Distributed Optimal Self-Organization in Ad Hoc Wireless Sensor Networks

Aditya Karnik and Anurag Kumar, *Fellow, IEEE*

***Abstract*—This work is motivated by the idea of using randomly deployed wireless networks of miniature smart sensors to serve as distributed instrumentation. In such applications, often the objective of the sensor network is to repeatedly compute and, if required, deliver to an observer some result based on the values measured at the sensors. We argue that in such applications it is important for the sensors to self-organize in a way that optimizes network throughput. We identify and discuss two main problems of optimal self-organization: 1) building an optimal topology, and 2) tuning network access parameters, such as the transmission attempt rate. We consider a simple random access model for sensor networks and formulate these problems as optimization problems. We then present centralized as well as distributed algorithms for solving them. Results show that the performance improvement is substantial and implementation of such optimal self-organization techniques may be worth the additional complexity.**

***Index Terms*—Distributed stochastic algorithms, self-organization, wireless sensor networks.**

I. Introduction

EQUIPPED with a microprocessor, memory, radio and a battery, miniature sensing devices now combine the functions of sensing, computing, and wireless communication into *smart sensors*. A smart sensor may have only modest computing power but the ability to communicate will allow a group of sensors to organize themselves into a network and collaborate to execute tasks more complex than just sensing and forwarding the information, for example, on-line distributed processing of the sensed data. By processing information collaboratively within the network, smart sensors will not only be able to monitor the physical environment but manipulate it as well [1], [2]. Thus the smart sensing technology portends the new dimension of *embedded computing and control* into distributed instrumentation [3]. The decisive factor, however, is the rate at which sensors can collaboratively process data, since the physical process being monitored will have certain processing requirements, and unlike conventional distributed processing machines, which have high speed communication buses between a pair of processors, a sensor network has low speed wireless links, lacks centralized control, is ad hoc in nature and may be energy constrained. It is, therefore, imperative that *sensors not only self-organize but do so optimally, in the sense of maximizing computing rate, for given task objectives and constraints*.

The computing rate of a given task, however, cannot be directly optimized except in simple cases. First, because finding a functional form for it is a formidable task since it depends on, among other things, the way the computation is "arranged" using a distributed algorithm. Second, even if such a form is known optimizing it would be a centralized problem since individual sensors will hardly have any notion of the global computing rate. Therefore, the optimization problem must be based only on objectives local to the sensors. Note that, a global computation proceeds in steps comprising of certain local computations at each sensor. Thus, the faster the sensors complete their share of local computations the higher will the global computation rate be. A simple class of distributed computing algorithms will require each sensor to periodically exchange the measurements and/or partial results of local computation with the other sensors. The more frequently exchanges of results among neighboring sensors can occur, the more rapidly will the overall computation converge. The more rapid the progress of the computation the faster the variations of a spatio-temporal process that can be tracked. Thus, the sensors must organize themselves in such a way as to optimize their communication throughput. They can do so by forming an optimal network topology and tuning to optimal transmission attempt rates since these are the crucial factors that determine the throughput. It is in these two aspects that we investigate optimal self-organization of sensor networks.

Our objective in this paper is to analytically formulate the notion of optimal network organization, investigate distributed algorithms leading to it and study the achievable performance gains. To this end, we propose an analytical model for sensor networks involved in processing data continuously or over long, possibly critical, active periods. We formulate the optimization problems of self-organization in our model and present distributed algorithms for solving them. Our results show that the performance improvements are substantial, therefore, implementation of such self-organization techniques is worth the additional complexity. In this respect, our work should be seen as a step towards eventually understanding algorithms for self-optimizing sensor networks.

This paper is organized as follows. In Section II we review the previous work in the area. Section III discusses our model of sensor networks. In Section IV we formulate the problem of optimal self-organization. The problem of optimal network topology is discussed in Section V and tuning to an optimal

Manuscript received December 1, 2004; revised May 1, 2005, and May 30, 2005; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Das. This work was supported in part by a grant from the Indo-French Centre for the Promotion of Advanced Research (IFCPAR) (Project No. 2900-IT), and in part by a fellowship from the IBM India Research Lab. This work was done while A. Karnikwas in the Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore. Part of this paper appeared in IEEE Infocom 2004.

A. Karnik is with the General Motors India Science Lab, Bangalore, India 560 066 (e-mail: aditya.karnik@gm.com).

A. Kumar is with the Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore 560 012, India (e-mail: anurag@ece.iisc.ernet.in).

Digital Object Identifier 10.1109/TNET.2007.896227

channel access rate in Section VI. Section VII follows with discussion and we conclude in Section VIII. Proofs are sketched in Section IX.

II. Related Work

In recent years, literature on sensor networks has grown tremendously. Self-organization of sensor networks, in particular, has received wide attention. Its various aspects, namely topology discovery and control, scheduling, localization and energy efficiency have been addressed in the previous papers, their main focus being the protocol design. For example, topology formation and scheduling for ad hoc networks have been discussed in [4] and [5]; note that the generation of schedules is an impractical task for large random networks. A survey on topology control in sensor network is presented in [6]. Ref. [7] presents a message efficient clustering algorithm for sensor networks. Various self-organizing procedures are described in [8], whereas specific protocols are discussed in [9]. Experimental performance studies of a protocol for formation of connected topology of sensor networks are reported in [10]. Localization aspects have been addressed in [11] and [12]. Ref. [13] discusses a protocol which allows nodes to build transmission schedules to conserve energy. Impact of energy conservation techniques (sleep/active schedules) on the performance of sensors (in terms of delivery delay) has been studied in [14].

Our work differs from the previous work in the following aspects. In contrast to typical data aggregation applications, we consider applications where sensors are engaged in "in-network" computing. Such applications would typically demand certain performance (computing rate or delay) from the network; hence we view *performance optimization as the objective for self-organization* and our algorithms are motivated by this goal. The current literature, on the other hand, has largely overlooked this issue of optimality of the performance of the resulting network organization. We believe that our analytical approach and formulations are the first of their kind in this area. We also substantiate our results by simulations.

III. A Model for Sensor Networks

In many applications the sensors will monitor (and possibly manipulate) the environment *autonomously by in-network computing*. For ease of implementation and robustness the computational algorithms will be based on strictly local interactions among sensors (e.g., [12], [15]). Our work focuses on this paradigm. We, therefore, model only the local computing and communication since a sink and, therefore, directed traffic flows are not prerequisites in this set-up. This, however, does not exclude all together the applications in which computational results need to be delivered to a sink. For example, delivering the maximum of sensed values to the sink can be seen as a computation in itself and is, thus, addressed by our model [21]. In other scenarios the computational results may percolate to the sink rather than through explicit flows; for example, in gossip based computation the sink can actually be part of the network participating in the local exchanges.

We consider a random access communication model. We believe that an elaborate MAC may not be suitable due to control data overheads and moreover, it is not clear how much improvement a scheme like RTS-CTS will lead to in dense networks. In addition distributed TDMA scheduling has certain limitations, such as scalability, fault tolerance, inflexibility to traffic conditions, dependency on simplistic interference models, etc. [16]. Therefore, random access is an attractive MAC for large ad hoc sensor networks. We assume slot synchronization among sensors. Time synchronization is vital for some sensing tasks [17]; hence our slotted time assumption may not be very restrictive. Moreover, even in the absence of time synchronization, slot synchronization can be achieved by distributed algorithms [18].

The following are the elements of our model.

Deployment: We assume that a large number (denoted by N) of static sensor nodes are placed (possibly randomly) in a region. The sensors are engaged in a generic sensing-and-computing task such as monitoring the level of some chemical contaminant.

Communication: All sensors transmit on a common carrier frequency using omni-directional antennas and a fixed (common) transmit power. A sensor cannot transmit and receive simultaneously. We consider only the path loss with exponent η. Letting d_0 denote the near field crossover distance, the power received at a distance r from a transmitter, $P_s(r) = (r/d_0)^{-\eta}$ if $r > d_0$ and $P_s(r) = 1$ if $r \leq d_0$. We say that a transmission can be "decoded" when its signal to interference ratio (SIR) exceeds a given threshold β (based on the BER requirement). Transmission range (denoted by R_0) is defined as the maximum distance at which a receiver can decode a transmitter in the absence of any co-channel interference. Thus for a transmission to be decoded a receiver not only needs to be within R_0 from a transmitter but the SIR needs to be above β as well. Time is slotted and channel access is random, i.e., in each slot, sensor i decides to transmit with probability α_i and decides to receive with probability $(1 - \alpha_i)$ independent of anything else; α_i is called the *attempt probability* of sensor i.

Computation: We assume that processing in the network is either continuous or over long activity periods after durations of dormancy [20]; "continuous" does not mean that all the sensors are always ON (see Section VII for further discussion). Fig. 1 shows a traffic model for a fragment of a sensor network engaged in distributed computation. A local algorithm running on each sensor uses the local measurements and updates from the other sensors to perform certain computations. The nodes within the transmission range of a sensor with whom it communicates the local data (raw measurements, computational results) are designated as its *neighbors*. The local data to be sent to the neighbors are queued up in a packet queue. The algorithm is such that the sensors communicate randomly with each other. In particular we assume that if a sensor decides to transmit, the destination of the head-of-the-line packet is equally likely to be any of the neighbors. A transmission is successful when the sensor to which it is addressed is in the receive mode, and is able to decode the transmission. If a transmission is successful, the corresponding packet is removed from the queue, i.e., instantaneous acknowledgements are assumed. A successfully received packet at a sensor invokes a new computation that *may* result in an update being sent to a neighbor. We model this probabilistically, i.e., the successful reception of a packet generates another packet to be sent to a neighbor with probability ν.

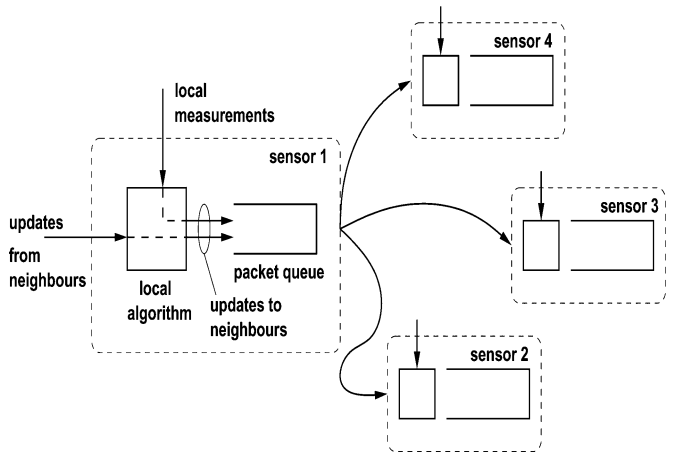

Fig. 1. A traffic model for sensors carrying out measurements and distributed computation.

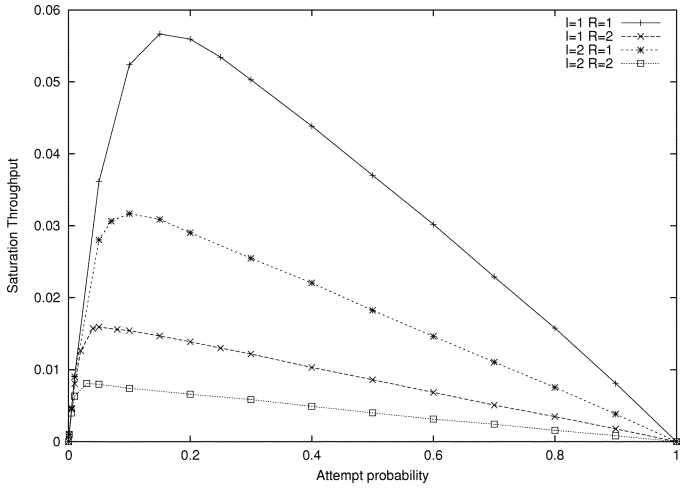

Fig. 2. Saturation throughput ($p_t^{(s)}$) variation with attempt probability (α) for $\lambda = 1$ and 2 per m^2. R equals 1 or 2 m.

Some important remarks are in order. Transmission to each neighbor with equal probability in our computation model is motivated by easy-to-implement asynchronous algorithms based on random (and strictly local) communication among sensors (gossip algorithms). The assumptions of omni-directional antenna and a fixed power level are made to impose minimal hardware requirements. With simple hardware, sensors may have few power levels to choose from, if not, a sophisticated power control. Though we do not consider multiple power levels, we believe that our techniques can be extended in that direction. The motivation for considering only the path loss on wireless links is that in short-path flat-terrain fixed wireless channel, temporal variations are very slow and there is a strong LOS path [19]. Nevertheless, the algorithms we develop for self-organization are *measurement-based and not model-based*, i.e., they are based on the estimates of the quantities of interest (e.g., sensor throughputs or transmission success probability on a link) obtained through on-line measurements, rather than on any particular analytical form for them. It will be clear from Section V and Section VI that many of the stated assumptions, in particular SIR-based decoding, communication without multipath effects, equal transmission ranges, transmission to each neighbor with equal probability,[1] instantaneous acknowledgements, probabilistic updates upon reception, are not essential for the algorithms *per se*. The main utility of these suppositions is to make the analysis tractable.

IV. Optimal Self-Organization: Formulation

In order to capture the temporal properties of a spatial process being monitored, the sensors need to sample it at a rate akin to a Nyquist rate.[2] These samples along with the packets triggered by updates from the neighbors form an arrival process into the queue (see Fig. 1). Therefore, this sampling rate cannot exceed the rate at which packets are drained from the queue. More precisely, assume that $\alpha_i = \alpha$ for each i and that each sensor has all the nodes within a fixed distance, say $R \leq R_0$ as its neighbors. In [21], we show that if γ denotes the arrival rate of measurements at each sensor, then for a given sample path of the Poisson distributed sensor field of intensity λ (denoted by $\mathcal{N}$) the packet queue at sensor i is stable if $\gamma < p_{i,t}^{(s)}(\lambda, \alpha; \mathcal{N}) - (1-\alpha)\nu$ where $p_{i,t}^{(s)}(\lambda, \alpha; \mathcal{N})$ denotes the probability of successful transmission (subscript t) by sensor i in a slot in saturation (superscript (s)), i.e., when the head-of-the-line packet, after its successful transmission, is immediately replaced by another packet at each queue. $p_{i,t}^{(s)}(\lambda, \alpha; \mathcal{N})$ is also called the *saturation throughput*. Now in applications in which the sensors are used in a control loop or even to detect critical phenomena they will need to sample and process data at faster time-scales. The "traffic" carried by their network in such situations may not be "light". It will also be governed by the computational accuracy requirements. Moreover, a wireless network is a queueing system with "load dependent service rate". The "service rate" is also affected inversely by the sensor density; a high density would be of interest for capturing the spatial properties of a spatial process. Hence, even if the sensors are sampling at a slower time-scale, the overall traffic which includes the "computational" traffic as well may not be "light" and may have the network near saturation. We, therefore, argue that in our model, the saturation throughput can be a good measure of performance.

To capture the random dispersion of sensors, we average $p_{i,t}^{(s)}(\lambda, \alpha; \mathcal{N})$ over all sensors and sample paths, and denote it by $p_t^{(s)}(\lambda, \alpha)$. This work is particularly motivated by Fig. 2 which shows the variation of $p_t^{(s)}(\lambda, \alpha)$ with α; we have assumed that $\alpha_i = \alpha$ for each i. We use 1000 Poisson distributed points as sensors on the plane for $\lambda = 1$ and 2 per m^2. We take $\eta = 4$, $\beta = 10$ dB, $R_0 = 6$ m and R equals 1 or 2 m. Throughputs are averaged over 1000 random point placements. Observe that, for a fixed value of λ, $p_t^{(s)}(\lambda, \alpha)$ decreases as R increases and for a fixed value of R, $p_t^{(s)}(\lambda, \alpha)$ decreases as λ increases. Thus, high values of $p_t^{(s)}(\lambda, \alpha)$ decree small values of R, however, arbitrarily small values of R result into a disconnected network. Note also from Fig. 2 that, for a fixed λ and R, there is a value of α which maximizes $p_t^{(s)}(\lambda, \alpha)$.

[1]This, we think, is a reasonable assumption to construct a topology because even before discovering its neighbors a sensor cannot determine its communication pattern. It may, however, be unnecessary once the topology is determined.

[2]We draw an analogy between the Nyquist rate and the local measurement rate of a sensor, however, we do not assume that the traffic generated by sensors is necessarily regular or periodic.

Thus, sensors need to form a network that is "optimally connected", and operated at an "optimal attempt rate". Since the transmission powers of sensors are assumed fixed, our notion of optimal connectivity is that of "interconnecting sensors" so as to build an efficient distributed computing machine and not of forming a topology by controlling the transmission powers. The transmission attempt rate, unlike power, can be easily modified.

Instead of addressing connectedness in terms of "connectivity range" we migrate to a more general concept of topology. Some notation is in order. Let G denote a connected weighted graph with vertex set V ($|V| = N$), edge set E and weight function $W : E \to R_+$. The weight of an edge $(i,j) \in E$ is denoted by $w(i,j)$. G can be a directed or an undirected graph. If G is directed, connectivity refers to strong connectivity. V_s denotes the set of sensors; each element in V_s is a triplet of the form (i, x_i, y_i) where $i \in \{1,2,\ldots,N\}$ is the sensor index, and x_i and y_i are the x-coordinate and y-coordinate of i respectively.

Definition 4.1: The *transmission graph*, G_{R_0}, is the symmetric directed graph with V_s as its vertex set, and $(i,j) \in E_{R_0}$ if sensor j is within a distance R_0 of sensor i. □

For randomly located N sensors the transmission graph G_{R_0} is a geometric random graph. Henceforth we will be concerned with the actual realization of G_{R_0}, and for brevity we refer to it by G_{R_0}. We will assume G_{R_0} to be a connected graph.[3] Thus, G_{R_0} lays out the possible "interconnections" of sensors and each subgraph of G_{R_0} specifies one such interconnection or in other words a computing topology, i.e., a set of *neighbors* for each sensor. In Section IV, we considered only special subgraphs of G_{R_0} obtained by connecting sensors within $R \le R_0$ of each other as the network topology. Let $M(G', \underline{\alpha})$ denote the network throughput, i.e., the sum of individual sensor throughputs with topology specified by G'. Now if all the sensors always have packets to send then $\frac{M(G',\underline{\alpha})}{N}$ is the average saturation throughput of the network. Then the discussion so far motivates the following problem:

$$\max_{\{G \in G_{\rm cs}, 0 \le \alpha \le 1\}} M(G,\alpha) \tag{1}$$

where $G_{\rm cs}$ is the set of all connected spanning subgraphs of G_{R_0}. Note that $G_{R_0} \in G_{\rm cs}$.

Proposition 4.1: For every $G \subset G_{\rm cs}$, $M(G,\alpha)$ is a strictly quasiconcave function of α.

Proof: See Section IX. □

$G_{\rm cs}$ is a finite set. Therefore, for a fixed α there is a G^* which maximizes M. On the other hand, for a fixed topology, G, a unique maximizer $\alpha^*(G)$ of $M(G,\alpha)$ exists by Proposition 4.1.[4] However, neither G^* nor $\alpha^*(G)$ can be computed *a priori* and used in the field owing to the random deployment of sensors, and hence *a priori* unknown G_{R_0}. Therefore, the sensors must learn an optimal topology and optimal α on their own. The previous discussion suggests an iterative approach to find G^* and $\alpha^*(G^*)$. However, modifying topology in search of an optimal one is practically infeasible. Moreover, tuning to a common value of α^* in a distributed way is difficult; more importantly, different sensors may need different values of α to counter the local inhomogeneities in the node placement. *Hence, our approach will be to formulate the problems (hence the notions of optimality) and the algorithms for topology and attempt rate separately*. Due to this "decoupling" the algorithms for topology can be used for any α_i's and the algorithms for attempt probabilities can be used on any topology. This way sensors can form a topology motivated by the computational requirements and then tune their attempt probabilities. In Section V, we undertake the problem of defining and building an optimal topology and in Section VI, the problem of defining and tuning to optimal attempt probabilities.

[3] G_{R_0} can also be taken as the giant component in the sensor field.

[4] Even for a known placement of sensors, computation of these values is difficult if not impossible.

V. Optimal Network Topology

Let $G' := (V', E')$ denote a subgraph of a given connected graph G. For $i \in V'$, let $d_i(G')$ denote the out-degree of node i in G'. For all $i \in V'$, let

$$\psi_i(G') = \frac{1}{d_i(G')} \sum_{(i,j) \in E'} w(i,j) \tag{2}$$

if $d_i(G') > 0$ and $\psi_i(G') = 0$ otherwise. Define a function ψ on G' as $\psi(G') := \sum_{i \in V'} \psi_i(G')$ and let $\tilde{G} = \arg\max_{G' \in G_{\rm cs}} \psi(G')$ where $G_{\rm cs}$ is the set of all connected spanning subgraphs of G. $G_{\rm cs}$ is nonempty since $G \in G_{\rm cs}$. $\tilde{G}$ maximizes the measure ψ over all connected spanning subgraphs of G. We call $\tilde{G}$ the *maximum average-weighted spanning subgraph* (MAWSS) of G. We will use the term MAWSS to also denote an algorithm for determining an MAWSS.

In (1) all the α's are the same. Here we will allow different α's for different sensors as well. Denote by $\underline{\alpha}$ the vector of α_i's, $\underline{\alpha} := (\alpha_1, \alpha_2, \ldots, \alpha_N)$. Let $\underline{\alpha}$ be fixed and let for $(i,j) \in E_{R_0}$, $p_{ij}(\underline{\alpha})$ denote the probability of successful transmission from sensor i to j under $\underline{\alpha}$. Recall that, we are assuming that all sensors have packets to send. Therefore, according to our model

$$p_{ij}(\underline{\alpha}) = \alpha_i(1-\alpha_j) P\left(\frac{\left(\frac{d_{ij}}{d_0}\right)^{-\eta}}{\sum_{k \ne i,j} \left(\frac{d_{kj}}{d_0}\right)^{-\eta} Y_k} \ge \beta \right) \tag{3}$$

The last term in (3) is $P(\Gamma_{ij} \ge \beta)$ where Γ_{ij} denotes the SIR of a transmission from i to j.[5] d_{kj} is the distance between k and j, and d_0 is the near-field crossover distance. $\{Y_k\}$ are independent random variables such that Y_k is 1 if k transmits and 0 otherwise. Since $p_{ij}(\underline{\alpha})$ depends on the geometry of interferers of sensor j, $p_{ji}(\underline{\alpha})$ need not equal $p_{ij}(\underline{\alpha})$ in general.

Let $N_i(G')$ (respectively $n_i(G')$) denote the set of neighbors (respectively the number of neighbors) of i in topology G'. Now for each i define

$$M_i(G', \underline{\alpha}) = \frac{1}{n_i(G')} \sum_{j \in N_i(G')} p_{ij}(\underline{\alpha}) \tag{4}$$

Thus, $M_i(G', \underline{\alpha})$ equals the time average throughput of sensor i and $M(G', \underline{\alpha}) = \sum_{i=1}^{N} M_i(G', \underline{\alpha})$ We have used our assumption that in transmit mode a sensor transmits a packet to one

[5] A dense sensor network is an interference constrained system so that thermal noise can be ignored from SIR. Thus, SIR does not depend on actual transmit powers.

of its neighbors with probability $\frac{1}{n_i(G')}$. Note that, the "out-degree" of a sensor in G' is simply the number of its neighbors, $n_i(G')$. It, thus, follows by comparing (2) and (4) that for a fixed $\underline{\alpha}$ if G' is a subgraph of G_{R_0}, and if for all $(i,j) \in E_{R_0}$, $w(i,j)$ equals $p_{ij}(\underline{\alpha})$, then $\psi(G')$ is $M(G', \underline{\alpha})$. Since a sensor network needs to be connected, it follows from our formulation in Section IV that, *the optimal topology of a sensor network is the MAWSS of its* G_{R_0}.

Proposition 5.1: MAWSS for directed and undirected graphs is NP-complete.

Proof: See Section IX. □

In the following, we discuss directed graphs in particular, and propose a heuristic algorithm for obtaining an approximation to the MAWSS. For generic computing tasks, a topology with symmetric connectivity would be preferred (if i has a link to j, j has to have the reverse link). Note that, symmetric topology problem is also NP-complete since MAWSS for undirected graphs is a special case of it. We call the optimal symmetric MAWSS topology the SYMMAWSS. For the lack for space we omit the discussion of SYMMAWSS; for the details see [21].

A. Centralized MAWSS Algorithm

Some notation is in order. For node i, $e_i(k)$ denotes the kth heaviest outgoing edge and $w_i(k)$ denotes its weight. Ties are resolved arbitrarily. $E_1(G) := \{e_i(1) | i \in G\}$, is the set of maximum weight outgoing edges of all the nodes in G. The basic idea is the following. It is clear that the MAWSS contains $E_1(G)$. Hence if $(V, E_1(G))$ is strongly connected, we are done. If not, we convert the "maximum average weight" problem to the "minimum sum of weights" problem by a suitable transformation of $w(i,j)$ to $\bar{w}(i,j)$. We consider the transformation $\bar{w}(i,j) = w_i(1) - w(i,j)$ and denote this weight function by $\bar{W}$. We, then, construct minimum weight out-branching (directed tree or arborescence) using $\bar{w}(i,j)$ rooted at each i. Recall that, any out-branching rooted at a given node contains one and only one edge incoming to every other node. The minimum weight branchings pick out edges with small $\bar{w}(i,j)$ which are the edges with large $w(i,j)$. The resulting graph is taken as an approximation to the MAWSS. An optimal algorithm for constructing optimal branchings is presented in [22]. The advantage of underlying trees is that they are efficient for information transport and in a variety of scenarios where the computing and the communication tasks can be merged, for example, calculation of the maximum of the sensed values [23].

Proposition 5.2: The output $\hat{G}$ of Algorithm 1 is a strongly connected spanning subgraph of G.

Proof: Note that Algorithm 1 constructs a route from every node to every other node. □

Algorithm 1 Algorithm for Finding an Approximation $\hat{G}$ to the MAWSS of a Directed Graph G

1: **if** $(V, E_1(G))$ is strongly connected **then**
2: $\quad \hat{G} = (V, E_1(G))$
3: **else**
4: $\quad$ For all $(i,j) \in E, \bar{w}(i,j) := w_i(1) - w(i,j)$ and set $\bar{G} = (V, E, \bar{W})$
5: $\quad$ For all $i \in V$, find $G^i_{\text{out}} = (V, E^i_{\text{out}})$, the minimum weight out-branching of $\bar{G}$ rooted at i
6: $\quad \hat{G} = (V, \cup_{i \in V} E^i_{\text{out}})$

B. A Distributed MAWSS Algorithm

At the time of deployment, neither G_{R_0} nor $p_{ij}(\underline{\alpha})$ is known to sensors. Over time, sensors "discover" each other by advertising their ids which can be simply their indices. Let $\underline{\alpha}$ and the locations of the sensors be fixed. At time 0, the sensors start broadcasting their ids. Let $G_n = (V_n, E_n)$ denote the subgraph of G_{R_0} discovered until time n, i.e., $V_n = V_s$ and $(i,j) \in E_n$ if there exists a time slot $m \leq n$ in which sensor j successfully received a transmission from i for the first time. $G_0 = (V_s, \phi)$. Note that G_n is a random graph. In addition to noting ids of its neighbors, a sensor also *counts the number of times it received a particular id*; the larger this number, the higher is the probability of successful transmission from that node to i. To make it precise, let $S_{ij}(n)$ denote the number of times sensor j successfully received i till time n. Then the following holds.

Proposition 5.3: Let $0 < \alpha_i < 1$ for each i. Then $G_n \to G_{R_0}$ and $\frac{S_{ij}(n)}{n} \to p_{ij}(\underline{\alpha})$ with probability 1.

Proof: See Section IX. □

The convergence of the discovery process is in itself an interesting problem since how fast G_n converges to G_{R_0} or in other words how fast sensors "discover" each other depends on $\underline{\alpha}$. Though finding an optimal, in the sense of minimizing the discovery time, value of α is difficult, a value which will expedite discovery is of practical importance since it would be pre-configured at the time of deployment; we denote it by α_d (subscript d denotes discovery) [21]. Practically, sensors will carry out the discovery process for either a pre-programmed number of slots, or during the discovery phase they will detect when the graph is connected and then stop. For this discussion we will assume that either G_{R_0} or a connected subgraph of it has been discovered and sensor i has an estimate of $p_{ij}(\underline{\alpha})$ for each (i,j) discovered; j counts the number of times it received the id from i and sends back the number to i; i divides it by the number of slots to form an estimate. Algorithm 1 can be distributed [24]. The algorithm works by iteratively forming node clusters, detecting cycles and selecting minimum weight incoming edge to a cluster in a distributed fashion. We omit the details.

C. Results

The setup is as explained in Section IV. Locations for 1000 sensors are realized from a planar Poisson process of $\lambda = 1$ per m^2. In this set of results, we use the same value of attempt probability for each sensor. Further, two "types" of α's need to be distinguished. The first, is α_d, the attempt probability sensors use while *discovering the topology*. We use $\alpha_d = 0.05$. For this value of α_d, the discovered graph is connected within 500 slots. Fig. 3 shows G_{500}; recall that G_n denotes the discovered graph at slot n. Fig. 4 and Fig. 5 respectively show MAWSS and SYMMAWSS constructed from G_{500}; MAWSS here refers to the graph obtained from Algorithm 1.

Once the topology formation is complete, sensors switch to an "operational value" of the attempt probability. Fig. 6 shows the variation of average saturation throughput of a sensor with the operational values of α for network topologies given by G_{500}, G_{1000}, MAWSS and SYMMAWSS. Recall that, for a given topology G' and $\underline{\alpha}$, $M(G', \underline{\alpha})$ denotes the network throughput. The average saturation throughput which we plot in Fig. 6 is simply $\frac{M(G', \underline{\alpha})}{N}$. Note that the throughput of G_{1000} is lower than

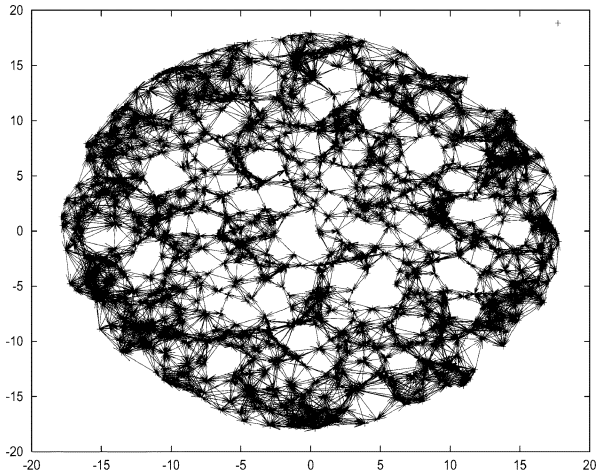

Fig. 3. Topology discovered till 500 slots (G_{500}) by 1000 sensors with $\lambda = 1$ per m^2 and $\alpha_d = 0.05$. x and y axis scale is distance in m.

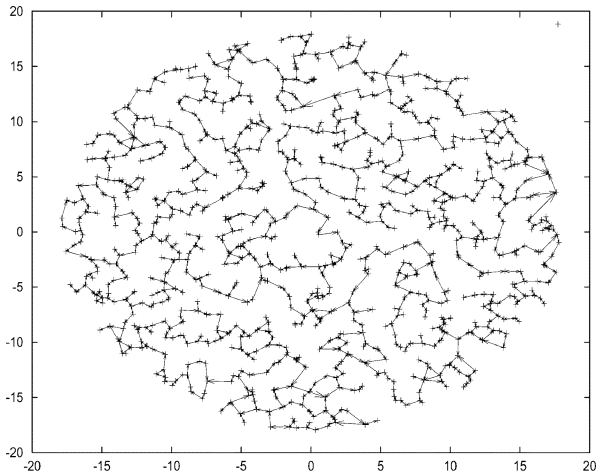

Fig. 4. MAWSS constructed from the discovered topology G_{500} of 1000 sensors with $\lambda = 1$ per m^2 and $\alpha_d = 0.05$. x and y axis scale is distance in m.

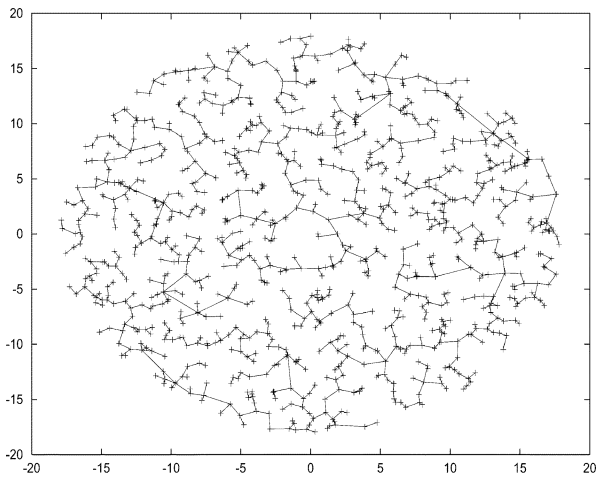

Fig. 5. SYMMAWSS constructed from the discovered topology G_{500} of 1000 sensors with $\lambda = 1$ per m^2 and $\alpha_d = 0.05$. x and y axis scale is distance in m.

G_{500} since it includes more edges of low probability of success discovered during additional 500 slots. MAWSS and SYMMAWSS, on the other hand, eliminate edges with low probability of success while maintaining the connectivity; hence, the maximum throughput achieved by them is almost 5 times of the corresponding discovered graphs. Since symmetric edges are considered in SYMMAWSS, there is slight throughput reduction for SYMMAWSS as compared to MAWSS.

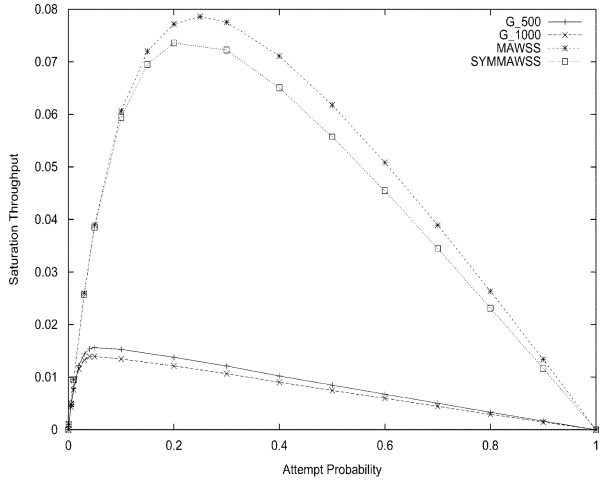

Fig. 6. Saturation throughput variation with attempt probability (α) for different topologies discovered using $\alpha_d = 0.05$: G_{500}, G_{1000}, MAWSS, and SYMMAWSS.

VI. Optimal Attempt Probabilities

A crucial observation from Fig. 6 is that the throughput is maximized at a different operational value of α than α_d. For example, the throughput is maximized at $\alpha = 0.25$ for MAWSS and is almost four times of that obtained with $\alpha_d = 0.05$, thereby, implying that it is indeed essential to operate at such a value. However, as argued in Section IV a distributed implementation warrants optimization over a vector of attempt probabilities, $\underline{\alpha}$. This leads to a peculiar problem that $\underline{\alpha}$ which maximizes the average throughput is degenerate, i.e., some sensors have $\alpha_i = 1$ and the remaining have $\alpha_i = 0$ [21]. This necessitates a new look at the problem formulation. Consider a sensor network in which N sensors are connected to a fusion center. The fusion center combines all the sensor readings into a computation of a statistic of the process being monitored; e.g., the time-varying average. Now the number of computations it can complete in time T equals the minimum of the number of readings of each sensor collected by it in that duration. It follows that the computing rate equals the minimum sensor throughput. In a spatially distributed sensor network in which the global computation is arranged on a tree (as in MAWSS), a sensor will update and transmit its result after receiving corresponding results/measurements from its children. Thus it locally acts a fusion center and the rate at which it can complete its local computation is constrained by the minimum of its children's throughputs. Thus in many computing scenarios it is imperative to improve the minimum of sensor throughputs since few faster (in the sense of throughput) sensors cannot in general expedite the convergence of a global objective. This motivates the problem of *maximizing the minimum of sensor throughputs*.

In order to get some insight into the throughput functions, $M_i(G, \underline{\alpha})$, recall that $N_i(G)$ (respectively $n_i(G)$) denotes the

set of neighbors (respectively number of neighbors) of i. Let $\underline{\alpha}^{ij}$ denote the vector $\underline{\alpha}$ with entries α_i and α_j omitted.

Proposition 6.1: For a fixed topology G, η and β, $M_i(G,\underline{\alpha}) = \frac{1}{n_i(G)}\sum_{j\in N_i(G)}\alpha_i(1-\alpha_j)g_{ij}(\underline{\alpha}^{ij})$. For each $j \in N_i(G), g_{ij}(.)$ either equals 1 or there exists a set $I_{ij} \subseteq V_s\backslash\{i,j\}$ such that $g_{ij}(.)$ is a decreasing and affine function of $\alpha_k, k \in I_{ij}$ and does not depend upon $\alpha_k, k \notin I_{ij}$. Moreover, $g_{ij}(\underline{1}) = 0$ and $g_{ij}(\underline{0}) = 1$.

Proof: See Section IX. □

Definition 6.1: Let the sensor locations be fixed. Then for fixed β, η and G, j is called an interferer of i if $M_i(G,\underline{\alpha})$ is decreasing in α_j. j is called a primary interferer of i if $M_i(G,\underline{\alpha}) = 0$ whenever $\underline{\alpha}$ is such that $\alpha_j = 1$. □

Thus, a neighbor of sensor i is also its interferer. Further, sensor j is a primary interferer of i if it must be silent, i.e., in receive mode, for any neighbor of i to receive successfully from i. Denote by I_i the set of interferers of sensor i. Let $S_j = \{i | j \in I_i\}$. Note that S_j includes sensors which have sensor j as their neighbor.

A. The MAXMIN Throughput

For a given network topology G, consider the following optimization problem.

$$\max_{\underline{\alpha}\in[0,1]^N} \min_{1\le i\le N} M_i(G,\underline{\alpha}) \tag{5}$$

It is clear from Proposition 6.1 that $M_i(G,.), 1 \le i \le N$ are continuous functions of $\underline{\alpha}$, and so is $\min_i M_i(G,.)$. Therefore, an optimum exists for the MAXMIN problem (5) by Weierstrass Theorem. Since topology G is fixed, henceforth we suppress it from the notation. It is, however, assumed that G is connected. Let $\underline{\alpha}^*$ denote an optimum of MAXMIN and M^* denote $\min_i M_i(\underline{\alpha}^*)$. We will call $\underline{\alpha}^*$, the *MAXMIN throughput attempt probabilities* (MMTAP). By $\underline{0} < \underline{\alpha}^* < \underline{1}$, we mean $0 < \alpha_i < 1, i = 1,2,\ldots,N$.

Proposition 6.2: If every sensor is a primary interferer of at least one sensor, then $\underline{0} < \underline{\alpha}^* < \underline{1}$.

Proof: If $\alpha_i^* = 0$ for some i then clearly $M_i(\underline{\alpha}^*) = 0$. If $\alpha_i^* = 1$ for some i then $M_j(\underline{\alpha}^*) = 0, i \in P_j$ where P_j are the primary interferers of j. Proposition 6.1 implies that if $\alpha_i \in (0,1)$ for all i, $M_i(G,\underline{\alpha}) > 0, 1 \le i \le N$. Hence, $\underline{0} < \underline{\alpha}^* < \underline{1}$. □

Consider first N collocated sensors; by collocated we mean that in any slot at most one transmission can be successful. Then $M_i(\underline{\alpha}) = \alpha_i \Pi_{j\neq i}(1-\alpha_j), 1 \le i \le N$ and $\alpha_i^* = \frac{1}{N}$ which is an intuitive and desirable operating for sensors in this scenario. Secondly, even when sensors are spatially distributed, $\underline{\alpha}^*$ equalizes the throughputs, i.e.,

Proposition 6.3: $\underline{0} < \underline{\alpha}^* < \underline{1} \Rightarrow M_i(\underline{\alpha}^*) = M_j(\underline{\alpha}^*), 1 \le i,j \le N$.

Proof: See Section IX. □

The *throughput equalizing property* makes the MMTAP particularly important since with MMTAP sensors operate at equal processing rates which is desirable in applications where computations are iterative.

B. A Generalized Gradient MMTAP Algorithm

Consider an iterative scheme to tune $\underline{\alpha}$ to the MMTAP in Algorithm 2.

Algorithm 2 An MMTAP Algorithm Using Generalized Gradient Ascent

$$\alpha_j(0) \in [0,1], j = 1,2,\ldots,N$$
$$u(k) = \min_{1\le i\le N} M_i(\underline{\alpha}(k)), \quad k \ge 0$$
$$U(k) = \{i | 1 \le i \le N, M_i(\underline{\alpha}(k)) = u(k)\}$$
$$\alpha_j(k+1) = \Pi\left[\alpha_j(k) + \frac{a_j(k)}{|U(k)|}\sum_{i\in U(k)}\frac{\partial M_i(\underline{\alpha}(k))}{\partial \alpha_j}\right] \quad j = 1,2,\ldots,N$$

Π denotes projection on $[0,1]$ and $|U(k)|$ the cardinality of set $U(k)$. $a_j(k)$ is the step size in the kth iteration at sensor j. Algorithm 2 is a "generalized gradient ascent" algorithm; $\frac{1}{|U(k)|}\sum_{i\in U(k)}\frac{\partial M_i(\underline{\alpha}(k))}{\partial \alpha_j}$ being a generalized gradient of $\min_i M_i(\underline{\alpha}(k))$ at $\underline{\alpha}(k)$ [25]. Informally the iterations can be explained as follows. $U(k)$ denotes the set of sensors whose throughput is the minimum under operating point $\underline{\alpha}(k)$. If $j \notin U(k)$, then α_j is reduced in the next iteration since $\frac{\partial M_i(\underline{\alpha})}{\partial \alpha_j} < 0, i \neq j$ (see Proposition 6.1). This leads to an increase in the throughput of $i \in U(k)$. If $j \in U(k)$, then α_j is increased or decreased based on how it affects others and how others affect its throughput. Thus the algorithm tries to equalize as well as maximize the sensor throughputs.

Proposition 6.4: Let $a_j(k) = a(k), 1 \le j \le N, k \ge 0$. If $a(k)$ satisfy $\lim_{k->\infty} a(k) = 0$ and $\sum_{k=0}^{\infty} a(k) = \infty$, then Algorithm 2 converges to the MMTAP.

Proof: See Section IX. □

C. Distributed Stochastic MMTAP Algorithm

Though fixed in form for a given placement of nodes, $M_i(.)$ is not known at sensor i and being a steady-state average, only *noisy measurements* of $M_i(.)$ are available for Algorithm 2. An unbiased estimator of $M_i(.)$, denoted by $\hat{M}_i(.)$, is $\frac{1}{\tau}\sum_{j=1}^{\tau} X_i(j)$ where $X_i(j) = 1$ if i transmits successfully in slot j and 0 otherwise. τ is the number of estimation slots. Sensors also need to *estimate the gradient* of $M_i(.)$ in order to use Algorithm 2. Since we need a distributed algorithm we consider simultaneous perturbation (SP, [26]) based gradient estimation. Instead of perturbing one component, i.e., α_i at a time to obtain the estimates of partial derivatives, in SP all α_is can be perturbed simultaneously given that perturbations for each α_i are zero mean independent random variables with some additional conditions [26]. This way, *by choosing the perturbation amount locally, sensors can simultaneously estimate the derivatives*. In the kth iteration, let $\underline{\Delta}(k)$ denote a vector of N independent Bernoulli random variables taking values in $\{-1,1\}$ such that $\{\underline{\Delta}(k)\}$ is an independent sequence with $\underline{\Delta}(k)$ independent of $\underline{\alpha}(0), \underline{\alpha}(1), \ldots, \underline{\alpha}(k)$. Then

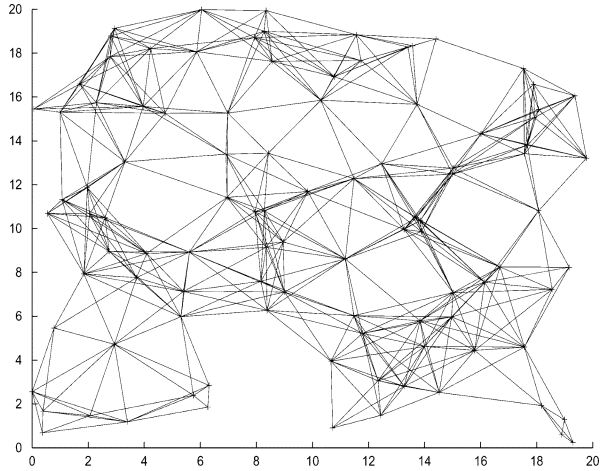

Fig. 7. Transmission graph, G_{R_0}, of a random 100 node sensor network.

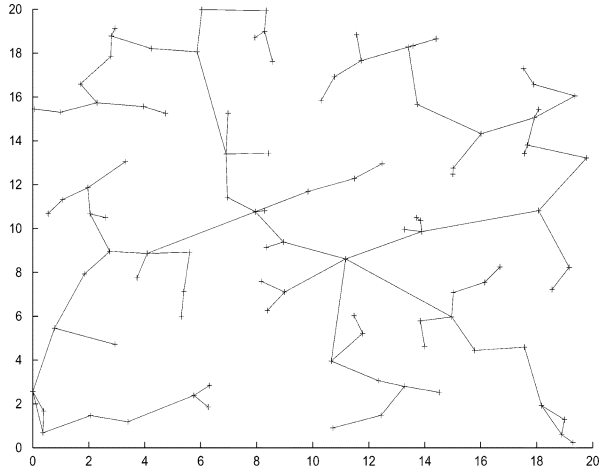

Fig. 8. SYMMAWSS of the sensor network in Fig. 7.

the central-difference estimator of $(\partial M_i(\alpha)(k))/(\partial \alpha_j)$ is $(\hat{M}_i(\underline{\alpha}(k)+c(k)\underline{\Delta}(k))-\hat{M}_i(\underline{\alpha}(k)-c(k)\underline{\Delta}(k)))/(2c(k)\Delta_j(k))$ where $c(k)$ is a scalar. SP requires $c(k) \rightarrow 0$ so that the estimator is asymptotically unbiased.

Proposition 6.5: In Algorithm 2, let the partial derivatives of $M_i(.), 1 \leq i \leq N$ be replaced by their estimates (biased or unbiased). Let $a_j(k) = a(k), 1 \leq j \leq N, k \geq 0$ and $a(k)$ satisfy $\sum_{k=1}^{\infty} a(k) = \infty$ and $\sum_{k=1}^{\infty} a(k)^2 < \infty$. Then the generated sequence $\{\underline{\alpha}(k), k \geq 1\}$ converges a.s. to the MMTAP.

Proof: See Section IX. □

For a complete distributed implementation we now only need a way of obtaining an estimate of $(\partial M_i(\underline{\alpha}(k)))/(\partial \alpha_j)$ for each $i \in U(k)$ at every sensor j in iteration k. This itself can be arranged as a computation on the underlying MAWSS tree. See [21] for the details.

D. Results

We consider a network of 100 sensors, each of transmission range 4 m, with locations realized randomly in a square field of area 400 m^2. The MMTAP are unknown for this network.[6] Fig. 8 shows the G_{R_0} and SYMMAWSS corresponding to it; scale in Fig. 8 is distance in m. $\eta = 4$, $\beta = 7$ dB and $\tau = 1000$ slots. $\alpha_i(0) = 0.1, 1 \leq i \leq 100$. We choose $a(k) = (0.1)/((k+1)^{0.7})$ and $c(k) = (0.1)/((k+1)^{0.15})$. Fig. 8 shows the evolution of the minimum sensor throughput in the network with iterations; the actual estimates of sensor throughputs are used and the resulting graph is smoothed by 5-point adjacent averaging to show the trend. The algorithm appears to have reached a throughput value of 0.032–0.035 packets/slot starting from 0.015 packets/slot, a more than 100% increase.

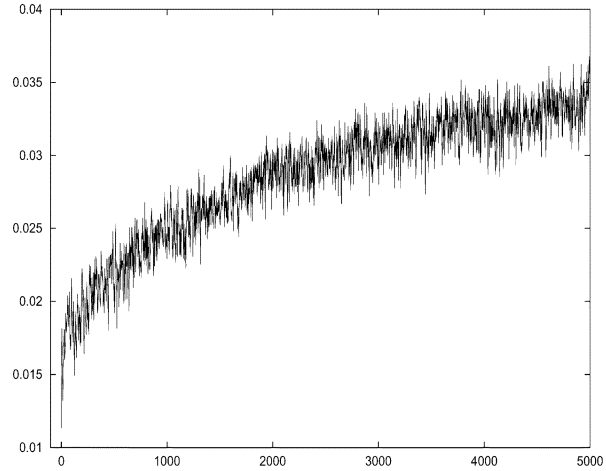

Fig. 9. Evolution of the *estimated* minimum sensor throughput in the 100 node sensor network.

VII. Discussion

It is essential that after their deployment, sensors organize into an optimal network as fast as possible. This is particularly true of the network topology. The time (and message) complexity of the distributed MAWSS algorithm discussed in Section V-B which finds N branchings equals $O(N^2)$ ([24]). This cost appears to be imperative for forming a throughput optimal topology. In return, as seen from Fig. 6, the performance gain is substantial. Algorithm 1 (and its distributed version) constructs directed trees rooted at each sensor, which can be used in a variety of computational algorithms (e.g., MMTAP) and for control information propagation. We expect that our approach can be also extended directly to additional topological requirements such as k-connectedness. Learning an optimal $\underline{\alpha}$ is an important but much harder problem. Our algorithm is fairly simple and makes use of measurements made locally. Its major complexity is in obtaining the estimates of partial derivatives of throughputs at each sensor. Being constrained by the bias-variance dilemma [27] convergence of our stochastic algorithms can be improved by careful selection of the parameters. It is also possible for the sensors to choose a good starting point from the knowledge of the probability of successful transmission obtained during the discovery phase. Moreover, Fig. 9 indicates that the improvement within few iterations may be significant.

[6]We have verified the MMTAP algorithm on networks simple enough to deduce $M_i(.)$'s (hence the MMTAP) easily [21].

The most important point regarding our algorithms is that they are *measurement-based not model-based*. This means that instead of particular analytical forms, they employ estimates of sensor throughputs or probabilities of success over different links obtained through on-line measurements. Therefore, the assumptions in Section III regarding communication (e.g., SIR-based decoding, communication without multipath effects, etc.) and computation (e.g., transmission to each neighbor with equal probability, etc.) are not essential for the algorithms *per se*. Moreover, it may be possible to extend our approach to other access schemes as well. Interestingly, these algorithms can be seen as a tool by which the *network slowly and continuously keeps on improving and adapting itself*. This aspect is particularly important from the point of view of device failures and task reconfiguration. The other important advantage of our MMTAP algorithm is that the throughput at each sensor is measured using real transmissions and no special packet transmissions are required. Hence, there is no extra energy consumption. Further, they can work even in the presence of energy saving techniques such as random sleep time and can account for energy constraints directly, for example, by upper bounding the attempt probabilities. In [21], we apply our algorithms to a realistic scenario in which sensors, by organizing themselves optimally, are able to compute accurately as well as efficiently, i.e., at higher sampling rates with low computational delay.

We designed algorithms so as to achieve the optimal performance and found correspondingly higher algorithmic complexity. Our future work, therefore, is to develop asynchronous algorithms with strictly local information exchange for scalability. This paper lends support to any such effort since it shows a way to obtain globally optimal performance against which the performance of other algorithms can be compared.

VIII. Conclusion

We viewed performance optimization as the objective for self-organization in sensor networks and argued that the rate at which a sensor network can process data in a distributed fashion is governed by its communication throughput; hence sensors must organize themselves in such a way as to optimize their communication throughput. Using a simple model, we showed that the network topology and the transmission attempt rate are the critical factors which determine the throughput. We obtained an optimal topology by the maximum average weight (MAWSS) formulation and optimal attempt probabilities by maximizing the minimum sensor throughput (MMTAP). The MMTAP were found to have an important throughput-equalizing property. The MAWSS algorithm is distributed and uses connectivity and probability of successful transmission information obtained locally. We presented a synchronous distributed stochastic algorithm for driving a sensor network to the MMTAP. The algorithm uses local throughput measurements and yields substantial performance improvement even within few iterations. The performance improvement is at the price of algorithmic complexity. However, this work shows that the performance gains from optimal self-organization can be substantial and the adaptive techniques discussed in this paper need to be considered during the protocol design.

IX. Proofs

Definition 9.1: A function $f : R^n \to R$ is called strictly quasiconcave if $f(\lambda x + (1-\lambda)y) > \min\{f(x), f(y)\}$ for $\lambda \in (0,1)$. □

Lemma 9.1: Consider function $f(x) = xg(x)$ where $x \in [0,1]$ and $g(x)$ is a strictly decreasing function of x. Then f is strictly quasiconcave.

Proof: We show that for $x, y \in (0,1)$, $f(y) > f(x) \Rightarrow f'(x)(y-x) > 0$. $f(y) > f(x)$ means $yg(y) > xg(x)$. f' denotes the derivative, $f'(x) = xg'(x) + g(x)$. Suppose $x > y$. Then, $xg(y) > yg(y) > xg(x)$. Thus, $x(g(y) - g(x)) > g(y)(x-y)$. Now $y \to x \Rightarrow -xg'(x) > g(x)$. Thus $g(x) + xg'(x) < 0$ which implies $(xg'(x) + g(x))(y-x) > 0$. If $x < y$, $yg(y) > xg(x) > xg(y)$. This, similar to the previous derivation, yields $g(x) + xg'(x) > 0$. Hence, $(xg'(x) + g(x))(y-x) > 0$. The strict inequality in one variable case implies strict quasi-concavity. □

Proof of Proposition 4.1: Recall that $\alpha_i = \alpha$ for all i. Note that (4), is of the form $\alpha g(\alpha)$ where $g(\alpha)$ combines the "receiving part" $(1-\alpha)$ and the "interference part" (see (3)). $g(\alpha)$ is strictly decreasing in α; $g(1) = 0$, $g(0) = 1$ and $g(\alpha) > 0$, $\alpha \in (0,1)$. Therefore, by Lemma 9.1 $M_i(G,\alpha)$ is strictly quasiconcave. The proof concludes by noting that for $G \in G_{\text{cs}}$ $M(G,\alpha) = \sum_{i=1}^{N} M_i(G,\alpha)$ has the same form $\alpha g(\alpha)$. □

Lemma 9.2: Consider the problem of maximizing $f(\underline{k}) = \frac{1}{k_1+1} + \frac{1}{k_2+1} + \cdots + \frac{1}{k_p+1}$ subject to the constraints that $k_i \in \{0,1,2,3\}$ for $1 \le i \le p$, and $k_1 + k_2 + \cdots + k_p = 3q$ $(p > q)$. Then $\max f(\underline{k}) = p - \frac{3q}{4}$ and is achieved by a $\underline{k}$ which has q k_is equal to 3 and the rest p-q equal to 0.

Proof: Relax the integer constraints to $0 \le k_i \le 3$ for $i = 1, \ldots, p$. Denote by $\underline{k}^*$ any $\underline{k}$ which has q k_is equal to 3 and the rest $p-q$ equal to 0 and by $\bar{\underline{k}} = (\frac{3q}{p}, \ldots, \frac{3q}{p})$. Then $(\underline{k}^*, \frac{1}{16})$ and $(\bar{\underline{k}}, \frac{p^2}{(3q+p)^2})$ are the stationary points of the Lagrangian of the relaxed problem. Single (linear) constraint implies that either of the two must be the maximum. $f(\underline{k}^*) = p - \frac{3q}{4} > f(\bar{\underline{k}}) = \frac{p^2}{3q+p}$. Moreover, $\underline{k}^*$ is an integer solution which implies that it is the optimal solution of the problem with integer constraints as well. □

Proof of Proposition 5.1: The proof for undirected graphs uses transformation from 3DM (3-dimensional matching, which is known to be NP-complete [28]), to MAWSS. An MAWSS instance is a weighted connected graph $G = (V, E, W)$, and a positive integer B and the question we ask is, is there a graph $G_1 \in G_{\text{cs}}$ such that $M(G_1) \ge B$? On the other hand, a 3DM instance is a set $S \subseteq X \times Y \times Z$, $|S| = p$, where X, Y and Z are disjoint sets having the same number q of elements and the question is does S contain a matching?

Consider the gadget in Fig. 10. R_x, X, R_y, Y, R_z and Z denote the sets of q vertices each whereas A is a set of p vertices $(p > q)$. q vertices in R_x and X are paired; the corresponding vertices have an edge of weight 1. Similarly for R_y, Y and R_z, Z. R contains a single vertex which has one edge of weight 1 from each vertex in A. The sets X, Y, Z and S in an instance of 3DM correspond to the set of vertices X, Y, Z and A respectively in Fig. 10. Thus, for each (x,y,z) in S, there are edges from vertices $x \in X$, $y \in Y$ and $z \in Z$ to a vertex in A and $|A| = p = |S|$. Each edge between a vertex in X (similarly

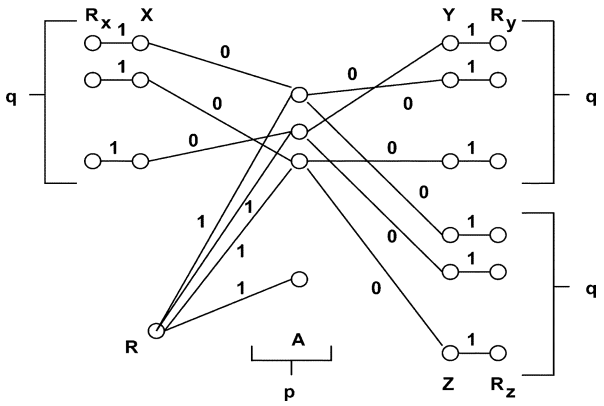

Fig. 10. Gadget for proving NP-completeness of undirected MAWSS.

in Y and Z) and a vertex in A has weight 0. We denote the resulting graph by G. In G, each vertex in R_x, R_y, R_z and R has an average weight of 1. The vertices in X, Y and Z have the average weights according to the connections resulting from S. Each vertex in A has 4 edges (one each from a vertex in X, Y, Z and R) and an average weight of $\frac{1}{4}$. The claim is that a matching exists if and only if G has a connected spanning subgraph with $M = (1+3q) + \frac{3q}{2} + (p - \frac{3q}{4})$.

Suppose that a matching exists. Then consider subgraph G_1 of G such that out of p, q vertices in A have 3 edges each of weight 0 from X, Y and Z corresponding to the matching, 1 edge to R and the rest $p-q$ vertices have only 1 edge connected to R. Thus, the sum of the average weights of vertices in X (similarly Y and Z) is $\frac{q}{2}$ whereas that in A is $(p-q)+q\times\frac{1}{4} = p - \frac{3q}{4}$. It follows that G_1 is a connected graph with $M = (1+3q) + \frac{3q}{2} + (p - \frac{3q}{4})$.

For the converse, note that the subgraphs of G in which each vertex in X, Y and Z has exactly one edge connected to a vertex in A are connected. Under such a condition the vertices in X, Y and Z attain their maximum average weight of $\frac{1}{2}$. It is easy to see that only such a subgraph of G will maximize M over G_{cs}. Now M for these subgraphs will be determined by their connections at the vertices in A. Let k_i denote the number of edges between a vertex $i \in A$ and X, Y and Z; by our construction $0 \leq k_i \leq 3$ and distinct vertices in X (similarly in Y and Z) are connected to distinct vertices in A. Hence, from Lemma 9.2 it follows that the assignment of edges between A and X, Y and Z which maximizes M is such that out of p vertices in A, q have 3 edges (one each from X, Y and Z) while the rest $(p-q)$ have none connected to X, Y and Z. The subgraph with this particular arrangement of edges among A, X, Y and Z has the maximum $M(= (1+3q) + \frac{3q}{2} + (p - \frac{3q}{4}))$ over G_{cs}. Therefore, if there is a connected subgraph of G with $M = (1+3q) + \frac{3q}{2} + (p - \frac{3q}{4})$, then the matching exists: q points with edges from X, Y, Z is the required matching. 3DM is, thus, reducible to MAWSS proving NP-completeness of MAWSS for undirected graphs. NP-completeness of MAWSS for directed graphs is established by noting that STA (strong connectivity augmentation, [29]) is its special case. □

Proof of Proposition 5.3: Since $0 < \alpha_i < 1$ for each i, $p_{ij}(\underline{\alpha}) > 0$ for each $(i,j) \in E_{R_0}$. Therefore, the probability that (i,j) is discovered in finite time is 1. Since N is finite, $G_n \to G_{R_0}$ in finite time with probability 1. The second limit follows from the strong law of large numbers. □

Proof of Proposition 6.1: Let the sensor locations be fixed. Recall (3) and (4). Note that, $g_{ij}(\underline{\alpha}^{ij})$ is $P(\Gamma_{ij} \geq \beta)$ with d_{kj} fixed; recall that Γ_{ij} denotes the SIR of a transmission from i to j. If $\frac{(\frac{d_{ij}}{d_0})^{-\eta}}{\sum_{k\neq i,j}(\frac{d_{kj}}{d_0})^{-\eta}+N_0} \geq \beta$, i.e., the SIR of a transmission from i to j is above the required threshold even if all the other sensors are simultaneously transmitting, then clearly $g_{ij}(\underline{\alpha}^{ij}) = 1$. If not, let $\underline{v}$ denote an N dimensional vector each component of which is either 0 or 1. Let, $\mathcal{V} = \{\underline{v} : \frac{(\frac{d_{ij}}{d_0})^{-\eta}}{\sum_{k\neq i,j}(\frac{d_{kj}}{d_0})^{-\eta v_k}+N_0} \geq \beta\}$. Then, $P(\Gamma_{ij} \geq \beta) = \sum_{\underline{v}\in\mathcal{V}} \Pi_{k\neq i,j} \alpha_k^{v_k}(1-\alpha_k)^{(1-v_k)}$ Let $\underline{v}_{-m}$ denote a vector with mth entry omitted and let $(\underline{v}_{-m}, v_m)$ represent $\underline{v}$. Then for each $k \neq i,j$, $P(\Gamma_{ij} \geq \beta) = \alpha_k a_k + (1-\alpha_k)b_k$, where $a_k = \sum_{\underline{v}\in(\underline{v}_{-k},1)} \Pi_{l\neq i,j,k} \alpha_l^{v_l}(1-\alpha_l)^{(1-v_l)}$ and $b_k = \sum_{\underline{v}\in(\underline{v}_{-k},0)} \Pi_{l\neq i,j,k} \alpha_l^{v_l}(1-\alpha_l)^{(1-v_l)}$. If for every $(\underline{v}_{-k},0) \in \mathcal{V}$, $(\underline{v}_{-k},1) \in \mathcal{V}$, then $a_k = b_k$. Hence, $P(\Gamma_{ij} \geq \beta)$ does not depend on k. Let I_{ij} be the set of sensors for which this condition fails. Then for $k \in I_{ij}$, $b_k > a_k$ since if $(\underline{v}_{-k},1) \in \mathcal{V}$ then so is $(\underline{v}_{-k},0)$, i.e., each term in the sum for a_k is also in b_k. Thus it follows that $g_{ij}(\underline{\alpha}^{ij})$ is decreasing and affine in $\alpha_k, k \in I_{ij}$. □

Lemma 9.3: If $\underline{0} < \underline{\alpha}^* < \underline{1}$ then $(\partial M_i(\underline{\alpha}^*))/(\partial\alpha_j) = 0$ if and only if $i \notin S_j$.

Proof: Follows from Proposition 6.1. □

Proof of Proposition 6.3: Note that MAXMIN is equivalent to the constrained optimization problem of maximizing x subject to $M_i(\underline{\alpha}) \geq x, i = 1,2,\ldots,N$, $x \geq 0$, and $\alpha_i \in [0,1], i = 1,2,\ldots,N$. Let $\tilde{x} := (x, \underline{\alpha})$. By hypothesis $\underline{0} < \underline{\alpha}^* < \underline{1}$. Hence $x^* = M^* > 0$. Moreover, the Mangasarian–Fromowitz constraint qualification holds at $\tilde{x}^*$. Therefore, by KKT Theorem ([30]) there exists a vector of multiplier, $\underline{\mu}^* \geq \underline{0}$, such that $\mu_i^*(M_i(\underline{\alpha}^*) - x^*) = 0$, and

$$\frac{\partial L(\tilde{x}^*, \underline{\mu}^*)}{\partial x} = 1 - \sum_{i=1}^{N} \mu_i^* = 0 \tag{6}$$

$$\frac{\partial L(\tilde{x}^*, \underline{\mu}^*)}{\partial \alpha_j} = \sum_{i=1}^{N} \mu_i^* \frac{\partial M_i(\underline{\alpha}^*)}{\partial \alpha_j} = 0. \tag{7}$$

Let $\mu_j^* = 0$ for some j. Then (7) corresponding to j implies that $\sum_{i:i\neq j} \mu_i^* \frac{\partial M_i(\underline{\alpha}^*)}{\partial \alpha_j} = 0$. Recall that I_i denotes the set of interferers of sensor i and $S_j = \{i | j \in I_i\}$. $S_j = \phi$ means that no sensor transmits to j and j is not interferer of any sensors. j is thus an isolated sensor and cannot belong to a connected network. Using Lemma 9.3, (7) reduces to $\sum_{i:i\in S_j} \mu_i^* \frac{\partial M_i(\underline{\alpha}^*)}{\partial \alpha_j} = 0$. For each such i, $\frac{\partial M_i(\underline{\alpha}^*)}{\partial \alpha_j} < 0$ and $\mu_i^* \geq 0$. It follows that for all $i \in S_j$, $\mu_i^* \frac{\partial M_i(\underline{\alpha}^*)}{\partial \alpha_j} = 0$ and therefore $\mu_i^* = 0$ i.e., for each sensor i that transmits to j, or is interfered with by j, $\mu_i^* = 0$. Continuing the argument for each such i and further, let A denote the final set $\{i | \mu_i^* = 0\}$. Let $B = \{k | k \notin A\}$. Then any such k does not transmit to any node in A and no sensor in A is an interferer of k. Since G is a (strongly) connected topology, this implies that for all $k \in B, k \notin G$. Thus, for all $i \in G$, $\mu_i^* = 0$. This contradicts (6). Therefore, $\mu_i^* > 0, 1 \leq i \leq N$ implying that $M_i(\underline{\alpha}^*) - M^* = 0, 1 \leq i \leq N$. □

Definition 9.2: [31] A function $f : R^n \to R$ is called generalized differentiable at $x \in R^n$ if in a vicinity of x there exists upper semicontinuous multivalued mapping $\bar{\partial}$ with convex compact values $\bar{\partial} f(x)$ such that, $f(y) = f(x) + g^T(y - x) + o(x, y, g)$, where $g \in \bar{\partial} f(x)$ and $\lim_k \frac{|o(x,y(k),g(k)|}{\|y(k)-x\|} = 0$ for any sequences $y(k) \to x$, $g(k) \to g$, $g(k) \in \bar{\partial} f(y(k))$. □

Consider the problem of minimizing a generalized differentiable function $f(x)$ over $X \subseteq R^n$. Let $X^* := \{x \in X | 0 \in \bar{\partial} f(x)\}$ and $f^* := \{f(x) | x \in X^*\}$.

Lemma 9.4: The cluster points of the iterative scheme [25]

$$x(k+1) = \Pi[x(k) - a(k)g(k)], k \geq 0$$

where Π denotes the projection X, $a(k)$ are nonnegative numbers, $x(0) \in X$ and $g(k) \in \bar{\partial} f(x(k))$, belong to a connected subset of X^* and $\{f(x(k))\}$ has a limit in f^* if f^* is finite, and $\lim_{k->\infty} a(k) = 0$ with $\sum_{k=0}^{\infty} a(k) = \infty$. □

Proof of Proposition 6.4: $F(\underline{\alpha}) := \min_{1 \leq i \leq N} M_i(\underline{\alpha})$ is generalized differentiable. Let $g(k)$ be a vector whose jth component is $\frac{1}{|U(k)|} \sum_{i \in U(k)} \frac{\partial M_i(\underline{\alpha}(k))}{\partial \alpha_j}$. Then $g(k) \in \bar{\partial} F(\underline{\alpha}(k)), k \geq 0$. Since the constraint set $[0, 1]^N$ is a Cartesian product of sets [0, 1], projection can be obtained component-wise. Convergence follows from Lemma 9.4. □

Proof of Proposition 6.5: See [21] and [25] for details. □

REFERENCES

[1] B. Sinopoli, B. Sharp, C. Schenato, L. Schaffert, and S. Sastry, "Distributed control applications within sensor networks," *Proc. IEEE*, vol. 91, no. 3, pp. 1235–1246, Mar. 2003.

[2] A. Knaian, "A wireless sensor network for smart roadbeds and intelligent transportation systems," M.Eng. thesis, MIT, Cambridge, MA, 2000.

[3] S. Graham and P. R. Kumar, "The convergence of control, communication, and computation," in *PWC 2003*.

[4] D. Baker and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm," *IEEE Trans. Commun.*, vol. 29, no. 11, pp. 1694–1701, Nov. 1981.

[5] M. Post, A. Kershenbaum, and P. Sarachik, "A distributed evolutionary algorithm for reorganizing network communications," in *Proc. IEEE MILCOM*, 1985.

[6] P. Santi, "Topology control in wireless ad hoc and sensor networks," *ACM Comput. Surveys*, vol. 37, no. 2, pp. 164–194, 2005.

[7] R. Krishnan and D. Starobinski, "Message-efficient self-organization of wireless sensor networks," in *Proc. IEEE WCNC 2003*, pp. 1603–1608.

[8] L. Clare, G. Pottie, and J. Agre, "Self-organizing distributed sensor networks," *SPIE—The Int. Soc. for Optical Eng.*, pp. 229–237, 1999.

[9] K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie, "Protocols for self-organization of a wireless sensor network," *IEEE Pers. Commun.*, vol. 7, no. 5, pp. 16–27, Oct. 2000.

[10] A. Cerpa and D. Estrin, ASCENT: Adaptive self-configuring sensor network topologies UCLA, Tech. Rep. UCLA/CSD-TR 01-0009, May 2001.

[11] R. Nagpal, H. Shrobe, and J. Bachrach, "Organizing a global coordinate system from local information on an ad hoc sensor networks," in *2nd Int. Workshop on Information Processing in Sensor Networks (IPSN)*, 2003.

[12] A. Karnik and A. Kumar, "Iterative localisation in ad hoc wireless sensor networks: One-dimensional case," in *SPCOM 2004*.

[13] K. Sohrabi and G. Pottie, "Performance of a novel self-organization protocol for wireless ad hoc sensor networks," *Proc. IEEE VTC 1999*, pp. 1222–1226.

[14] C. Chiasserini and M. Garetto, "Modeling the performance of wireless sensor networks," *Proc. IEEE INFOCOM 2004*, pp. 220–231.

[15] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Gossip algorithms: Design, analysis and applications," *Proc. IEEE INFOCOM 2005*, pp. 1653–1664.

[16] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: A survey," *IEEE Commun. Mag.*, vol. 44, no. 4, pp. 115–121, Apr. 2006.

[17] K. Romer, "Time synchronization in ad hoc networks," *Proc. MobiHoc*, 2001.

[18] A. Ebner, H. Rohling, M. Lott, and R. Halfmann, "Decentralized slot synchronization in highly dynamic ad hoc networks," in *Proc. Int. Symp. Wireless Personal Multimedia Communications*, 2002, pp. 27–30.

[19] A. Domazetovic, L. Greenstein, N. Mandayam, and I. Sekar, "Estimating the Doppler spectrum of a short-range fixed wireless channel," *IEEE Commun. Lett.*, vol. 7, no. 5, pp. 227–229, May 2002.

[20] S. Tilak, N. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of sensor network communication models," *Mobile Comput. Commun. Rev.*, vol. 6, no. 2, Apr. 2002.

[21] A. Karnik, "Optimal self-organization of ad hoc wireless sensor networks," Ph.D. dissertation, Indian Inst. of Science, Bangalore, 2004.

[22] R. E. Tarjan, "Finding optimal branchings," *Networks*, vol. 7, pp. 25–35, 1977.

[23] N. Khude, A. Kumar, and A. Karnik, "Time and energy complexity of distributed computation in wireless sensor networks," in *Proc. IEEE INFOCOM 2005*.

[24] P. Humblet, "A distributed algorithm for minimum weight directed spanning trees," *IEEE Trans. Commun.*, vol. COM-31, no. 6, pp. 756–762, Jun. 1983.

[25] Y. Ermoliev and V. Norkin, Stochastic generalized gradient method with application to insurance risk management Int. Inst. for Applied Systems Analysis, Tech. Rep. IR-97-021, 1997.

[26] J. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Trans. Automat. Contr.*, vol. 37, no. 3, pp. 332–341, Mar. 1992.

[27] B. Bharath and V. Borkar, "Stochastic approximation algorithms: Overview and recent trends," *Sadhana*, vol. 24, pp. 425–452, 1999.

[28] M. Garey and D. Johnson, *Computers and Intractability*. New York: W. H. Freeman, 1979.

[29] G. Frederickson and J. Ja'Ja, "Approximation algorithms for several graph augmentation problems," *SIAM J. Comput.*, vol. 10, no. 2, pp. 270–283, May 1981.

[30] D. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 1995.

[31] V. Norkin, "On nonlocal algorithms for optimization of nonsmooth functions," *Cybernetics*, vol. 14, no. 5, pp. 704–707, 1978.

Aditya Karnik received the B.E. degree in electronics and telecommunications from the University of Pune, Pune, India, and the M.E. and Ph.D. degrees, both in electrical communication engineering, from the Indian Institute of Science, Bangalore, India. During his Ph.D. work, he was a recipient of the IBM Research Fellowship.

He is currently with the Manufacturing Enterprise Modelling group in the General Motors India Science Lab, Bangalore. His research interests are in control and optimization theory in general, and its application to communication and manufacturing systems in particular.

Anurag Kumar (F'06) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, and the Ph.D. degree from Cornell University, Ithaca, NY.

He was with Bell Laboratories, Holmdel, NJ, for over six years. Since 1988, he has been with the Department of Electrical Communication Engineering, Indian Institute of Science (IISc), Bangalore, where he is now a Professor, and is also the Chairman of the department. He is a coauthor of the textbook *Communication Networking: An Analytical Approach* (Morgan Kaufmann, 2004). His area of research is communication networking, specifically, modeling, analysis, control and optimization problems arising in communication networks and distributed systems.

He is a Fellow of the IEEE, the Indian National Science Academy (INSA), and the Indian National Academy of Engineering (INAE). He is an Area Editor for IEEE TRANSACTIONS ON NETWORKING and for *IEEE Communications Surveys and Tutorials*.