Time and Energy Complexity of Function Computation Over Networks

Nikhil Karamchandani, *Student Member, IEEE*, Rathinakumar Appuswamy, *Student Member, IEEE*, and Massimo Franceschetti, *Senior Member, IEEE*

***Abstract*—This paper considers the following network computation problem: n nodes are placed on a $\sqrt{n} \times \sqrt{n}$ grid, each node is connected to every other node within distance $r(n)$ of itself, and it is assigned an arbitrary input bit. Nodes communicate with their neighbors and a designated sink node computes a function f of the input bits, where f is either the *identity* or a *symmetric* function. We first consider a model where links are interference and noise-free, suitable for modeling wired networks. Then, we consider a model suitable for wireless networks. Due to interference, only nodes which do not share neighbors are allowed to transmit simultaneously, and when a node transmits a bit, all of its neighbors receive an independent noisy copy of the bit. We present lower bounds on the minimum number of transmissions and on the minimum number of time slots required to compute f. We also describe efficient schemes that match both of these lower bounds up to a constant factor and are thus jointly (near) optimal with respect to the number of transmissions and the number of time slots required for computation. At the end of the paper, we extend results on symmetric functions to general network topologies, and obtain a corollary that answers an open question posed by El Gamal in 1987 regarding the computation of the *parity* function over ring and tree networks.**

***Index Terms*—Communication complexity, error correction, function computation, information theory, sensor networks.**

I. Introduction

SENSOR networks consist of a large number of small nodes, capable of sensing, processing, and communicating data. These networks are typically required to sample a field of interest, do "in-network" computations, and then communicate a relevant summary of the data to a designated node(s), most often a function of the raw sensor measurements. For example, in environmental monitoring a relevant function can be the average temperature in a region. Another example is an intrusion detection network, where a node switches its value from 0 to 1 if it detects an intrusion and the function to be computed is the maximum of all the node values. For schemes that perform such in-network aggregation, two relevant measures of complexity are the latency, i.e., the time it takes to perform the computation, and the corresponding energy spent.

Network computation has been studied extensively in the literature, under a wide variety of models. In wired networks with point-to-point noiseless communication links, computation has been traditionally studied in the context of communication complexity [1]. Wireless networks, on the other hand, have three distinguishing features: the inherent *broadcast* medium, *interference*, and *noise*. Due to the broadcast nature of the medium, when a node transmits a message, all of its neighbors receive it. Due to noise, the received message at each neighbor is a noisy copy of the transmitted one. Due to interference, simultaneous transmissions can lead to message collisions. To avoid interference, a simple *protocol model* introduced in [2] allows only nodes which do not share neighbors to transmit simultaneously. The works in [3]–[5] study computation restricted to the protocol model of operation and assuming noiseless transmissions. A noisy communication model over independent binary symmetric channels was proposed in [6] in which when a node transmits a bit, all of its neighbors receive an independent noisy copy of the bit. Using this model, the works in [7]–[9] consider computation in a *complete network* where each node is connected to every other node and only one node is allowed to transmit at any given time. An alternative to the complete network is the random geometric network in which n nodes are randomly deployed in continuous space inside a $\sqrt{n} \times \sqrt{n}$ square and each node can communicate with all other nodes within a range[1] r. Computation in such networks under the protocol model of operation and with noisy communication has been studied in [10]–[13]. In these works the connection radius r is assumed to be of order[2] $\Theta\left(\sqrt{\log n}\right)$, which is the threshold required to obtain a connected random geometric network, see [14, Chapter 3].

In this paper, we consider the class of grid geometric networks in which every node in a $\sqrt{n} \times \sqrt{n}$ grid is connected to every other node within distance r from it, see Fig. 1. This construction has many useful features. By varying the connection radius we can study a broad variety of networks with contrasting structural properties, ranging from the sparsely connected grid network for $r = 1$ to the densely connected complete network

Manuscript received September 23, 2009; revised May 20, 2011; accepted July 12, 2011. Date of current version December 07, 2011. This work was supported in part by the National Science Foundation (NSF CNS-0916778, CNS 0546235, and CCF-0916465) and in part by the Center for Wireless Communications, University of California San Diego.

The authors are with the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093-0407 USA (e-mail: nikhil@ucsd.edu; rathnam@ucsd.edu; massimo@ece.ucsd.edu).

Communicated by R. A. Berry, Associate Editor for Communication Networks.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2011.2168902

[1]The connection radius r can be a function of n, but we suppress this dependence in the notation for ease of exposition.

[2]Throughout the paper we use the following subset of the Bachman-Landau notation for positive functions of the natural numbers: $f(n) = O(g(n))$ as $n \to \infty$ if $\exists k > 0, n_0 : \forall\, n > n_0 f(n) \leq k g(n)$; $f(n) = \Omega(g(n))$ as $n \to \infty$ if $g(n) = O(f(n))$; $f(n) = \Theta(g(n))$ as $n \to \infty$ if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. The intuition is that f is asymptotically bounded up to constant factors from above, below, or both, by g.

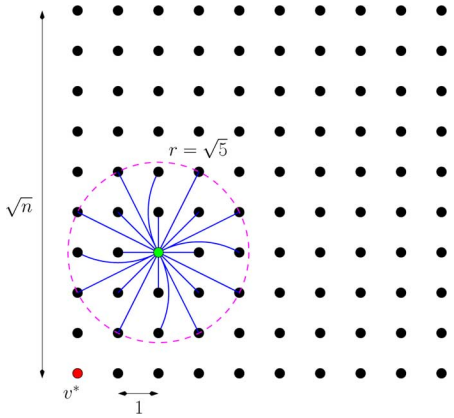

Fig. 1. Network $\mathcal{N}(n, r)$: each node is connected to all nodes within distance r. The (red) node v^* is the sink that has to compute a function f of the input.

when $r \geq \sqrt{2n}$. This provides intuition about how network properties like the average node degree impact the cost of computation. Above the critical connectivity radius for the random geometric network $r = \Theta\left(\sqrt{\log n}\right)$, the grid geometric network has structural properties similar to its random geometric counterpart and all the results in this paper also hold in that scenario. Thus, our study includes the two network structures studied in previous works as special cases. At the end of the paper, we also present some extensions of our results to arbitrary network topologies.

We consider both noiseless wired communication over binary channels and noisy wireless communication over binary symmetric channels using the protocol model. We focus on computing two specific classes of functions with binary inputs, and measure the latency by the number of time slots it takes to compute the function and the energy cost by the total number of transmissions made in the network. The *identity* function (i.e., recover all source bits) is of interest because it can be used to compute any other function and thus gives a baseline to compare with when considering other functions. The class of *symmetric* functions includes all functions f such that for any input $\mathbf{x} \in \{0,1\}^n$ and permutation π on $\{1, 2, \ldots, n\}$

$$f(x_1, x_2, \ldots, x_n) = f\left(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)}\right).$$

In other words, the value of the function only depends on the arithmetic sum of the arguments, i.e., $\sum_{i=1}^{n} x_i$. Many functions which are useful in the context of sensor networks are symmetric, for example the *average*, *maximum*, *majority*, and *parity*.

A. Statement of Results

Under the communication models described above, and for any connection radius $r \in [1, \sqrt{2n}]$, we prove lower bounds on the latency and on the number of transmissions required for computing the identity function (Theorems III.1 and IV.1). We then describe a scheme which matches these bounds up to a constant factor (Theorems III.2 and IV.2). Next, we consider the class of symmetric functions. For a particular symmetric function (parity function), we provide lower bounds on the latency and on the number of transmissions for computing the function (Theorems III.3 and IV.3). We then present a scheme which can compute any symmetric function while matching the above bounds up to a constant factor (Theorems III.4 and IV.6). These results are summarized in Tables I and II. They illustrate the effect of the average node degree $\Theta(r^2)$ on the cost of computation under both communication models. By comparing the results for the identity function and symmetric functions, we can also quantify the gains in performance that can be achieved by using in-network aggregation for computation, rather than collecting all the data and performing the computation at the sink node. Finally, we extend our schemes for computing symmetric functions to more general network topologies (Theorem V.1) and obtain a lower bound on the number of transmissions required for arbitrary connected networks (Theorem V.2). A corollary of this result answers an open question originally posed by El Gamal in [6] regarding the computation of the parity function over ring and tree networks.

We point out that most of previous work ignored the issue of latency and it is only concerned with minimizing the number of transmissions required for computation. Our schemes are latency-optimal, in addition to being efficient in terms of the number of transmissions required. The works in [5], [11] consider the question of latency, but only for the case of $r = \Theta\left(\sqrt{\log n}\right)$.

B. Organization of the Paper

The rest of the paper is organized as follows. We formally describe the problem and present some preliminary remarks in Section II. Grid geometric networks with noiseless links are considered in Section III and their noisy counterparts are studied in Section IV. Extensions to general network topologies are presented in Section V. In Section VI we draw conclusions and mention some open problems.

II. Problem Formulation

A network $\mathcal{N}$ of n nodes is represented by an undirected graph. Nodes in the network represent communication devices and edges represent communication links. For each node i, let $N(i)$ denote its set of neighbors. Each node i is assigned an input bit $x_i \in \{0,1\}$. Let $\mathbf{x}$ denote the vector whose i^{th} component is x_i. We refer to $\mathbf{x}$ as the input to the network. The nodes communicate with each other so that a designated sink node v^* can compute a *target function* f of the input bits

$$f : \{0,1\}^n \to \mathcal{B}$$

where $\mathcal{B}$ denotes the co-domain of f. Time is divided into slots of unit duration. The communication models are as follows.

- Noiseless point-to-point model: If a node i transmits a bit on an edge (i, j) in a time slot, then node j receives the bit without any error in the same slot. All the edges in the network can be used simultaneously, i.e., there is no interference.
- Noisy broadcast model: If a node i transmits a bit b in time slot t, then each neighboring node in $N(i)$ receives an independent noisy copy of b in the same slot. More precisely, neighbor $j \in N(i)$ receives $b \oplus \eta_{i,j,t}$ where $\oplus$ denotes the modulo-2 sum. $\eta_{i,j,t}$ is a bernoulli random variable that takes value 1 with probability ϵ and 0 with probability $1-\epsilon$.

TABLE I
RESULTS FOR NOISELESS GRID GEOMETRIC NETWORKS

Function	No. of time slots	No. of transmissions
Identity	$\Theta\left(n/r^2\right)$	$\Theta\left(n^{3/2}/r\right)$
Symmetric	$\Theta\left(\sqrt{n}/r\right)$	$\Theta\left(n\right)$

TABLE II
RESULTS FOR NOISY GRID GEOMETRIC NETWORKS

Function	No. of time slots	No. of transmissions
Identity	$\max\{\Theta(n), \Theta\left(r^2 \log\log n\right)\}$	$\max\{\Theta\left(n^{3/2}/r\right), \Theta\left(n \log\log n\right)\}$
Symmetric	$\max\{\Theta\left(\sqrt{n}/r\right), \Theta\left(r^2 \log\log n\right)\}$	$\max\{\Theta\left(n \log n/r^2\right), \Theta\left(n \log\log n\right)\}$

The noise bits $\eta_{i,j,t}$ are independent over i, j, and t. A network in the noisy broadcast model with link error probability ϵ is called an ϵ-noise network. We restrict to the protocol model of operation, namely two nodes i and j can transmit in the same time slot only if they do not have any common neighbors, i.e., $N(i) \cap N(j) = \phi$. Thus, any node can receive at most one bit in a time slot. In the protocol model originally introduced in [2], communication is reliable. In our case, even if bits do not collide at the receiver because of the protocol model of operation, there is still a probability of error ϵ which models the inherent noise in the wireless communication medium.

A scheme for computing a target function f specifies the order in which nodes in the network transmit and the procedure for each node to decide what to transmit in its turn. A scheme is defined by the total number of time slots T of its execution, and for each slot $t \in \{1, 2, \ldots, T\}$, by a collection of S_t simultaneously transmitting nodes $\left\{v_1^t, v_2^t, \ldots v_{S_t}^t\right\}$ and the corresponding encoding functions $\left\{\phi_1^t, \phi_2^t, \ldots, \phi_{S_t}^t\right\}$. In any time slot $t \in \{1, 2, \ldots, T\}$, node v_j^t computes the function $\phi_j^t : \{0,1\} \times \{0,1\}^{\varphi_j^t} \to \{0,1\}$ of its input bit and the φ_j^t bits it received before time t and then transmits this value. In the noiseless point-to-point case, nodes in the list S_t are repeated for each distinct edge on which they transmit in a given slot. After the T rounds of communication, the sink node v^* computes an estimate $\hat{f}$ of the value of the function f. Note that the duration T of a scheme and the total number of transmissions $\sum_{i=1}^{T} S_t$ are constants for all inputs $\mathbf{x} \in \{0,1\}^n$.

Our definition of a computing scheme has a number of desirable properties. First, schemes are *oblivious* in the sense that in any time slot, the node which transmits is decided ahead of time and does not depend on a particular execution of the scheme. Without this property, the noise in the network may lead to multiple nodes transmitting at the same time, thereby causing collisions and violating the protocol model. Second, the definition rules out communication by *silence*: when it is a node's turn to transmit, it must send something.

We call a scheme a δ-error scheme for computing f if for any input $\mathbf{x} \in \{0,1\}^n$, $\mathsf{Pr}\left(\hat{f}(\mathbf{x}) \neq f(\mathbf{x})\right) \leq \delta$. For both the noiseless and noisy broadcast communication models, our objective is to characterize the minimum number of time slots T and the minimum number of transmissions required by any δ-error scheme for computing a target function f in a network $\mathcal{N}$. We first focus on grid geometric networks of connection radius r, denoted by $\mathcal{N}(n, r)$, and then extend our results to more general network topologies.

A. Preliminaries

We now present a few useful observations.

Remark II.1: For any connection radius $r < 1$, every node in the grid geometric network $\mathcal{N}(n, r)$ is isolated, and hence, computation is infeasible. On the other hand, for any $r \geq \sqrt{2n}$, the network $\mathcal{N}(n, r)$ is fully connected. Thus, the interesting regime is when the connection radius $r \in [1, \sqrt{2n}]$.

Remark II.2: For any connection radius $r \in [1, \sqrt{2n}]$, every node in the grid geometric network $\mathcal{N}(n, r)$ has $\Theta\left(r^2\right)$ neighbors.

Theorem II.3: (Gallager's Coding Theorem), [10, Page 3, Theorem 2], [15]: For any $\gamma > 0$ and any integer $m \geq 1$, there exists a code for sending an m-bit message over a binary symmetric channel using $O(m)$ transmissions such that the message is received correctly with probability at least $1 - e^{-\gamma m}$.

III. NOISELESS GRID GEOMETRIC NETWORKS

We begin by considering computation of the identity function. We have the following straightforward lower bound.

Theorem III.1: Let f be the identity function, let $\delta \in [0, 1/2)$, and let $r \in [1, \sqrt{2n}]$. Any δ-error scheme for computing f over $\mathcal{N}(n, r)$ requires at least $\Omega\left(n/r^2\right)$ time slots and $\Omega\left(n^{3/2}/r\right)$ transmissions.

Proof: To compute the identity function the sink node v^* should receive at least $(n-1)$ bits. Since v^* has $O\left(r^2\right)$ neighbors and can receive at most one bit on each edge in a time slot, it will require at least $\Omega\left(n/r^2\right)$ time slots to compute the identity function.

Let a cut be any set of edges separating at least one node from the sink v^*. It is easy to verify that there exists a collection of $\Omega\left(\sqrt{n}/r\right)$ disjoint cuts such that each cut separates $\Omega(n)$ nodes from the sink v^*, see Fig. 2 for an example. Thus, to ensure that v^* can compute the identity function, there should be at least $\Omega(n)$ transmissions across each cut. The lower bound on the total number of transmissions follows. ∎

We now present a simple scheme for computing the identity function which is order-optimal in both the latency and the number of transmissions.

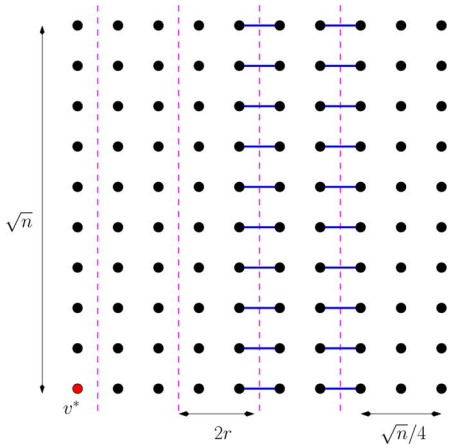

Fig. 2. Each dashed (magenta) line represents a cut of network $\mathcal{N}(n,r)$ which separates at least $n/4$ nodes from the sink v^*. Since the cuts are separated by a distance of at least $2r$, the edges in any two cuts, denoted by the solid (blue) lines, are disjoint.

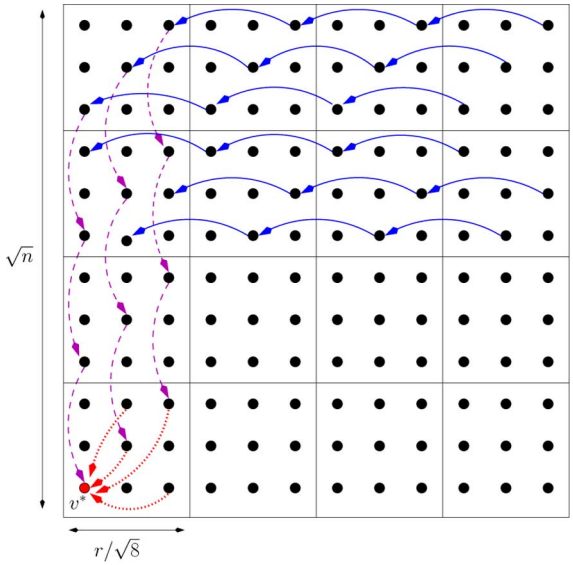

Fig. 3. Scheme for computing the identity function works in three phases: the solid (blue) lines depict the first horizontal aggregation phase, the dashed (magenta) lines denote the second vertical aggregation phase, and the dotted (red) lines represent the final phase of downloading data to the sink.

Theorem III.2: Let f be the identity function and let $r \in [1, \sqrt{2n}]$. There exists a zero-error scheme for computing f over $\mathcal{N}(n,r)$ which requires at most $O\left(n/r^2\right)$ time slots and $O\left(n^{3/2}/r\right)$ transmissions.

Proof: Let $c = r/\sqrt{8}$. Consider a partition of the network $\mathcal{N}(n,r)$ into cells of size $c \times c$, see Fig. 3. Note that each node is connected to all nodes in its own cell as well as in any neighboring cell. The scheme works in three phases, see Fig. 3. In the first phase, bits are horizontally aggregated towards the left-most column of cells along parallel linear chains. In the second phase, the bits in the left-most cells are vertically aggregated towards the nodes in the cell containing the sink node v^*. In the final phase, all the bits are collected at the sink node.

The first phase has bits aggregating along $O(\sqrt{n}r)$ parallel linear chains each of length $O(\sqrt{n}/r)$. By pipelining the transmissions, this phase requires $O(\sqrt{n}/r)$ time slots and a total of $O\left(\sqrt{n}r \times n/r^2\right)$ transmissions in the network. Since each node in the left-most column of cells has $O(\sqrt{n}/r)$ bits and there are $O\left(r^2\right)$ parallel chains each of length $O(\sqrt{n}/r)$, the second phase uses $O\left(r^2 \times \sqrt{n}/r \times n/r^2\right)$ transmissions and in $O(\sqrt{n}/r \times \sqrt{n}/r)$ time slots. In the final phase, each of the $O\left(r^2\right)$ nodes in the cell with v^* has $O\left(n/r^2\right)$ bits, and hence, it requires $O(n)$ transmissions and $O\left(n/r^2\right)$ slots to finish. Adding the costs, the scheme can compute the identity function with $O\left(n^{3/2}/r\right)$ transmissions and $O\left(n/r^2\right)$ time slots. ■

Now we consider the computation of symmetric functions. We have the following straightforward lower bound:

Theorem III.3: Let $\delta \in [0, 1/2)$ and let $r \in [1, \sqrt{2n}]$. There exists a symmetric target function f such that any δ-error scheme for computing f over $\mathcal{N}(n,r)$ requires at least $\Omega(\sqrt{n}/r)$ time slots and $(n-1)$ transmissions.

Proof: Let f be the parity function. To compute this function, each non-sink node in the network should transmit at least once. Hence, at least $(n-1)$ transmissions are required. Since the bit of the farthest node requires at least $\Omega(\sqrt{n}/r)$ time slots to reach v^*, we have the desired lower bound on the latency of any scheme.

Next, we present a matching upper bound. ■

Theorem III.4: Let f be any symmetric function and let $r \in [1, \sqrt{2n}]$. There exists a zero-error scheme for computing f over $\mathcal{N}(n,r)$ which requires at most $O(\sqrt{n}/r)$ time slots and $O(n)$ transmissions.

Proof: We present a scheme which can compute the arithmetic sum of the input bits over $\mathcal{N}(n,r)$ in at most $O(\sqrt{n}/r)$ time slots and $O(n)$ transmissions. This suffices to prove the result since f is symmetric, and thus, its value only depends on the arithmetic sum of the input bits.

Again, consider a partition of the noiseless network $\mathcal{N}(n,r)$ into cells of size $c \times c$ with $c = r/\sqrt{8}$. For each cell, pick one node arbitrarily and call it the "cell-center". For the cell containing v^*, choose v^* to be the cell-center. The scheme works in two phases, see Fig. 4.

First Phase: All the nodes in a cell transmit their input bits to the cell-center. This phase requires only one time-slot and n transmissions and at the end of the phase each cell-center knows the arithmetic sum of the input bits in its cell, which is an element of $\{0, 1, \ldots, \Theta\left(r^2\right)\}$.

Second Phase: In this phase, the bits at the cell-centers are aggregated so that v^* can compute the arithmetic sum of all the input bits in the network. There are two cases, depending on the connection radius r.

- $r \leq \sqrt{8 \log n}$: Since each cell-center is connected to the other cell-centers in its neighboring cells, this phase can be mapped to computing the arithmetic sum over the noiseless network $\mathcal{N}\left(\Theta\left(n/r^2\right), 1\right)$ where each node observes a message in $\{0, 1, \ldots, \Theta\left(r^2\right)\}$. See Fig. 4(a) for an illustration. In Appendix I we present a scheme to complete this phase using $O\left(n/r^2\right)$ transmissions and $O(\sqrt{n}/r)$ time slots.
- $r > \sqrt{8 \log n}$: The messages at cell-centers are aggregated towards v^* along a tree, see Fig. 4(b). The value at each cell-center can be viewed as a $\lceil \log n \rceil$-length binary

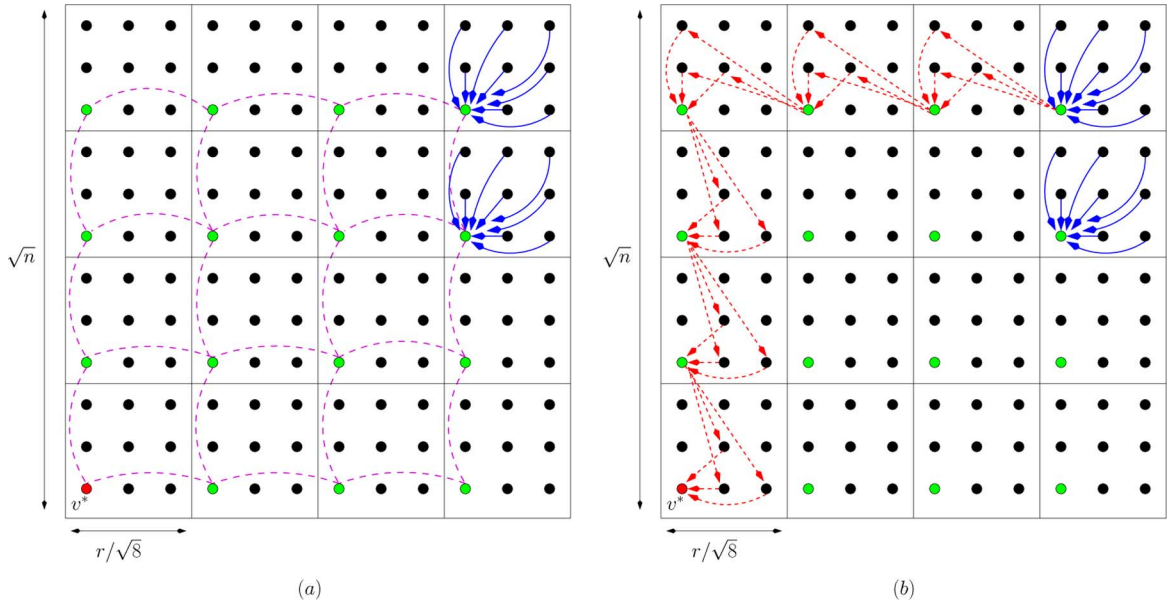

Fig. 4. Figures (a) and (b) represent the cases $r \leq \sqrt{8 \log n}$ and $r > \sqrt{8 \log n}$ respectively. The scheme for computing any symmetric function works in two phases: the solid (blue) lines indicate the first phase which is the same in both cases. The second phase differs in the two cases. It is represented by the dashed (magenta) lines in Fig. (a) and the dashed (red) lines in Fig. (b).

vector. To transmit its vector to the parent (cell-center) node in the tree, every leaf node (in parallel) transmits each bit of the vector to a distinct node in the parent cell. In the next time slot, each of these intermediate nodes relays its received bit to the corresponding cell-center. The parent cell-center can then reconstruct the message and aggregate it with its own value to form another $\lceil \log n \rceil$-length binary vector. Note that it requires two time slots and $O(\log n)$ transmissions by a cell-center to traverse one level of depth in the aggregation tree. This step is performed repeatedly (in succession) till the sink node v^* receives the sum of all the input bits in the network. Since the depth of the aggregation tree is $O(\sqrt{n}/r)$, the phase requires $O(\sqrt{n}/r)$ time slots. There are $O(\log n)$ transmissions in each cell of the network. Hence, the phase requires a total of $O\left(n/r^2 \times \log n\right) = O(n)$ transmissions.

Adding the costs of the two phases, we conclude that it is possible to compute any symmetric function using $O(n)$ transmissions and $O(\sqrt{n}/r)$ time slots. ∎

IV. Noisy Grid Geometric Networks

We start by considering the computation of the identity function. We have the following lower bound.

Theorem IV.1: Let f be the identity function. Let $\delta \in (0, 1/2)$, let $\epsilon \in (0, 1/2)$, and let $r \in [1, \sqrt{2n}]$. Any δ-error scheme for computing f over an ϵ-noise grid geometric network $\mathcal{N}(n, r)$ requires at least $\max\{n - 1, \Omega\left(r^2 \log\log n\right)\}$ time slots and $\max\{\Omega\left(n^{3/2}/r\right), \Omega(n \log\log n)\}$ transmissions.

Proof: The lower bound of $\Omega\left(n^{3/2}/r\right)$ transmissions follows from the same argument as in the proof of Theorem III.1. The other lower bound of $\Omega(n \log\log n)$ transmissions follows from [8, Corollary 2].

We now turn to the number of time slots required. For computing the identity function, the sink node v^* should receive at least $(n-1)$ bits. However, the sink can receive at most one bit in any slot, and hence, any scheme for computing the identity function requires at least $(n-1)$ time slots. For the remaining lower bound, consider a partition of the network $\mathcal{N}(n, r)$ into cells of size $c \times c$ with $c = r/\sqrt{8}$. Since the total number of transmissions in the network is at least $\Omega(n \log\log n)$ and there are $O\left(n/r^2\right)$ cells, there is at least one cell where the number of transmissions is at least $\Omega\left(r^2 \log\log n\right)$. Since all nodes in a cell are connected to each other, at most one of them can transmit in a slot. Thus, any scheme for computing the identity function requires at least $\Omega\left(r^2 \log\log n\right)$ time slots. ∎

Next, we present an efficient scheme for computing the identity function in noisy broadcast networks, which matches the above bounds.

Theorem IV.2: Let f be the identity function. Let $\delta \in (0, 1/2)$, let $\epsilon \in (0, 1/2)$, and let $r \in [1, \sqrt{2n}]$. There exists a δ-error scheme for computing f over an ϵ-noise grid geometric network $\mathcal{N}(n, r)$ which requires at most $\max\{O(n), O\left(r^2 \log\log n\right)\}$ time slots and $\max\{O\left(n^{3/2}/r\right), O(n \log\log n)\}$ transmissions.

Proof: Consider the usual partition of the network $\mathcal{N}(n, r)$ into cells of size $c \times c$ with $c = r/\sqrt{8}$. By the protocol model of operation any two nodes are allowed to transmit in the same time slot only if they do not have any common neighbors. In line with the protocol model of operation, cells are scheduled according to the scheme shown in Fig. 5. Thus, each cell is scheduled once every 7×7 time slots. Within a cell, at most one node can transmit in any given time slot and nodes take turns to transmit one after the other. For each cell, pick one node arbitrarily and call it the "cell-center". The scheme works in three phases, see Fig. 6.

First Phase: There are two different cases, depending on the connection radius r.

- $r \leq \sqrt{n}/\log n$: In this case, each node in its turn transmits its input bit to the corresponding cell-center using a code-

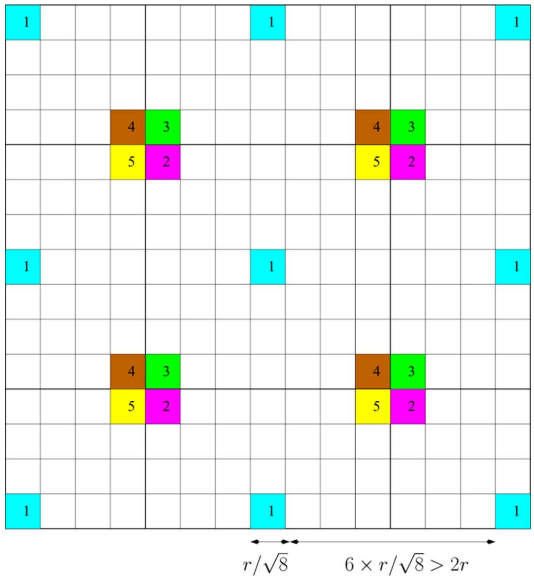

Fig. 5. Cells with the same number (and color) can be active in the same time slot and different numbers (colors) activate one after the other. Each cell is active once in 49 slots.

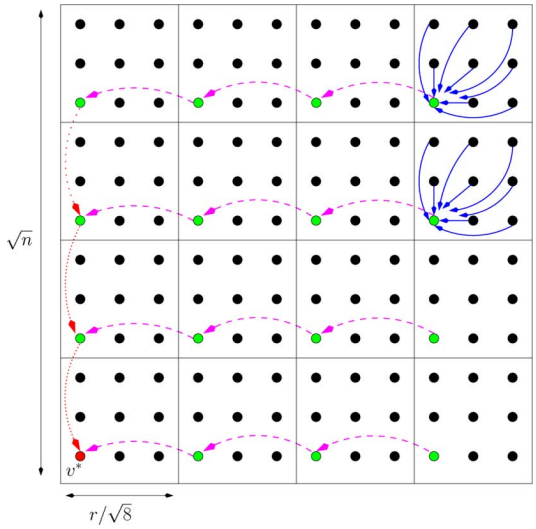

Fig. 6. Scheme for computing the identity function in a noisy network involves three phases: the solid (blue) lines indicate the first in-cell aggregation phase, the dashed (magenta) lines represent the second horizontal aggregation phase, and the dotted (red) lines represent the final vertical aggregation phase.

word of length $O(\log n)$ such that the cell-center decodes the message correctly with probability at least $1 - 1/n^2$. The existence of such a code is guaranteed by Theorem II.3. This phase requires at most $O\left(r^2 \log n\right)$ time slots and at most $O(n \log n)$ transmissions in the network. Since there are $O\left(n/r^2\right)$ cells in the network, the probability that the computation fails in at least one cell is bounded by $O(1/n)$.

- $r \geq \sqrt{n}/\log n$: In this case, each cell uses the more sophisticated scheme described in [8, Section 7] for recovering all the input messages from the cell at the cell-center. This scheme requires at most $O\left(r^2 \log\log n\right)$ time slots and a total of at most $O\left(n/r^2 \times r^2 \log\log n\right)$ transmissions in the network. At the end of the scheme, a cell-center has all the input messages from its cell with probability of error at most $O(\log n/n)$. Since there are at most $\log^2 n$ cells in the network for this case, the probability that the computation fails in at least one cell is bounded by $O(\log^3 n/n)$.

Thus, at the end of the first phase, all cell-centers in the network have the input bits of the nodes in their cells with probability at least $1 - O(\log^3 n/n)$.

Second Phase: In this phase, the messages collected at the cell-centers are aggregated horizontally towards the left-most cells, see Fig. 6. Note that there are $\sqrt{n}/r$ horizontal chains and each cell-center has $O\left(r^2\right)$ input messages. In each such chain, the rightmost cell-center maps its set of messages into a codeword of length $O\left(\sqrt{n}r\right)$ and transmits it to the next cell-center in the horizontal chain. The receiving cell-center decodes the incoming codeword, appends its own input messages, re-encodes it into a codeword of length $O\left(\sqrt{n}r\right)$, and then transmits it to the next cell-center, and so on. This phase requires at most $O\left(\sqrt{n}r \times \sqrt{n}/r\right)$ time slots and a total of at most $O\left(\sqrt{n}r \times n/r^2\right)$ transmissions in the network. From Theorem II.3, this step can be executed without error with probability at least $1 - O(1/n)$.

Third Phase: In the final phase, the messages at the cell-centers of the left-most column are aggregated vertically towards the sink node v^*, see Fig. 6. Each cell-center maps its set of input messages into a codeword of length $O\left(\sqrt{n}r\right)$ and transmits it to the next cell-center in the chain. The receiving cell-center decodes the incoming message, re-encodes it, and then transmits it to the next node, and so on. By pipelining the transmissions, this phase requires at most $O\left(\sqrt{n}r \times \sqrt{n}/r\right)$ time slots and at most $O\left(\sqrt{n}r \times n/r^2\right)$ transmissions in the network. This phase can also be executed without error with probability at least $1 - O(1/n)$.

It now follows that at the end of the three phases, the sink node v^* can compute the identity function with probability of error at most $O(\log^3 n/n)$. Thus, for n large enough, we have a δ-error scheme for computing any symmetric function in the network $\mathcal{N}(n, r)$. Adding the costs of the phases, the scheme requires at most $\max\{O(n), O\left(r^2 \log\log n\right)\}$ time slots and $\max\{O\left(n^{3/2}/r\right), O(n \log\log n)\}$ transmissions. ■

We now discuss the computation of symmetric functions in noisy broadcast networks. We begin with a lower bound on the latency and the number of transmissions required.

Theorem IV.3: Let $\delta \in (0, 1/2)$, let $\epsilon \in (0, 1/2)$, and let $r \in [1, n^{1/2-\beta}]$ for any $\beta > 0$. There exists a symmetric target function f such that any δ-error scheme for computing f over an ϵ-noise grid geometric network $\mathcal{N}(n, r)$ requires at least $\max\{\Omega\left(\sqrt{n}/r\right), \Omega\left(r^2 \log\log n\right)\}$ time slots and $\max\{\Omega\left(n \log n/r^2\right), \Omega(n \log\log n)\}$ transmissions.

We briefly describe the idea of the proof before delving into details. Let f be the parity function. First, we notice that [12, Theorem 1.1, page 1057] immediately implies that any δ-error scheme for computing f over $\mathcal{N}(n, r)$ requires at least $\Omega(n \log\log n)$ transmissions. So, we only need to establish that any such scheme also requires $\Omega\left(n \log n/r^2\right)$ transmissions.

Suppose there exists a δ-error scheme $\mathcal{P}$ for computing the parity function in an ϵ-noise grid geometric network $\mathcal{N}(n, r)$ which requires S transmissions. In Lemma IV.5, we translate the given scheme $\mathcal{P}$ into a new scheme $\mathcal{P}_1$ operating on a "noisy

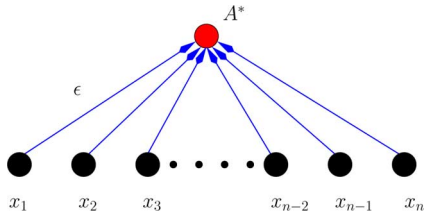

Fig. 7. n-noisy star network.

star" network (see Fig. 7) whose noise parameter depends on Sr^2/n, such that the probability of error for the new scheme $\mathcal{P}_1$ is also at most δ. In Lemma IV.4, we derive a lower bound on the probability of error of the scheme $\mathcal{P}_1$ in terms of the noise parameter of the noisy star network. Combining these results we obtain the desired lower bound on the number of transmissions S. We remark that while the proof of the lower bound in [12, Theorem 1.1, page 1057] operates a transformation to a problem over "noisy decision trees", here we need to transform the problem into one over a noisy star network. Hence, the two different transformations lead to different lower bounds on the number of transmissions required for computation.

An n-noisy star network consists of n input nodes and one auxiliary node A^*. Each of the n input nodes is connected directly to A^* via a noisy link, see Fig. 7. We have the following result for any scheme which computes the parity function in an n-noisy star network:

Lemma IV.4: Consider an n-noisy star network with noise parameter ϵ and let the input $\mathbf{x}$ be distributed uniformly over $\{0,1\}^n$. For any scheme $\mathcal{P}_1$ which computes the parity function (on n bits) in the network and in which each input node transmits its input bit only once, the probability of error is at least $\left(1-(1-2\epsilon)^n\right)/2$.

Proof: See Appendix II-A. ∎

We have the following lemma relating the original network $\mathcal{N}(n,r)$ and a noisy star network.

Lemma IV.5: Let $\alpha \in (0,1)$. If there is a δ-error scheme $\mathcal{P}$ for computing the parity function (on n input bits) in $\mathcal{N}(n,r)$ with S transmissions, then there is a δ-error scheme $\mathcal{P}_1$ for computing the parity function (on αn input bits) in an αn-noisy star network with noise parameter $\epsilon^{O(Sr^2/n)}$, with each input node transmitting its input bit only once.

Proof: See in Appendix II-B. ∎

We are now ready to complete the proof of Theorem IV.3.

Proof (of Theorem IV.3): Let $\alpha \in (0,1)$. If there is a δ-error scheme for computing the parity function in $\mathcal{N}(n,r)$ which requires S transmissions, then by combining Lemmas IV.5 and IV.4, the following inequalities must hold:

$$\begin{aligned}
&\delta \geq \frac{1-\left(1-2\epsilon^{O\left(Sr^2/n\right)}\right)^{\alpha n}}{2} \\
\implies &\left(1-2\epsilon^{O\left(Sr^2/n\right)}\right)^{\alpha n} \geq 1-2\delta \\
\implies &\left(2^{-2\epsilon^{O\left(Sr^2/n\right)}}\right)^{\alpha n} \overset{(a)}{\geq} 1-2\delta \\
\implies &S \geq \Omega\left(\frac{n\left(\log n - \log\log\left(1/(1-2\delta)\right)\right)}{r^2\log(1/\epsilon)}\right) \qquad (1)
\end{aligned}$$

where (a) follows since $2^{-x} \geq 1-x$ for every $x>0$. Thus, we have that any δ-error scheme for computing the parity function in an ϵ-noise network $\mathcal{N}(n,r)$ requires at least $\Omega\left(n\log n/r^2\right)$ transmissions.

We now consider the lower bound on the number of time slots. Since the message of the farthest node requires at least $\Omega\left(\sqrt{n}/r\right)$ time slots to reach v^*, we have the corresponding lower bound on the duration of any δ-error scheme. The lower bound of $\Omega\left(r^2\log\log n\right)$ time slots follows from the same argument as in the proof of Theorem IV.1. ∎

We now present an efficient scheme for computing any symmetric function in a noisy broadcast network which matches the above lower bounds.

Theorem IV.6: Let f be any symmetric function. Let $\delta \in (0,1/2)$, let $\epsilon \in (0,1/2)$, and let $r \in [1,\sqrt{2n}]$. There exists a δ-error scheme for computing f over an ϵ-noise grid geometric network $\mathcal{N}(n,r)$ which requires at most $\max\{O\left(\sqrt{n}/r\right), O\left(r^2\log\log n\right)\}$ time slots and $\max\{O\left(n\log n/r^2\right), O\left(n\log\log n\right)\}$ transmissions.

Proof: We present a scheme which can compute the arithmetic sum of the input bits over $\mathcal{N}(n,r)$. Note that this suffices to prove the result since f is symmetric, and thus, its value only depends on the arithmetic sum of the input bits.

Consider the usual partition of the network $\mathcal{N}(n,r)$ into cells of size $c\times c$ with $c=r/\sqrt{8}$. For each cell, we pick one node arbitrarily and call it the "cell-center". As before, cells are scheduled according to Fig. 5 to prevent interference between simultaneous transmissions. The scheme works in three phases.

First Phase: The objective of the first phase is to ensure that each cell-center computes the arithmetic sum of the input messages from the corresponding cell. Depending on the connection radius r, this is achieved using two different strategies.

- $r \leq \sqrt{\log n/\log\log n}$: In Appendix III, we describe a scheme which can compute the partial sums at all cell-centers with probability at least $1-O(1/n)$ and requires $O\left(n/r^2\times\log n\right)$ total transmissions and $O\left(\log n\right)$ time slots.
- $r > \sqrt{\log n/\log\log n}$: In this case, we first divide each cell into smaller sub-cells with $\Theta\left(\log n/\log\log n\right)$ nodes each, see Fig. 8. Each sub-cell has an arbitrarily chosen "head" node. In each sub-cell, we use the *Intracell scheme* from [10, Section III] to compute the sum of the input bits from the sub-cell at the corresponding head node. This requires $O\left(\log\log n\right)$ transmissions from each node in the sub-cell. Since there are $O\left(r^2\right)$ nodes in each cell and only one node in a cell can transmit in a time slot, this step requires $O\left(r^2\log\log n\right)$ time slots and a total of $O\left(n\log\log n\right)$ transmissions in the network. The probability that the computation fails in at least one sub-cell is bounded by $O(1/n)$.

Next, each head node encodes the sum of the input bits from its sub-cell into a codeword of length $O\left(\log n\right)$ and transmits it to the corresponding cell-center. This step requires a total of $O\left(n\log\log n\right)$ transmissions in the network and $O\left(r^2\log\log n\right)$ time slots and can be performed also with probability of error at most $O(1/n)$.

The received values are aggregated so that at the end of the first phase, all cell-centers know the sum of their input

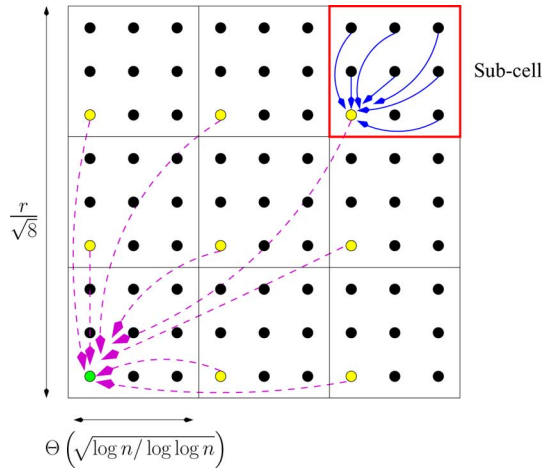

Fig. 8. Each cell in the network $\mathcal{N}(n,r)$ is divided into sub-cells. Every sub-cell has a "head", denoted by a yellow node. The sum of input messages from each sub-cell is obtained at its head node, depicted by the solid (blue) lines. These partial sums are then aggregated at the cell-center. The latter step is represented by the dashed (magenta) lines.

bits in their cell with probability at least $1 - O(1/n)$. The phase requires $O(n \log\log n)$ transmissions in the network and $O\left(r^2 \log\log n\right)$ time slots to complete.

Second Phase: In this phase, the partial sums stored at the cell-centers are aggregated along a tree (see for example, Fig. 6) so that the sink node v^* can compute the sum of all the input bits in the network. We have the following two cases, depending on the connection radius r.

- $r \geq (\sqrt{n}\log n)^{1/3}$: For this regime, our aggregation scheme is similar to the *Intercell scheme* in [10, Section III]. Each cell-center encodes its message into a codeword of length $\Theta(\log n)$. Each leaf node in the aggregation tree sends its codeword to the parent node which decodes the message, sums it with its own message and then re-encodes it into a codeword of length $\Theta(\log n)$. The process continues till the sink node v^* receives the sum of all the input bits in the network. From Theorem II.3, this phase carries a probability of error at most $O(1/n)$. It requires $O\left(n\log n/r^2\right)$ transmissions in the network and $O\left(\sqrt{n}/r \times \log n\right)$ time slots.
- $r \leq (\sqrt{n}\log n)^{1/3}$: In this regime, the above simple aggregation scheme does not match the lower bound for the latency in Theorem IV.3. A more sophisticated aggregation scheme is presented in [11, Section V], which uses ideas from [16] to efficiently simulate a scheme for noiseless networks in noisy networks. The phase carries a probability of error at most $O(1/n)$. It requires $O\left(n\log n/r^2\right)$ transmissions in the network and $O\left(\sqrt{n}/r\right)$ time slots.

Combining the two phases, the above scheme can compute any symmetric function with probability of error at most $O(1/n)$. Thus, for n large enough, we have a δ-error scheme for computing any symmetric function in the network $\mathcal{N}(n,r)$. It requires at most $\max\{O\left(\sqrt{n}/r\right), O\left(r^2\log\log n\right)\}$ time slots and $\max\{O\left(n\log n/r^2\right), O\left(n\log\log n\right)\}$ transmissions. ■

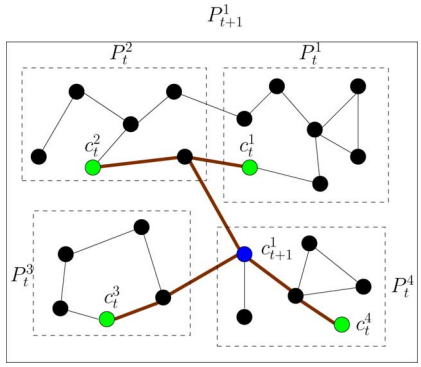

Fig. 9. Cell P_{t+1}^1 is composed of $\{P_t^k\}_{k=1}^4$ smaller cells from the previous level in the hierarchy. Each of the cell-centers c_t^k (denoted by the green nodes) holds the sum of the input bits in the corresponding cell P_t^k. These partial sums are aggregated along the minimum Steiner tree S_{t+1}^1 (denoted by the brown bold lines) so that the cell-center c_{t+1}^1 (denoted by the blue node) can compute the sum of all the input bits in P_{t+1}^1.

V. General Network Topologies

In the previous sections, we focused on grid geometric networks for their suitable regularity properties and for ease of exposition. The extension to random geometric networks in the continuum plane when $r = \Omega\left(\sqrt{\log n}\right)$ is immediate, and we focus here on extensions to more general topologies. First, we discuss extensions of our schemes for computing symmetric functions and then present a generalized lower bound on the number of transmissions required to compute symmetric functions in arbitrary connected networks.

A. Computing Symmetric Functions in Noiseless Networks

One of the key components for efficiently computing symmetric functions in noiseless networks in Theorem III.4 was the hierarchical scheme proposed for computing the arithmetic sum function in the grid geometric network $\mathcal{N}(n,1)$. The main idea behind the scheme was to consider successively coarser partitions of the network and at any given level aggregate the partial sum of the input messages in each individual cell of the partition using results from the finer partition in the previous level of the hierarchy. Using this idea we extend the hierarchical scheme to any connected noiseless network $\mathcal{N}$ and derive an upper bound on the number of transmissions required for the scheme. Let each node in the network start with an input bit and denote the set of nodes by $\mathcal{V}$. The scheme is defined by the following parameters:

- The number of levels h.
- For each level i, a partition $\Pi_i = \left\{P_i^1, P_i^2, \ldots, P_i^{s_i}\right\}$ of the set of nodes in the network $\mathcal{V}$ into s_i disjoint cells such that each $P_i^j = \cup_{k\in T_i^j} P_{i-1}^k$ where $T_i^j \subseteq \{1,2,\ldots,s_{i-1}\}$, i.e., each cell is composed of one or more cells from the next lower level in the hierarchy. See Fig. 9 for an illustration. Here, $\Pi_0 = \{\{i\} : i \in \mathcal{V}\}$ and $\Pi_h = \{\mathcal{V}\}$.
- For each cell P_i^j, a designated cell-center $c_i^j \in P_i^j$. Let c_h^1 be the designated sink node v^*.
- For each cell P_i^j, let S_i^j denote a Steiner tree with the minimum number of edges which connects the corresponding cell-center with all the cell-centers of its component cells

P_{i-1}^k, i.e., the set of nodes $\cup_{k\in T_i^j} c_{i-1}^k \cup c_i^j$. Let l_i^j denote the number of edges in S_i^j.

Using the above definitions, the hierarchical scheme from Theorem III.4 can now be easily extended to general network topologies. We start with the first level in the hierarchy and then proceed recursively. At any given level, we compute the partial sums of the input messages in each individual cell of the partition at the corresponding cell-center by aggregating the results from the previous level along the minimum Steiner tree. It is easy to verify that after the final level in the scheme, the sink node v^* possesses the arithmetic sum of all the input messages in the network $\mathcal{N}$. The total number of transmissions made by the scheme is at most

$$\sum_{t=1}^{h}\sum_{j=1}^{s_{t+1}} l_t^j \cdot \log\left(\left|P_t^j\right|\right).$$

Thus, we have a scheme for computing the arithmetic sum function in any arbitrary connected network. In the proof of Theorem III.4, the above bound is evaluated for the grid geometric network $\mathcal{N}(n,1)$ with $h=\log\sqrt{n}$, $s_t = n/2^{2t}$, $l_t^j \le 4\cdot 2^{t-1}$, $\left|P_t^j\right| = 2^{2t}$, and is shown to be $O(n)$.

B. Computing Symmetric Functions in Noisy Networks

We generalize the scheme in Theorem IV.6 for computing symmetric functions in a noisy grid geometric network $\mathcal{N}(n,r)$ to a more general class of network topologies and derive a corresponding upper bound on the number of transmissions required. The original scheme consists of two phases: an intracell phase where the network is partitioned into smaller cells, each of which is a clique, and partial sums are computed in each individual cell; and an intercell phase where the partial sums in cells are aggregated to compute the arithmetic sum of all input messages at the sink node. We extend the above idea to more general topologies. First, for any $z \ge 1$, consider the following definition:

Clique-cover property $\mathcal{C}(z)$: a network $\mathcal{N}$ of n nodes is said to satisfy the clique-cover property $\mathcal{C}(z)$ if the set of nodes $\mathcal{V}$ is covered by at most $\lfloor n/z \rfloor$ cliques, each of size at most $\log n/\log\log n$.

For example, a grid geometric network $\mathcal{N}(n,r)$ with $r = O(\sqrt{\log n/\log\log n})$ satisfies $C(z)$ for $z = O(r^2)$. On the other hand, a tree network satisfies $C(z)$ only for $z \le 2$. Note that any connected network satisfies property $C(1)$. By regarding each disjoint clique in the network as a cell, we can easily extend the analysis in Theorem IV.6 to get the following result, whose proof is omitted.

Theorem V.1: Let $\delta \in \left(0,\frac{1}{2}\right)$, $\epsilon \in \left(0,\frac{1}{2}\right)$, and $\mathcal{N}$ be any connected network of n nodes with $n \ge 2/\delta$. For $z \ge 1$, if $\mathcal{N}$ satisfies $\mathcal{C}(z)$, then there exists a δ-error scheme for computing any symmetric function over $\mathcal{N}$ which requires at most $O(n\log n/z)$ transmissions.

C. A Generalized Lower Bound for Symmetric Functions

The proof techniques that we use to obtain lower bounds are also applicable to more general network topologies. Recall that $N(i)$ denotes the set of neighbors for any node i. For any network, define the average degree as

$$d(n) = \frac{\sum_{i\in\mathcal{V}} |N(i)|}{n}.$$

A slight modification to the proof of Theorem IV.3 leads to the following result:

Theorem V.2: Let $\delta \in \left(0,\frac{1}{2}\right)$ and let $\epsilon \in \left(0,\frac{1}{2}\right)$. There exists a symmetric target function f such that any δ-error scheme for computing f over any connected network of n nodes with average degree $d(n)$, requires at least $\Omega\left(\frac{n\log n}{d(n)}\right)$ transmissions.

Proof: Let f be the parity function. The only difficulty in adapting the proof of Theorem IV.3 arises from the node degree not being necessarily the same for all the nodes. We circumvent this problem as follows: in addition to decomposing the network into the set of source nodes $\mathcal{I}$ and auxiliary nodes $\mathcal{A}$, such that $|\mathcal{I}| = \alpha n$ for $\alpha \in (0,1)$, as in the proof of Lemma IV.5 (see Appendix II-B); we also let every source node with degree more than $\frac{2d(n)}{\alpha}$ be an auxiliary node. There can be at most $\frac{\alpha n}{2}$ of such nodes in the network since the average degree is $d(n)$. Thus, we obtain an $\left(\frac{\alpha n}{2}, (1-\frac{\alpha}{2})n\right)$ decomposition of the network such that each source node has degree at most $\frac{2d(n)}{\alpha}$. The rest of the proof then follows in the same way. ∎

As an application of the above result, we have the following lower bound for ring or tree networks.

Corollary V.3: Let f be the parity function, let $\delta \in \left(0,\frac{1}{2}\right)$, and let $\epsilon \in \left(0,\frac{1}{2}\right)$. Any δ-error scheme for computing f over any ring or tree network of n nodes requires at least $\Omega(n\log n)$ transmissions.

The above result answers an open question, posed originally by El Gamal [6].

VI. Conclusion

We conclude with some observations and directions for future work.

A. Target Functions

We considered all symmetric functions as a single class and presented a worst-case characterization (up to a constant) of the number of transmissions and time slots required for computing this class of functions. A natural question to ask is whether it is possible to obtain better performance if one restricts to a particular sub-class of symmetric functions. For example, two subclasses of symmetric functions are considered in [3]: *type-sensitive* and *type-threshold*. Since the parity function is a type-sensitive function, the characterization for noiseless networks in Theorems III.3 and III.4, as well as noisy broadcast networks in Theorems IV.3 and IV.6 also holds for the restricted sub-class of type-sensitive functions. A similar general characterization is not possible for type-threshold functions since the trivial function ($f(\mathbf{x}) = 0$ for all $\mathbf{x}$) is also in this class and it requires no transmissions and time slots to compute. The following result, whose proof follows similar lines as the results in previous

sections and is omitted, characterizes the number of transmissions and the number of time slots required for computing the maximum function, which is an example type-threshold function. This can be compared with the corresponding results for the whole class of symmetric functions in Theorems IV.3 and IV.6.

Theorem VI.1: Let f be the maximum function. Let $\delta \in (0,1/2)$, $\epsilon \in (0,1/2)$, and $r \in [1,\sqrt{2n}]$. Any δ-error scheme for computing f over an ϵ-noise network $\mathcal{N}(n,r)$ requires at least $\max\{\Omega(\sqrt{n}/r),\Omega(r^2)\}$ time slots and $\max\{\Omega(n\log n/r^2),\Omega(n)\}$ transmissions. Further, there exists a δ-error scheme for computing f which requires at most $\max\{O(\sqrt{n}/r),O(r^2)\}$ time slots and $\max\{O(n\log n/r^2),O(n)\}$ transmissions.

B. On the Role of ϵ and δ

Throughout the paper, the channel error parameter ϵ and the threshold δ on the probability of error are given constants. It is also interesting to study how the cost of computation depends on these parameters. The careful reader might have noticed that our proposed schemes also work when only an upper bound on the channel error parameter ϵ is considered, and always achieve a probability of error δ that is either zero or tends to zero as $n \to \infty$. It is also clear that the cost of computation should decrease with smaller values of ϵ and increase with smaller values of δ. Indeed, from (1) in the proof of Theorem IV.3, we see that the lower bound on the number of transmissions required for computing the parity function depends on ϵ as $1/(-\log\epsilon)$. On the other hand, from the proof of Theorem IV.6 the upper bound on the number of transmissions required to compute any symmetric function depends on ϵ as $1/(-\log(\epsilon(1-\epsilon)))$. The two expressions are close for small values of ϵ.

C. Network Models

We assumed that each node in the network has a single bit value. Our results can be immediately adapted to obtain upper bounds on the latency and the number of transmissions required for the more general scenario where each node i observes a block of input messages $x_i^1, x_i^2, \ldots, x_i^k$ with each $x_i^j \in \{0,1,\ldots,q\}$, $q \geq 2$. However, finding matching lower bounds seems to be more challenging.

Appendix I
Computing the Arithmetic Sum Over $\mathcal{N}(n,1)$

Consider a noiseless network $\mathcal{N}(n,1)$ where each node i has an input message $x_i \in \{0,1,\ldots,q-1\}$. We present a scheme which can compute the arithmetic sum of the input messages over the network in $O(\sqrt{n}+\log q \cdot \log n)$ time slots and using $O(n+\log q)$ transmissions. We briefly present the main idea of the scheme before delving into details. Our scheme divides the network into small cells and computes the sum of the input messages in each individual cell at designated cell-centers. We then proceed recursively and in each iteration we double the size of the cells into which the network is partitioned and compute the partial sums by aggregating the computed values from the previous round. This process finally yields the arithmetic sum of all the input messages in the network.

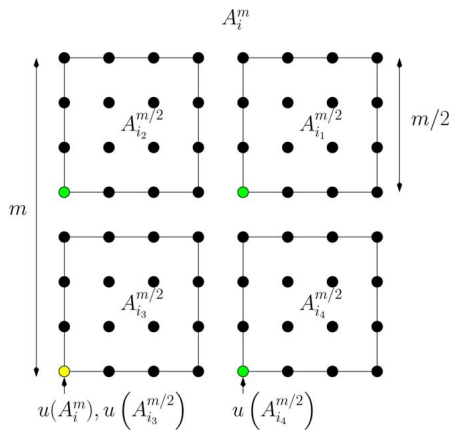

Fig. 10. A_i^m is a square cell of size $m \times m$. This figure illustrates some notation with regards to A_i^m.

Before we describe the scheme, we define some notation. Consider an $m \times m$ square cell in the network, see Fig. 10. Denote this cell by A_i^m and the node in the lower-left corner of A_i^m by $u(A_i^m)$. For any m which is a power of 2, $m \geq 2$, A_i^m can be divided into 4 smaller cells, each of size $m/2 \times m/2$. Denote these cells by $\left\{A_{i_j}^{m/2}\right\}_{j=1}^{4}$.

Without loss of generality, let n be a power of 4. The scheme has the following steps:

1) Let $k = 0$.
2) Consider the partition of the network into cells $\left\{A_i^{2^{k+1}}\right\}_{i=1}^{\frac{n}{2^{2(k+1)}}}$ each of size $2^{k+1} \times 2^{k+1}$, see Fig. 11. Note that each cell $A_i^{2^{k+1}}$ consists of exactly four cells $\left\{A_{i_1}^{2^k},\ldots,A_{i_4}^{2^k}\right\}$, see Fig. 12. Each corner node $u\left(A_{i_j}^{2^k}\right)$, $j = 1, 2, 3, 4$ possesses the sum of the input messages corresponding to the nodes in the cell $A_{i_j}^{2^k}$.
   The partial sums stored at $u\left(A_{i_j}^{2^k}\right)$, $j = 1, 2, 3, 4$ are aggregated at the node $u\left(A_i^{2^{k+1}}\right)$, along the tree shown in Fig. 12. Each node in the tree makes at most $\log\left(2^{2(k+1)}q\right)$ transmissions.
   At the end of this step, each corner node $u\left(A_i^{2^{k+1}}\right)$ has the sum of the input messages corresponding to the nodes in the cell $A_i^{2^{k+1}}$. By pipelining the transmissions along the tree, this step takes at most

$$2\left(2^k + \log\left(2^{2(k+1)}q\right)\right) \text{ time slots.}$$

   The total number of transmissions in the network for this step is at most

$$\frac{n}{2^{2(k+1)}} \cdot 4 \cdot 2^k \cdot \log\left(2^{2(k+1)}q\right) = \frac{4n}{2^{k+1}}(k+1+\log q).$$

3) Let $k \leftarrow k+1$. If $2^{k+1} \leq \sqrt{n}$, return to step 2, else terminate.

Note that at the end of the process, the sink node v^* can compute the sum of the input messages for any input $\mathbf{x} \in \{0,1,\ldots,q-1\}^n$. The total number of steps in the

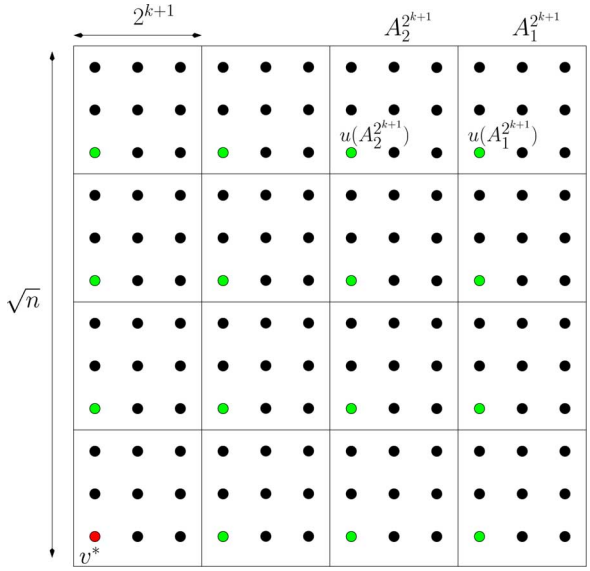

Fig. 11. This figure illustrates the partition of the network $\mathcal{N}(n,1)$ into smaller cells $\left\{A_i^{2^{k+1}}\right\}_{i=1}^{\frac{n}{2^{2(k+1)}}}$, each of size $2^{k+1} \times 2^{k+1}$.

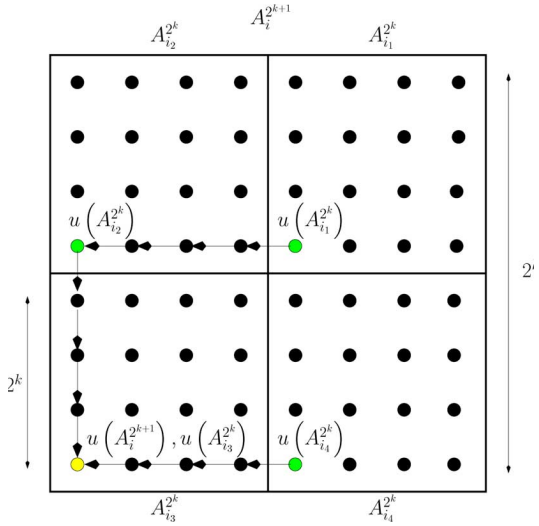

Fig. 12. Step 2 of the scheme for computing the sum of input messages. The network is divided into smaller cells, each of size $2^{k+1} \times 2^{k+1}$. For any such cell $A_i^{2^{k+1}}$, $j \in \{1,2,3,4\}$, each corner node $u\left(A_{i_j}^{2^k}\right)$ has the sum of the input messages corresponding to the nodes in the cell $A_{i_j}^{2^k}$. Then the sum of the input messages corresponding to the cell $A_i^{2^{k+1}}$ is aggregated at $u\left(A_i^{2^{k+1}}\right)$, along the tree shown in the figure.

scheme is $\log\sqrt{n}$. The number of time slots that the scheme takes is at most

$$\sum_{k=0}^{\log\sqrt{n}-1} 2\left(2^k + \log\left(2^{2(k+1)}q\right)\right) \leq O\left(\log q \cdot \log n + \sqrt{n}\right).$$

The total number of transmissions made by the scheme is at most

$$\sum_{k=0}^{\log\sqrt{n}-1} \frac{4n(k+1+\log q)}{2^{k+1}} \leq O\left(n + \log q\right).$$

APPENDIX II
COMPLETION OF THE PROOF OF THEOREM IV.3

A) Proof of Lemma IV.4: For every $i \in \{1,2,\ldots,n\}$, let y_i be the noisy copy of x_i that the auxiliary node A^* receives. Denote the received vector by $\mathbf{y}$. The objective of A^* is to compute the parity of the input bits $x_1, x_2, \ldots, x_n$. Thus, the target function f is defined as

$$f(\mathbf{x}) = x_1 \oplus x_2 \oplus \cdots \oplus x_n.$$

Since the input $\mathbf{x}$ is uniformly distributed over $\{0,1\}^n$, we have $\mathsf{Pr}(f(\mathbf{x})=0) = \mathsf{Pr}(f(\mathbf{x})=1) = 1/2$. In the following, we first show that Maximum Likelihood estimation is equivalent to using the parity of the received bits $y_1, y_2, \ldots, y_n$ i.e., $f(\mathbf{y})$ as an estimate for $f(\mathbf{x})$, and then compute the corresponding probability of error. From the definition of Maximum Likelihood estimation, we have

$$\hat{f}(\mathbf{x}) = \begin{cases} 1, & \text{if } \mathsf{Pr}(\mathbf{y}|f(\mathbf{x})=1) > \mathsf{Pr}(\mathbf{y}|f(\mathbf{x})=0) \\ 0, & \text{otherwise.} \end{cases}$$

Next

$$\begin{aligned} &\mathsf{Pr}(\mathbf{y}|f(\mathbf{x})=1) \\ &= \sum_{\substack{\mathbf{x}\in\{0,1\}^n \\ \text{s.t} f(\mathbf{x})=1}} \mathsf{Pr}(\mathbf{x}|f(\mathbf{x})=1)\cdot\mathsf{Pr}(\mathbf{y}|\mathbf{x}) \\ &\overset{(a)}{=} \kappa \sum_{\substack{\mathbf{x}\in\{0,1\}^n \\ \text{s.t} f(\mathbf{x})=1}} \mathsf{Pr}(y_1|x_1)\,\mathsf{Pr}(y_2|x_2)\ldots\mathsf{Pr}(y_n|x_n) \end{aligned}$$

where $\kappa = 2^{-(n-1)}$ and (a) follows since $\mathbf{x}$ is uniformly distributed over $\{0,1\}^n$ and from the independence of the channels between the sources and the auxiliary node. Similarly

$$\mathsf{Pr}(\mathbf{y}|f(\mathbf{x})=0) = \kappa \sum_{\substack{\mathbf{x}\in\{0,1\}^n \\ \text{s.t} f(\mathbf{x})=0}} \mathsf{Pr}(y_1|x_1)\,\mathsf{Pr}(y_2|x_2)\ldots\mathsf{Pr}(y_n|x_n).$$

Putting things together, we have

$$\begin{aligned} &\mathsf{Pr}(\mathbf{y}|f(\mathbf{x})=0) - \mathsf{Pr}(\mathbf{y}|f(\mathbf{x})=1) \\ &= \kappa\left(\sum_{\substack{\mathbf{x}\in\{0,1\}^n \\ \text{s.t} f(\mathbf{x})=0}} \prod_{i=1}^{n} \mathsf{Pr}(y_i|x_i) - \sum_{\substack{\mathbf{x}\in\{0,1\}^n \\ \text{s.t} f(\mathbf{x})=1}} \prod_{i=1}^{n} \mathsf{Pr}(y_i|x_i)\right) \\ &\overset{(a)}{=} \kappa \prod_{i=1}^{n} \left(\mathsf{Pr}(y_i|x_i=0) - \mathsf{Pr}(y_i|x_i=1)\right) \\ &= \kappa\,(-1)^{n_1(\mathbf{y})}\,(1-2\epsilon)^n \end{aligned} \tag{2}$$

where $n_1(\mathbf{y})$ is the number of components in $\mathbf{y}$ with value 1. The above equality (a) can be verified by noting that the product in (a) produces a sum of 2^n monomials and that the sign of each

monomial is positive if the number of terms of the monomial conditioned on $x_i = 1$ is even, and negative otherwise. From (2), we now have

$$\mathsf{Pr}\left(\mathbf{y}|f(\mathbf{x})=0\right)-\mathsf{Pr}\left(\mathbf{y}|f(\mathbf{x})=1\right)\begin{cases}>0, & \text{if } f(\mathbf{y})=0\\ <0, & \text{if } f(\mathbf{y})=1.\end{cases}$$

Thus, we have shown that Maximum Likelihood estimation is equivalent to using $f(\mathbf{y})$ as an estimate for $f(\mathbf{x})$. The corresponding probability of error is given by

$$\begin{aligned}\mathsf{Pr}_{ML}\left(\text{Error}\right)&=\mathsf{Pr}\left(f(\mathbf{y})\neq f(\mathbf{x})\right)\\&=\sum_{j=1}^{\lceil n/2\rceil}\mathsf{Pr}\left(\sum_{i=1}^{n}x_i\oplus y_i=2j-1\right)\\&=\sum_{j=1}^{\lceil n/2\rceil}\binom{n}{2j-1}\epsilon^{2j-1}\left(1-\epsilon\right)^{n-2j+1}\\&=\frac{1-(1-2\epsilon)^n}{2}.\end{aligned}$$

Hence, for any scheme $\mathcal{P}_1$ which computes the parity function f in an n-noisy star network, the probability of error is at least $\left(1-(1-2\epsilon)^n\right)/2$.

B) Proof of Lemma IV.5: We borrow some notation from [8] and [12]. Consider the nodes in a network and mark a subset $\mathcal{I}$ of them as input nodes and the rest $\mathcal{A}$ as auxiliary nodes. Such a decomposition of the network is called an $(|\mathcal{I}|,|\mathcal{A}|)$-decomposition. An input value to this network is an element of $\{0,1\}^{|\mathcal{I}|}$. Consider a scheme $\mathcal{P}$ on such a network which computes a function f of the input. The scheme is said to be m-bounded with respect to an $(|\mathcal{I}|,|\mathcal{A}|)$-decomposition if each node in $\mathcal{I}$ makes at most m transmissions. Recall from Section II that for any scheme in our model, the number of transmissions that any node makes is fixed a priori and does not depend on a particular execution of the scheme. Following [8] and [12] we define the *semi-noisy network*, in which whenever it is the turn of an input node to transmit, it sends its input bit whose independent ϵ-noisy copies are received by its neighbors, while the transmission made by auxiliary nodes are not subject to any noise.

The proof now proceeds by combining three lemmas. Suppose there exists a δ-error scheme $\mathcal{P}$ for computing the parity function in an ϵ-noise network $\mathcal{N}(n,r)$ which requires S transmissions. We first show in Lemma II.1 that this implies the existence of a suitable decomposition of the network and a δ-error, $O(S/n)$-bounded scheme $\mathcal{P}_d$ for computing the parity function in this decomposed network. Lemma II.2 translates the scheme $\mathcal{P}_d$ into a scheme $\mathcal{P}_s$ for computing in a semi-noisy network and Lemma II.3 translates $\mathcal{P}_s$ into a scheme $\mathcal{P}_1$ for computing in a noisy star network, while ensuring that the probability of error does not increase at any intermediate step. The proof is completed using the fact that the probability of error for original scheme $\mathcal{P}$ is at most δ.

Let $\alpha\in(0,1)$. We have the following lemma.

Lemma II.1: If there is a δ-error scheme $\mathcal{P}$ for computing the parity function (on n input bits) in $\mathcal{N}(n,r)$ with S transmissions, then there is an $(\alpha n,(1-\alpha)n)$-decomposition of $\mathcal{N}(n,r)$ and a δ-error, $O(S/n)$-bounded scheme $\mathcal{P}_d$ for computing the parity function (on αn bits) in this decomposed network.

Proof: If all nodes in the network make $O(S/n)$ transmissions, then the lemma follows trivially. Otherwise, we decompose the network into the set of input nodes $\mathcal{I}$ and auxiliary nodes $\mathcal{A}$ as follows. Consider the set of nodes which make more than $\frac{S}{n(1-\alpha)}$ transmissions each during the execution of the scheme $\mathcal{P}$. Since $\mathcal{P}$ requires S transmissions, there can be at most $(1-\alpha)n$ of such nodes. We let these nodes be auxiliary nodes and let their input be 0. Thus, we have an $(\alpha n,(1-\alpha)n)$-decomposition of the network $\mathcal{N}(n,r)$. The scheme now reduces to computing the parity (on αn bits) over this decomposed network. By construction, each input node makes at most $\frac{S}{n(1-\alpha)}$ transmissions, and hence, the scheme is $O(S/n)$-bounded. ∎

The following lemma is stated without proof, as it follows immediately from [8, Section 6, page 1833], or [12, Lemma 5.1, page 1064].

Lemma II.2: (FROM NOISY TO SEMI-NOISY) For any function $f:\{0,1\}^{\alpha n}\to\{0,1\}$ and any δ-error, $O(S/n)$-bounded scheme $\mathcal{P}_d$ for computing f in an $(\alpha n,(1-\alpha)n)$-decomposition of $\mathcal{N}(n,r)$, there exists an $(\alpha n,n)$-decomposed semi-noisy network of $(1+\alpha)n$ nodes such that each input node has at most $O\left(r^2\right)$ neighbors and a δ-error, $O(S/n)$-bounded scheme $\mathcal{P}_s$ for computing f in the semi-noisy network.

We now present the final lemma needed to complete the proof.

Lemma II.3: (FROM SEMI-NOISY TO NOISY STAR) For any function $f:\{0,1\}^{\alpha n}\to\{0,1\}$ and any δ-error, $O(S/n)$-bounded scheme $\mathcal{P}_s$ for computing f in an $(\alpha n,n)$-decomposed semi-noisy network where each input node has at most $O\left(r^2\right)$ neighbors, there exists a δ-error scheme $\mathcal{P}_1$ for computing f in an αn-noisy star network with noise parameter $\epsilon^{O(Sr^2/n)}$, with each input node transmitting its input bit only once.

Proof: In a semi-noisy network, when it is the turn of an input node to transmit during the execution of $\mathcal{P}_s$, it transmits its input bit. Since the bits sent by the input nodes do not depend on bits that these nodes receive during the execution of the scheme, we can assume that the input nodes make their transmissions at the beginning of the scheme an appropriate number of times, and after that only the auxiliary nodes communicate without any noise. Further, since any input node in the $(\alpha n,n)$-decomposed network has at most $O\left(r^2\right)$ neighbors, at most $O\left(r^2\right)$ auxiliary nodes receive independent ϵ-noisy copies of each such input bit. Since $\mathcal{P}_s$ is an $O(S/n)$-bounded scheme, each input node makes at most $O(S/n)$ transmissions, and hence, the auxiliary nodes receive a total of at most $O\left(Sr^2/n\right)$ independent ϵ-noisy copies of each input bit.

Next, we use the scheme $\mathcal{P}_s$ to construct a scheme $\mathcal{P}_1$ for computing f in an αn-noisy star network of noise parameter $\epsilon^{O(Sr^2/n)}$ with each input node transmitting its input bit only

once. Lemma II.4 shows that upon receiving an $\epsilon^{O(Sr^2/n)}$-noisy copy for every input bit, the auxiliary node A^* in the noisy star network can generate $O\left(Sr^2/n\right)$ independent ϵ-noisy copies for each input bit. Then onwards, the auxiliary node A^* can simulate the scheme $\mathcal{P}_s$. This is true since for $\mathcal{P}_s$ only the auxiliary nodes operate after the initial transmissions by the input nodes, and their transmissions are not subject to any noise. ∎

Lemma II.4: [8, Lemma 36, page 1834] Let $t \in \mathbb{N}$, $\epsilon \in (0, 1/2)$, and $\gamma = \epsilon^t$. There is a randomized algorithm that takes as input a single bit b and outputs a sequence of t bits such that if the input is a γ-noisy copy of 0 (respectively of 1), then the output is a sequence of independent ϵ-noisy copies of 0 (respectively of 1).

APPENDIX III
SCHEME FOR COMPUTING PARTIAL SUMS AT CELL-CENTERS

We describe an adaptation of the scheme in [10, Section III], which requires at most $O\left(\frac{n\log n}{r^2}\right)$ transmissions and $O(\log n)$ time slots. The scheme in [10, Section III], is described for $r > \sqrt{\log n}$ and while the same ideas work for $r \leq \sqrt{\log n/\log\log n}$, the parameters need to be chosen carefully so that the scheme can compute efficiently in the new regime.

Recall that the network is partitioned into cells of size $c \times c$ where $c = r/\sqrt{8}$. Consider any cell A_1 in the network and denote its cell-center by v_1. The scheme has the following steps:

1) Each node in A_1 takes turn to transmit its input bit x_i, $\frac{8}{\lambda}\left(\frac{\log n}{c^2}\right)$ times, where $\lambda = -\ln(4\epsilon(1-\epsilon))$. Thus, every node in A_1 receives $\frac{8}{\lambda}\left(\frac{\log n}{c^2}\right)$ independent noisy copies of the entire input. This step requires $n \cdot \frac{8}{\lambda}\left(\frac{\log n}{c^2}\right)$ transmissions and $\frac{8\log n}{\lambda}$ time slots.
2) Each node in A_1 forms an estimate for the input bits of the other nodes in A_1 by taking the *majority* of the noisy bits that it received from each of them. It is easy to verify that the probability that a node has a wrong estimate for any given input bit to be at most $c^2 \cdot e^{-\frac{4\log n}{c^2}}$, see for example Gallager's book [15, page 125]. Each node then computes the arithmetic sum of all the decoded bits and thus has an estimate of the sum of all the input bits in the cell A_1.
3) Each node in A_1 transmits its estimate to the cell-center v_1 using a codeword of length $\frac{k\log n}{c^2}$ such that k is a constant and the cell-center decodes the message with probability of error at most $e^{-\frac{4\log n}{c^2}}$. The existence of such a code is guaranteed by Theorem II.3 and from the fact that the size of the estimate in bits $\log(c^2+1) \leq \log n/c^2$, since $c^2 = \frac{r^2}{8} \leq \frac{\log n}{8\log\log n}$. At the end of this step, v_1 has c^2 independent estimates for the sum of the input bits corresponding to the nodes in A_1. The total number of transmissions for this step is at most $n \cdot \frac{k\log n}{c^2}$ and it requires at most $k\log n$ time slots.
4) The cell-center v_1 takes the *mode* of these c^2 values to make the final estimate for the sum of the input bits in A_1.

We can now bound the probability of error for the scheme as follows:

$$\begin{aligned}
&\mathsf{Pr}(\text{Error}) \\
&\leq \left(4\left(c^2 e^{-\frac{4\log n}{c^2}} + e^{-\frac{4\log n}{c^2}}\right)\left(1 - (c^2 e^{-\frac{4\log n}{c^2}} + e^{-\frac{4\log n}{c^2}})\right)\right)^{c^2} \\
&\overset{(a)}{\leq} \left(4 \cdot 9c^2 e^{-\frac{4\log n}{c^2}}\right)^{c^2} \\
&\overset{(b)}{\leq} \frac{1}{n^2}, \text{ for } n \text{ large enough}
\end{aligned}$$

where (a) follows since $8c^2 = r^2 \geq 1$; and (b) follows since $c^2\log c^2 \leq \log n$. Thus, every cell-center v_i can compute the sum of the input bits corresponding to the nodes in A_i with probability of error at most $\frac{1}{n^2}$. The total probability of error is then at most $\frac{n}{c^2} \cdot \frac{1}{n^2} \leq \frac{1}{n}$. The total number of transmissions in the network for this scheme is at most $\left(\frac{8}{\lambda}+k\right) \cdot \frac{n\log n}{c^2}$, i.e., $O\left(\frac{n\log n}{r^2}\right)$ and it takes at most $\left(\frac{8}{\lambda}+k\right)\cdot\log n$, i.e., $O(\log n)$ time slots.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their comments which have greatly improved the presentation of this paper.

REFERENCES

[1] E. Kushilevitz and N. Nisan, *Communication Complexity*. Cambridge, U.K.: Cambridge Univ. Press, 1997.
[2] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.
[3] A. Giridhar and P. R. Kumar, "Computing and communicating functions over sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 755–764, Apr. 2005.
[4] S. Subramanian, P. Gupta, and S. Shakkottai, "Scaling bounds for function computation over large networks," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, 2007, pp. 136–140.
[5] N. Khude, A. Kumar, and A. Karnik, "Time and energy complexity of distributed computation in wireless sensor networks," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, 2005, pp. 2625–2637.
[6] A. E. Gamal, "Reliable communication of highly distributed information," in *Open Problems in Communication and Computation*, T. M. Cover and B. Gopinath, Eds. New York: Springer-Verlag, 1987, pp. 60–62.
[7] R. G. Gallager, "Finding parity in a simple broadcast network," *IEEE Trans. Inf. Theory*, vol. 34, no. 2, pp. 176–180, Mar. 1988.
[8] N. Goyal, G. Kindler, and M. Saks, "Lower bounds for the noisy broadcast problem," *SIAM J. Comput.*, vol. 37, no. 6, pp. 1806–1841, Mar. 2008.
[9] C. Li, H. Dai, and H. Li, "Finding the k largest metrics in a noisy broadcast network," in *Proc. Annu. Allerton Conf. Communication, Control, and Computing*, 2008, pp. 1184–1190.
[10] L. Ying, R. Srikant, and G. E. Dullerud, "Distributed symmetric function computation in noisy wireless sensor networks," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4826–4833, Dec. 2007.
[11] Y. Kanoria and D. Manjunath, "On distributed computation in noisy random planar networks," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, 2007, pp. 626–630.
[12] C. Dutta, Y. Kanoria, D. Manjunath, and J. Radhakrishnan, "A tight lower bound for parity in noisy communication networks," in *Proc. 19th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2008, pp. 1056–1065.

[13] C. Li and H. Dai, "Towards efficient designs for in-network computing with noisy wireless channels," in *Proc. IEEE Int. Conf. Computer Communications (INFOCOM)*, 2010, pp. 1–8.
[14] M. Franceschetti and R. Meester, *Random Networks for Communication*. Cambridge, U.K.: Cambridge Univ. Press, 2007.
[15] R. G. Gallager, *Information Theory and Reliable Communication*. Hoboken, NJ: Wiley, 1968.
[16] S. Rajagopalan and L. J. Schulman, "A coding theorem for distributed computation," in *Proc. Annu. ACM Symp. Theory of Computing (STOC)*, 1994, pp. 790–799.

Nikhil Karamchandani (S'05) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Bombay, in 2005, the M.S. degree in electrical engineering from the University of California at San Diego, La Jolla, in 2007, and is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering, University of California at San Diego. His research interests are in communication theory and include network coding, information theory, and random networks. Mr. Karamchandani received the California Institute for Telecommunications and Information Technology (CalIT2) fellowship in 2005.

Rathinakumar Appuswamy (S'05) received the B.Tech. and M.Tech. degrees in electrical engineering from Anna University, Chennai, and the Indian Institute of Technology, Kanpur, respectively, in 2004. He received the M.A. in mathematics from the University of California at San Diego, La Jolla, in 2008. He is currently a doctoral student at the University of California at San Diego, where he is a member of the Information and Coding Laboratory as well as the Advanced Network Sciences group. His research interests include network coding, communication for computing, and network information theory.

Massimo Franceschetti (SM'98) is an associate professor in the Department of Electrical and Computer Engineering, University of California at San Diego (UCSD), La Jolla. He received the Laurea degree (*magna cum laude*) in Computer Engineering from the University of Naples in 1997, and the M.S. and Ph.D. degrees in Electrical Engineering from the California Institute of Technology in 1999 and 2003. Before joining UCSD, he was a post-doctoral scholar at University of California at Berkeley for two years.

Prof. Franceschetti was awarded the C. H. Wilts Prize in 2003 for best doctoral thesis in Electrical Engineering at Caltech; the S. A. Schelkunoff award in 2005 for best paper in the IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION; an NSF CAREER award in 2006, an ONR Young Investigator award in 2007; and the IEEE Communications society best tutorial paper award in 2010.

He has held visiting positions at the Vrije Universiteit Amsterdam in the Netherlands, the Ecole Polytechnique Federale de Lausanne in Switzerland, and the University of Trento in Italy.

He is associate editor for communication networks of the IEEE TRANSACTIONS ON INFORMATION THEORY for 2009–2012 and has served as guest editor for two issues of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATION.

His research interests are in communication systems theory and include random networks, wave propagation in random media, wireless communication, and control over networks.