` whereas all nodes are initially active in `COALESCENT`.

Hence for $t > \tau_C$ (and for $t > \tau_S$) such that for one single node $i(t) \in V$ we have

$$v_i(t) = \begin{cases} F_n(\lambda_1, \lambda_2, \cdots, \lambda_n) & \text{if } i = i(t) \\ e & \text{else} \end{cases}$$
$$\xi_i(t) = \begin{cases} 1 & i = i(t) \\ 0 & \text{else.} \end{cases}$$

Therefore τ_S and τ_C are stopping times for `SIMPLE-WALK` and `COALESCENT`, respectively in the sense that $F_n(\lambda_1, \lambda_2, \cdots, \lambda_n)$ is computed by τ_S (and τ_C) and is known to a single active node in the network. If n is known in advance a counter can be attached to the tokens and the algorithms are terminated according to that counter.

IV. Time and Message Complexities

In this section we study the messages and time complexities of the considered algorithms. We begin with the definitions of message and time complexities.

Definition 4.1: *Average time complexity* of algorithm `SIMPLE-WALK` (resp. `COALESCENT`) refers to $E[\tau_S]$ (resp. $E[\tau_C]$).

In adopting a measure of messaging complexity, let $\eta_S(t)$ and $\eta_C(t)$ be the total number of transmitted messages in the network by time t under algorithms `SIMPLE-WALK` and `COALESCENT` respectively:

Definition 4.2: *Average per-node message complexity* of algorithm `SIMPLE-WALK` (resp. `COALESCENT`) refers to $n^{-1}E[\eta_S(\tau_S)]$ (resp. $n^{-1}E[\eta_C(\tau_C)]$).

For each time t define $\xi(t) \triangleq (\xi_1(t), \xi_2(t), \cdots, \xi_n(t))$. Note that $(\xi(t) : t \geq 0)$ is a time-homogeneous Markov process under both algorithms. More precisely, $(\xi(t) : t \geq 0)$ is a random walk on G under `SIMPLE-WALK`, and a coalescing

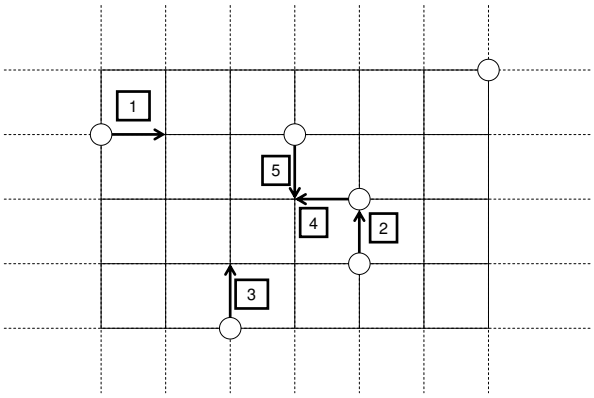


Fig. 2. An illustration of evolution of active nodes in algorithm COALESCENT. The circles in the figure represent nodes that are active at some arbitrary time t_o and arrows represent message transmissions after t_o, in the order indicated by their numbers. After message 1 the transmitting and receiving nodes exchange their status, and the value of the transmitting node passes onto the receiver. Message 2 results in one less active node in the network the transmitter becomes idle whereas the receiver maintain its active status. The number of active nodes do not change due to messages 3 or 4, but decrease again due to message 5.

random walk on G under COALESCENT. In particular the average time complexity of SIMPLE-WALK is the mean cover time of G. Average message complexities of the algorithms are characterized by the following lemma:

Lemma 4.1:

$$E[\eta_S(\tau_S)] = E\left[\int_0^{\tau_S} \sum_i \xi_i(t)dt\right] = E[\tau_S],$$
$$E[\eta_C(\tau_C)] = E\left[\int_0^{\tau_C} \sum_i \xi_i(t)dt\right].$$

Proof: Let $\mathcal{F}_t$ denote the sigma-field generated by $(|\xi(s)| : s \leq t)$ and let $(\phi(t) : t \geq 0)$ be a Poisson process with unit rate. Note that $\eta_S(t)$ has the same distribution as $\phi(\int_0^t |\xi(s)|ds)$ since each carrier node transmits messages at unit rate and idle nodes do not engage in message transmission, and thus $|\xi(t)|$ is the instantaneous rate of message generation in the network at time t. In particular the process $(\mu(t) : t \geq 0)$ with

$$\mu(t) = \eta(t) - \int_0^t |\xi(s)|ds$$

is a martingale adapted to $\{\mathcal{F}_t\}$. Note also that σ_1 is an $\{\mathcal{F}_t\}$ stopping time and it is almost surely finite and $\sup_{t\geq 0} |\xi(t)| \leq N^d$; hence it follows by the optional sampling theorem [14, Theorem 2.2.13] that $E[\mu(\sigma_1)] = 0$. ∎

Recall that average time complexity of SIMPLE-WALK is the mean cover time of G and the cover time results for explicit graphs already obtained in the explicit graphs we consider. See [11, Chapter 5] for the cover time results and by Lemma 4.1 it is clear how the message complexity is related to average time complexity for SIMPLE-WALK. Therefore we consider the average messages and time complexities for COALESCENT.

a) Completely connected graph: A completely connected graph is a graph where each vertex has an edge with every other vertex and the process *'The Coalescent'* in [9] is exactly COALESCENT on a completely connected graph. Therefore the time and message complexity results are obtained from [9].

b) Ring and d-dimensional torus: A d-dimensional torus is a graph where all the vertices have exactly $2d$ neighbors and it can be formed by joining the facing boundaries of a grid hence yielding a completely symmetric structure. A ring is simply a 1 dimensional torus. Consider $n = N^d$ for a d dimensional torus.

Let

$$s_N = \begin{cases} N^2 & \text{if } d = 1 \\ N^2 \log N & \text{if } d = 2 \\ N^d & \text{if } d \geq 3. \end{cases}$$

The following theorem gives the time complexity result for COALESCENT.

Theorem 4.1: [5, Theorem 6]

$$0 < \liminf_{N\to\infty} E[\tau_C]/s_N = \limsup_{N\to\infty} E[\tau_C]/s_N < \infty.$$

Consider a typical sample path of the process $(\sum_i \xi_i(t) : t \geq 1)$ in Figure 3, which is closely related to the number of messages transmitted. It can be noticed that initial phases the curve rapidly decreases however in the later phases the decay is slower. Therefore the process entails two different phases. In the initial phase there are plenty of nodes but due to high coalescences their number diminish very quickly. In this initial short phase there are high number of transmissions. In the later longer phase, however, fewer messages are transmitted. Therefore one needs to take into account the fact that these phases behave very differently in order to find a complexity result for the number of messages transmitted.

Let

$$m_N = \begin{cases} N^2 & \text{if } d = 1 \\ N^2(\log N)^2 & \text{if } d = 2 \\ N^d \log N & \text{if } d \geq 3. \end{cases}$$

The following theorem is an upper bound for the message complexity.

Theorem 4.2: [3, Theorem 7]

$$\limsup_{N\to\infty} E[\eta_C(\tau_C)]/m_N < \infty.$$

$E[\eta_C(\tau_C)] > E[\tau_C]$ due to $\sum_i \xi_i(t) \geq 1$ for all t therefore Theorem 4.1 is a lower bound on the growth rate of $E[\eta_C(\tau_C)]$. For a ring,i.e. 1-d torus, it is tight and it is off by at most a factor $\log N$ for $d \geq 2$. Numerical simulations suggest that $E[\eta_C(\tau_C)] = \omega(s_N)$ and $E[\eta_C(\tau_C)] = o(m_N)$.

V. Comparison with Gossip Algorithms

Recall Example 2.1. Let $k = 1$ so that $M^{k\times k} = \mathbb{R}$ and $x_i = 1$ uniformly for all $i = 1, 2, \cdots, n$. Therefore f simply computes averages. In this section we would like to compare the performance of COALESCENT and SIMPLE-WALK to that

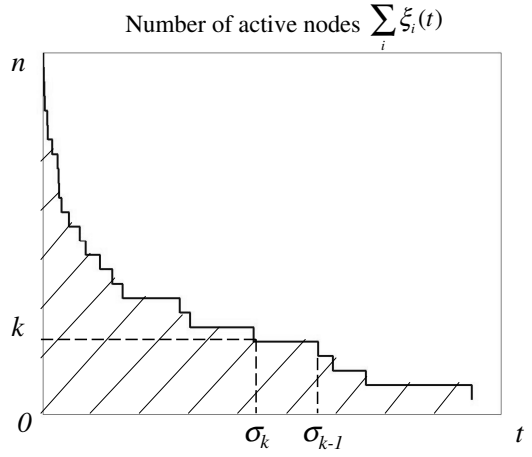


Fig. 3. A sample path of the number of active nodes under algorithm COALESCENT. σ_k denotes the first time that k active nodes remain in the network. This trajectory has qualitatively different statistics for small and large values of t/s_N. The mean of the shaded area is the mean aggregate number of transmitted messages in the network.

of so-called gossip algorithms, and obtain message and time complexity results under the same settings in explicit graphs.

Gossip algorithms refer to distributed randomized algorithms that are based on pairwise relaxations between randomly chosen node pairs. In the present context a pairwise relaxation refers to averaging of two values available at distinct nodes. We next give a full description of this algorithm as studied in [4]. In what follows a stochastic matrix $P = [P_{ij}]_{n\times n}$ is called *admissible* for G if $P_{ij} = 0$ unless nodes i and j are neighbors in G. The algorithm is parameterized by such P:

Algorithm GOSSIP-AVE(P): Each node i maintains a real valued variable with initial value $z_i(0) = x_i$. At the tick of a local Poisson clock, say at time t_o, node i chooses a neighbor j with respect to the distribution $(P_{ij} : j = 1, 2, \cdots, n)$ and both nodes update their internal variables as $z_i(t_o) = z_j(t_o) = (z_i(t_o^-) + z_j(t_o^-))/2$.

Let $\bar{x}$ denote the average of $x_1, x_2, \cdots, x_n$, let $z(t)$ denote the vector $(z_1(t), z_2(t), \cdots, z_n(t))$ of node values at time t, and $\mathbf{1}$ denote the vector of all 1s. Define τ_k as the kth time instant such that some local clock ticks and thereby triggers messaging in the network. For $\varepsilon > 0$ let the deterministic quantity $K(\varepsilon, P)$ be defined by

$$K(\varepsilon, P) = \sup_{z(0)} \inf \left\{ k \ : \ Pr\left(\frac{\|z(\tau_k) - \bar{x}\mathbf{1}\|_2}{\|z(0)\|_2} \geqslant \varepsilon \right) \leqslant \varepsilon \right\}.$$

In [4] $K(\varepsilon, P)$ is considered as a termination time for Algorithm GOSSIP-AVE(P) and minimization of $K(\varepsilon, P)$ is sought by proper choice of P. Here we adopt the same interpretation for comparison purposes. It should perhaps be noted here that this is a fairly weak stopping criterion as $\|z(\tau_{K(\varepsilon,P)}) - \bar{x}\mathbf{1})\|_\infty/|\bar{x}|$ may be much larger than ε.

Since a gossip algorithm is in essence a matrix multiplication algorithm the convergence rate of $z(t)$ to the all average vector must be some explicit function of the second greatest eigenvalue of P. It is indeed true as the next proposition states.

	COALESCENT	SIMPLE-WALK	GOSSIP
Com. connected	$\Theta(\log n)$	$\Theta(\log n)$	$\Theta(1)$
Ring	$\Theta(n)$	$\Theta(n)$	$\Theta(n^2)$
Torus ($d = 2$)	$O((\log n)^2)$	$\Theta((\log n)^2)$	$\Theta(n)$
Torus ($d \geq 3$)	$O(\log n)$	$\Theta(\log n)$	$\Theta(n^{2/d})$

(a)

	COALESCENT	SIMPLE-WALK	GOSSIP
Com. connected	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(1)$
Ring	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$
Torus ($d = 2$)	$\Theta(n \log n)$	$\Theta(n(\log n)^2)$	$\Theta(n)$
Torus ($d \geq 3$)	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^{2/d})$

(b)

TABLE I

(A) MESSAGE AND (B) TIME COMPLEXITIES FOR THREE TOPOLOGIES WITH n NODES.

Proposition 5.1: [4, Theorem 3] Let P be a stochastic matrix that completely characterizes a gossip algorithm, n number of nodes in the network and $W = I - D/2n + (P + P^T)/2n$ where D is a diagonal matrix with entries $D_i = \sum_{j=1}^n (P_{ij} + P_{ji})$. Let $\lambda_2(W)$ denote the second greatest eigenvalue of W. Then for any fixed $\epsilon > 0$

$$\frac{0.5 \log \epsilon^{-1}}{\log \lambda_2(W)^{-1}} \leq K(\epsilon, P) \leq \frac{3 \log \epsilon^{-1}}{\log \lambda_2(W)^{-1}}.$$

Hence once $\lambda_2(W)$ is computed as a function of n one can obtain the time complexity results for gossip algorithms. Per node message complexity results are in the same order as the time complexity results.

c) Completely connected graph: Let P_C denote the stochastic matrix that characterizes the gossip algorithm on a completely connected graph. If P_C is chosen such that each node chooses its neighbor with equal probability then $P_C(i, j) = 1/(n-1)$ if $i \neq j$ and 0 otherwise. Therefore

$$W = (1 - \frac{1}{n} - \frac{1}{n(n-1)})\mathbf{I} - \frac{1}{n(n-1)}\mathbf{1}$$

yielding $\lambda_2(W) = (1 - \frac{1}{n} - \frac{1}{n(n-1)})$ hence $K(\epsilon, P) = \Theta(n)$.

d) Ring and d-dimensional torus: The following theorem provides a lower bound for $K(\varepsilon, P)$ that applies uniformly for all P on the d-dimensional lattice torus:

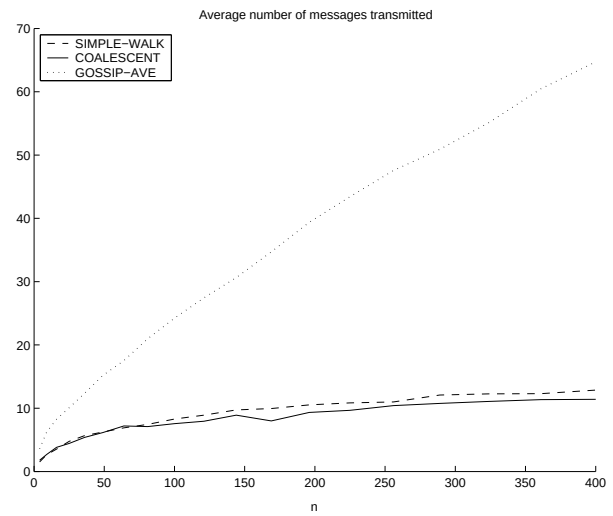
Theorem 5.1: [3, Theorem 8] Let G be d-dimensional lattice torus with N^d nodes. Given $\varepsilon > 0$, $K(\varepsilon, P) = \Omega(N^{d+2})$ for all admissible P.

The next theorem summarizes the time and message complexity results obtained for all algorithms.

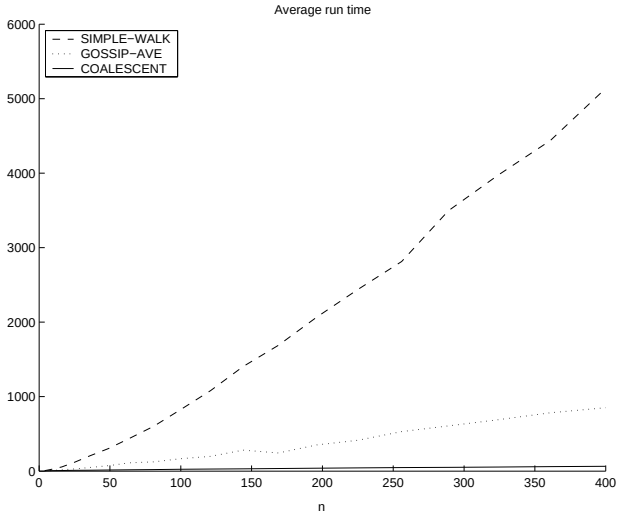
Theorem 5.2: Time and message complexity results of algorithms SIMPLE-WALK, COALESCENT and GOSSIP-AVE are summarized in Table I.

VI. NUMERICAL RESULTS

In this section we numerically verify the analytical results of Table I on a $2-d$ torus as an example and obtain numerical

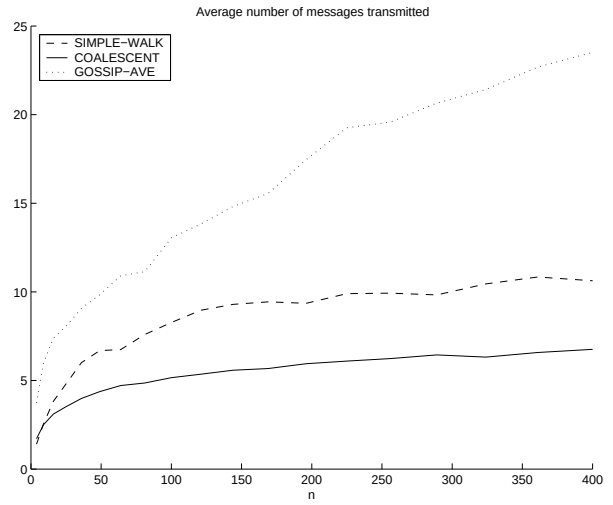


(a) Per-node message complexities

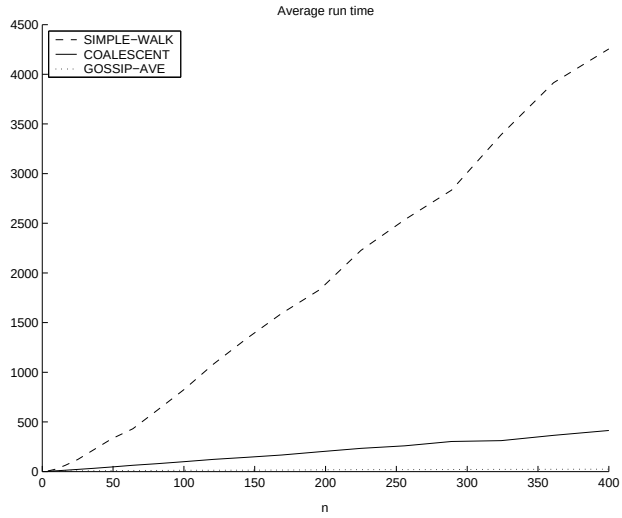


(b) Time complexities

Fig. 4. Average run times and number of messages per node on the 2-dimensional torus with n nodes. The solid curve represents simulation results for `COALESCENT`, the dashed curve for `SIMPLE-WALK` whereas the dotted curve represent lower bounds for `GOSSIP-AVE`(P) based on a lower bound for $K(\varepsilon, P)$. Note also that exact value of $F_n(\lambda_1, \lambda_2, \cdots, \lambda_n)$ is obtained at the termination of `COALESCENT` or `SIMPLE-WALK` whereas no such claim can be made for `GOSSIP-AVE`(P).



(a) Per-node message complexities



(b) Time complexities

Fig. 5. Average run times and number of messages per node on geometric random graphs.

results for average run times and average number of messages transmitted on geometric random graphs.

Geometric random graphs have received much attention recently in wireless network community due to their success to model wireless networks. See for example [15] and [4]. Therefore we find the numerical analysis of the time and message complexities of the algorithms on geometric random graphs useful.

Definition 6.1: A geometric random graph in $d \geq 1$ dimensions with n nodes and radius $r(n)$, denote by $\Gamma = (V_\Gamma, E_\Gamma)$, is a graph where nodes are distributed in a unit hypercube and $(i, j) \in E_\Gamma$ if and only if the Euclidian distance between i and j is smaller than $r(n)$.

Specifically we have chosen a 2 dimensional geometric random graph of n nodes distributed in a unit square, with radius $\sqrt{2 \log n / n}$. Hence the graph has good connectivity and interference is minimized ([15], [16]). However in order to avoid trivialities, we considered only connected geometric random graphs and ignored the disconnected ones in our simulations.

In order to make a fair comparison of `GOSSIP-AVE` with

COALESCENT and SIMPLE-WALK we use

$$K_2(\epsilon, z(0)) = \inf\left\{k \ : \ \frac{\|z(\tau_k) - \bar{z}\mathbf{1}\|_2}{\|z(0)\|_2} \leqslant \varepsilon\right\}$$

as a stopping criterion for GOSSIP-AVE.

The numerical results of average number of messages and run times on 2-D torus and geometric random graphs are in Figures 4 and 5, respecctively.

VII. Conclusion

We have studied the time and message of complexities of two algorithms, namely SIMPLE-WALK and COALESCENT, and have compared our results to gossip algorithms. The results suggest a trade-off between run times and transmitted messages. For example for a 2 dimensional torus running COALESCENT over gossip algorithms provides $O(n(\log n)^{-2})$ gain in terms of per-node message transmissions for a loss of $\Theta(\log n)$ in run time. Therefore the sensor network designer may choose to employ COALESCENT in order to save energy.

Acknowledgment

This research was supported by Presidential Early Career Award N00014-02-100362, NSF CAREER Programs ANI-0238397, ECS-0449194 and NSF programs CCF-0430983, and CNS-0435353.

References

[1] M. Alanyali, S. Venkatesh, O. Savas, and S. Aeron. Distributed bayesian hypothesis testing in sensor networks. *American Control Conference*, Boston, MA, July 2004.

[2] V. Saligrama, M. Alanyali and O. Savas. Distributed detection in sensor networks with packet losses and finite capacity links. *IEEE Transactions on Signal Processing*, to appear, 2005.

[3] O. Savas, M.Alanyali, and V. Saligrama, Efficient In-Network Processing through Local Ad-hoc Information Coalescence. Technical Report. Boston University, Dept. of Electrical and Computer Engineering. 2006.

[4] S. Boyd, A. Ghosh, B. Prabhakar and D. Shah. Gossip algorithms: Design, analysis and applications. *IEEE INFOCOM 2005*, 2005.

[5] J.T. Cox. Coalescing random walks and voter model consensus times on the torus. *The Annals of Probability*, 17(4):1333-1366, 1989.

[6] A. Giridhar and P.R. Kumar, Computing and communicating functions over sensor networks. *IEEE Journal on Selected Areas in Communications.* pp. 755–764, vol. 23, no. 4, April 2005.

[7] N. Khude, A. Kumar and A. Karnik, Time and Energy Complexity of Distributed Computation in Wireless Sensor Networks, *IEEE INFOCOM 2005*, 2005.

[8] D.S. Scherber and H.C. Papadopoulos. Distributed computation of averages over ad hoc networks. *IEEE Journal on Selected Areas in Communications*, vol. 23(4), 2005.

[9] S. Tavaré. Line-of-descent and genealogical processes, and their applications in population genetics models. *Theoret. Population Biol.*, vol. 26, pp. 119-164, 1984.

[10] D. Zuckerman, A technique for lower bounding the cover time, *SIAM Journal on Discrete Mathematics*, no. 5, pp. 81-87, 1992.

[11] D. Aldous and J. Fill, Reversible Markov Chains and Random Walks on Graphs, monograph available at http://www.stat.berkeley.edu/ aldous/RWG/book.html.

[12] D. Kempe, A. Dobra and J. Gherke, Gossip-based computation of aggregate information. In Proc. IEEE Conference of Foundations of Computer Science (FOCS), 2003.

[13] A. G. Dimakis, A. D. Sarwate and M. J. Wainwright, Geographic gossip: Efficient aggregation for sensor networks, IPSN 2006, 2006.

[14] S. N. Ethier and T. G. Kurtz, Markov Processes, John Wiley and Sons Inc., New York, 1986.

[15] P. Gupta and P. Kumar, The capacity of wireless networks, IEEE Trans. on Information Theory, 46(2):388-404, March 2000.

[16] A. E. Gamal, J. Mamman, B. Prabhakar and D. Shah, Throughput-delay trade-off in wireless networks, INFOCOM 2004. 2004.