

# Towards Efficient Designs for In-network Computing with Noisy Wireless Channels

Chengzhi Li and Huaiyu Dai

Department of Electrical and Computer Engineering

North Carolina State University, Raleigh, NC

email: {cli3, hdai}@ncsu.edu

**Abstract**—In this paper we study distributed function computation in a noisy multi-hop wireless network, in which  $n$  nodes are uniformly and independently distributed in a unit square. We adopt the adversarial noise model, for which independent binary symmetric channels are assumed for any point-to-point transmissions, with (not necessarily identical) crossover probabilities bounded above by some constant  $\epsilon$ . Each node holds an  $m$ -bit integer per instance and the computation is started after each node collects  $N$  readings. The goal is to compute a global function with a certain fault tolerance, in this distributed setting; we mainly deal with divisible functions, which essentially covers the main body of interest for wireless applications. We focus on protocol designs that are efficient in terms of communication complexity. We first devise a general protocol for evaluating any divisible functions, addressing both one-shot ( $N = O(1)$ ) and block computation, and both constant and large  $m$  scenarios; its bottleneck in different scenarios is also analyzed. Based on this analysis, we then endeavor to improve the design for two special cases: identity function, and some restricted type-threshold functions, both focusing on the constant  $m$  and  $N$  scenario.

## I. INTRODUCTION

Networked systems of intelligent devices are playing an increasingly important role in our life. In particular, they will facilitate monitoring and control of the nation's critical infrastructures; seamless surveillance, intelligent transportation, and secure Internet are a few such examples.

Designing efficient protocols to facilitate information processing among distributed nodes is crucial to the success of these networked systems. While there has been extensive research in traditional distributed computing [1]–[3], the influence of channel noise is largely ignored. In previous study targeting VLSI or wireline networks, noise-free communications can be fairly assumed; instead some consideration was given on the fault tolerance of collapsed or Byzantine nodes. However, as we deal with networked information processing in wireless networks, consideration of noisy channels becomes necessary. Usually a protocol originally designed for noiseless channels fails in the noisy communication due to messages errors and discrepancy in individual interpretations of the

communication history. Also, the problems of interest in wireless applications are usually different; here we are interested in some real (possibly vector) functions of the data at all nodes, typically with physical meanings. A good example is the summary or statistics of collected data in wireless sensor networks.

Devising communication protocols with noisy channels imposes additional challenges. In some sense, it is the counterpart of Shannon's channel coding in the much more challenging network setting. In general, increase in communication complexity is inevitable even if a constant error tolerance is allowed, and any protocols working in the noisy environment should make this penalty as small as possible. Meanwhile, it is also required that the protocol be oblivious, i.e., the transmission schedule of nodes is pre-determined, independent of initial inputs and the communication history; this avoids transmission contention and out-of-order execution.

Time and message complexity are two key measures for the efficiency of distributed computing protocols. In this work, we concentrate on the latter, as communication cost typically dominates the energy consumption and life time of wireless networks. Also, we focus on the bit complexity, representing fundamental limits in the theoretical approaches. This naturally draws the connection with the theory of communication complexity [14]. Like computational complexity, communication complexity is an inherent property of a problem; it measures the hardness of a problem in terms of the communication (rather than the execution time) required for the most efficient solution.

Communication complexity of distributed computing in a noisy broadcast network was first considered by El Gamal in [5], where each of the  $n$  nodes, holding one binary input, can broadcast to all others through independent binary symmetric channels. Gallager showed that complexity  $O(n \log \log n)$  is achievable for computing any functions in a noisy broadcast network [6], which was further shown optimal for the identity function in [12]. In [13], Yao posed the question whether there exist nontrivial Boolean functions that can be computed with  $O(n)$  broadcasts; this is answered in the affirmative for the threshold functions in [14] with the independent and identically distributed (i.i.d.) random noise model, and in [7] for the OR function with the more realistic (and more general) adversarial model (where a "benign" adversary is allowed to arbitrarily reduce the error probability of each link at the

beginning, or even dynamically cancel any errors on the fly, so as to prevent the protocols from exploiting the stochastic regularities of the previous i.i.d. model). While most work in this area focuses on computing Boolean functions of binary variables, our recent work [16] extends the study to find the  $K$  largest integer values.

Due to concerns on energy consumption and scalability, transmissions are typically carried out in a multi-hop fashion in wireless networks. In [4] the authors exploited block computation to study the communication complexity of evaluating symmetric functions. In [8] the authors explored the minimal time and power consumption for evaluation of the max function. Both works in [4] and [8] focused on *noiseless* sensor networks. Distributed computing in a *noisy* multi-hop network is arguably more challenging, and so far there are still few works in this area. In [9] a protocol is proposed to compute the histogram with  $O(n \log \log n)$  transmissions per instance. In [11], an algorithm for the max function is proposed, taking advantage of the “witness discovery” protocol in [7] and the coding strategy in [10], and shown order optimal in both the number of transmissions and computation time. In this paper, we consider efficient protocol designs for computing divisible functions in a noisy multi-hop network, which constitutes the main body of interest for wireless applications. After devising a general protocol for evaluating any divisible functions we analyze its bottleneck in different scenarios, which may provide some insight for further studies. Then we endeavor to improve the design for two special cases: identity function, and some restricted type-threshold functions.

The remainder of this paper is organized as follows. The system model and some preliminaries are given in Section II. Our main results are summarized in Section III. A general protocol is proposed for divisible functions in Section IV, with performance analysis and further discussion. Then in Section V and VI a more efficient protocol is proposed for the identity function and some restricted type-threshold functions, respectively. The conclusion and future directions are provided in Section VII.

## II. PROBLEM FORMULATION

In this section, we first discuss the models and assumptions used in this work, then introduce some existing results needed for our analysis.

We use the following order notations throughout our paper. Given non-negative functions  $f(n)$  and  $g(n)$ :

- $f(n) = \Omega(g(n))$  if there exists a positive constant  $c_1$  and an integer  $k_1$  such that  $f(n) \geq c_1 g(n)$  for all  $n \geq k_1$ .
- $f(n) = O(g(n))$  if there exists a positive constant  $c_2$  and an integer  $k_2$  such that  $f(n) \leq c_2 g(n)$  for all  $n \geq k_2$ .
- $f(n) = \Theta(g(n))$  if  $f(n) = \Omega(g(n))$  and  $f(n) = O(g(n))$ .
- $f(n) = o(g(n))$  if there exists a positive constant  $k_3$  such that  $f(n) \leq c_3 g(n)$  for all  $c_3 > 0$  and  $n \geq k_3$ .
- $f(n) = \omega(g(n))$  if there exists a positive constant  $k_4$  such that  $f(n) \geq c_4 g(n)$  for all  $c_4 > 0$  and  $n \geq k_4$ .

### A. System Model

We consider a multi-hop wireless network with  $n$  nodes uniformly and independently distributed in a unit square. Each node  $i \in \{1, \dots, n\} \triangleq [n]$  holds an  $m$ -bit integer  $x_i(t)$  at time  $t$ , taking values from some finite set  $\chi$  ( $|\chi| \leq 2^m$ ). The computation is performed after each node collects a block of  $N$  readings. The goal is to calculate a divisible function  $f(x_1(t), x_2(t), \dots, x_n(t))$  correctly with a certain fidelity in this distributed setting. Without loss of generality, we assume that the result is made known to a special node, named sink node. This is common for applications in sensor networks; when necessary, the result at the sink node can be distributed to the whole network typically at a similar complexity as what shown below.

A divisible function  $f$  is defined as in [4], for which:

- $r = |\mathcal{R}(f, n)|$  is nondecreasing in  $n$ , where  $\mathcal{R}(f, n)$  is the function range;
- given any partition  $\Pi(S) = S_1, S_2, \dots, S_p$  of  $S \subset [n]$ , there exists a function  $g^{\Pi(S)}$ , such that for any  $\underline{x} \in \chi^n$

$$f(\underline{x}_S) = g^{\Pi(S)}(f(\underline{x}_{S_1}), f(\underline{x}_{S_2}), \dots, f(\underline{x}_{S_p})), \quad (1)$$

where  $\underline{x}_S = \{x_i\}$ ,  $i \in S$ .

Divisible functions essentially cover the main body of interest for distributed computing in wireless networks; identity function, histogram, parity, mean, max/min are a few examples.

We assume identical transmission power  $P$  for each node. For point-to-point transmissions in the network, two models are widely used in literature. Namely, a transmission is made from node  $X_i$  to node  $X_{R(i)}$  successfully if [17]

- (Protocol model): the distance between the transmitter and the receiver is no more than  $r_n$ , i.e.,  $|X_i - X_{R(i)}| \leq r_n$ , where  $r_n$  is the transmission range; for every node  $X_k$ ,  $k \neq i$ , transmitting at the same time,  $|X_k - X_{R(i)}| \geq (1 + \Delta)r_n$ , where  $\Delta$  is the protocol-specified guard-factor to limit interference.
- (Physical model): the signal to interference plus noise ratio (SINR) at the receiver is no less than a threshold  $\beta$ ,

$$\frac{\frac{P}{|X_i - X_{R(i)}|^\alpha}}{N_0 + \sum_{k \in \mathcal{T}, k \neq i} \frac{P}{|X_k - X_{R(i)}|^\alpha}} \geq \beta, \quad (2)$$

where  $\{X_k; k \in \mathcal{T}\}$  is the subset of nodes transmitting simultaneously, and  $N_0$  is the ambient noise power level. The signal power decays with distance  $d$  as  $\frac{1}{d^\alpha}$ , where  $\alpha > 2$  depends on the propagation environment.

Our proposed protocols work for both models. And we adopt the adversarial noise model, for which independent (but not necessarily identical) binary symmetric channels are assumed for any transmissions, with crossover probabilities bounded above by some constant  $\epsilon$ . In other words, for any transmission the received bit is flipped with some probability  $p \leq \epsilon$ .

### B. Preliminaries

In this subsection we introduce a few known results. The first is a generalization of the Chernoff bound.

1) *Hoeffding inequality* [23]:

*Lemma 1:* Let  $X_i$  ( $1 \leq i \leq n$ ) be  $n$  i.i.d. random variables over the interval  $[a, b]$ . For the sum of these variables  $S = \sum_{i=1}^n X_i$ , we have the inequity

$$P(S - E(S) \geq n\delta) \leq e^{-\frac{2n\delta^2}{b-a}}, \quad \delta > 0.$$

2) *Constant-rate, Constant-fraction, Minimum-distance Codes* [12], [24]:

*Lemma 2:* For  $\gamma \in (0, \frac{1}{2})$  there is an integer  $C_1 = C_1(\gamma)$  such that for each positive integer  $n$ ,  $\forall C \geq C_1$ , there is a binary code  $\Gamma_n$  of size  $2^n$  and length  $Cn$ , such that for all  $v, w \in \Gamma_n$  with  $v \neq w$ , the hamming distance  $d(v, w) \geq \gamma Cn$ .

Based on the above facts, the following result can be derived, which will be extensively used in our protocol design and analysis.

*Corollary 1:* With  $O(m)$  transmissions over a binary symmetric channel (with error probability bounded above by a constant  $\epsilon$ ), an  $m$ -bit integer can be correctly received by a receiver with probability at least  $1 - e^{-\gamma m}$ ,  $\forall \gamma > 0$ .

*Remark 1:* Any positive value can be chosen for  $\gamma$  without affecting the scaling law. Also,  $m$  does not need to be large.

### III. OVERVIEW OF MAIN RESULTS

In this paper, we present three protocols: a general protocol suitable for computing all divisible functions, and two specific efficient protocols, one for the identity function and one for some restricted type-threshold functions. The metric of interest is communication complexity of the protocol, i.e., the total number of bits exchanged in the network to get the task completed. We study the scaling law of this metric with respect to various parameters including  $n$ ,  $N$ , and  $m$ . Note that this metric is closely related to energy consumption. In particular, the complexity multiplied by a factor  $r_n^\alpha$ , where  $r_n$  is the transmission range in the protocol model, coincides with the energy usage in [9].

We address both one-shot ( $N = O(1)$ ) and block computation, and both constant and large  $m$  scenarios<sup>1</sup>. Our results are summarized below.

*Theorem 1:* The sink node can compute any divisible functions correctly with probability at least  $1 - o(1)$ , at the cost of

$$O\left(\frac{n}{N} \max(Nm, \log \log n) + \max(N \log r, \log n) \frac{n}{N \log n}\right) \quad (3)$$

transmitted bits per instance.

For some functions the general protocol actually achieve the best results we know in literature, while for some others improvement is still necessary. After analyzing the bottleneck of the general protocol we propose more efficient protocols for identity function and some type threshold functions as indicated below.

<sup>1</sup>large  $m$  has relevance, e.g., when node identities are used as inputs, a common practice in the study of distributed computing.

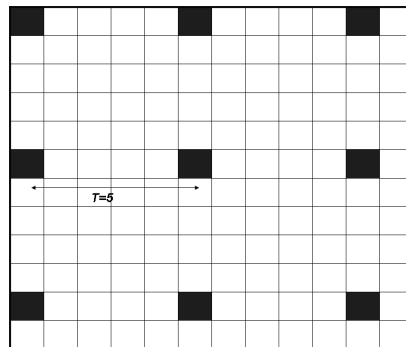


Fig. 1: TDMA scheduling scheme for  $T = 5$ .

*Theorem 2:* Identity function can be computed by the sink node correctly with probability at least  $1 - o(1)$  at the cost of  $O(n \sqrt{\frac{n}{\log n}})$  transmitted bits for constant  $m$  and  $N$ .

*Theorem 3:* If a type threshold function can be evaluated on a set of arguments with size bounded above by  $K (= o(\log n))$ , it can be computed by the sink node correctly with probability at least  $1 - O(1)$  at the cost of  $O(n \log \log K)$  transmitted bits for constant  $m$  and  $N$ .

### IV. A GENERAL PROTOCOL FOR DIVISIBLE FUNCTIONS

In this section we introduce a general protocol for computing any divisible functions. As a common approach, the unit square area is tessellated into cells with side  $c_n = \sqrt{\frac{2 \log n}{n}}$ . The transmission is restricted between neighboring cells. The following lemma (lemma 3.1 in [20]) shows that the number of nodes in each cell is  $\Theta(\log n)$  with high probability<sup>2</sup>.

*Lemma 3:* Let  $K > \frac{1}{\log(4/\epsilon)}$ , and let  $u^* \in (0, 1)$  be the sole root of the equation

$$-u^* + (1 + u^*) \log(1 + u^*) = 1/K. \quad (4)$$

Then the following limit holds for any  $u > u^*$ :

$$\lim_{n \rightarrow \infty} P_r \{ \max_i |n_i - K \log n| \leq uK \log n \} = 1. \quad (5)$$

where  $n_i$  is the number of nodes in cell  $i$ .

#### A. Protocol

The proposed protocol is composed of an intra-cell protocol and an inter-cell protocol, corresponding to two sequential executive stages: intra-cell processing and inter-cell routing, respectively. In the intra-cell processing, a cell head  $h_i$  is designated in the  $i$ th cell, to coordinate the protocol execution. This node takes (much) more responsibility besides its normal role as other nodes in the protocols; its role can be rotated among nodes in the cell. The intra-cell protocol ends up with the local information in a cell aggregated at its cell head. In the inter-cell protocol, the aggregated information is routed over the cell heads to the sink node, with possible further processing along the way.

<sup>2</sup>approaching 1 as  $n \rightarrow \infty$

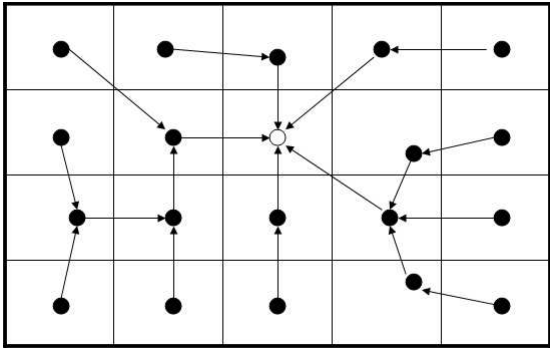


Fig. 2: spanning tree rooted at the sink

For the purpose of generality our intra-cell protocol computes the identity function, i.e., the cell heads collect all the information in their cells. To limit the interference caused by simultaneous transmissions, the  $T^2$ -TDMA cell scheduling scheme is adopted during the execution of the intra-cell protocol, where the selection of constant  $T$  is given in Appendix I. Each time slot is assigned to one of  $T^2$  cells in a  $T \times T$  super cell and the concurrently transmitting cells are at least  $T$  cells away from each other. Fig. 1 depicts the 25-TDMA scheduling scheme, where nodes in the shaded cells transmit simultaneously. The intra-cell protocol is executed in each cell when it is active, and nodes in the active cell take turns<sup>3</sup> to transmit their information.

For the  $i$ th cell, assuming that its members are ordered from 1 to  $n_i$ , the intra-cell protocol is executed in two phases, which follows from a modification of the work in [12], [16].

*Intra-cell protocol:*

- (i) Each node in the  $i$ th cell encodes its data, composed of  $N$  observations, into a codeword with size  $O(\max(Nm, \log(n_i)))$  and broadcasts it.
- (ii) After decoding the codewords from all the other nodes in its cell, each node concatenates all the results (including its own integer) into a word and encodes it into a codeword with size  $O(Nmn_i)$ , which is equally partitioned into  $n_i$  blocks. Then each node  $j$  in the  $i$ th cell takes turn to broadcast the  $j$ th block of its codeword. The cell head  $h_i$  collects all the blocks from its cell and decodes it.

After executing the intra-cell protocol, the information is accumulated at the cell heads. Similar to the architecture of [4], in the inter-cell protocol, a random spanning tree (for example, see Fig. 2) rooted at the sink node is formed beforehand, whose vertices include all the cell heads and whose edges only connect neighboring cells. The information flow is from the leaves to the root and each cell head only transmits once<sup>4</sup>.

*Inter-cell protocol:*

<sup>3</sup>Each node is assumed to broadcast in its designated slot. Just as tessellation, this is another common approach in the theoretical study, and also conforms to the requirement that the protocols dealing with noisy communications be oblivious.

<sup>4</sup>Any scheduling introduced to the inter-protocol only induce constant factor to its time complexity, which is not our focus in this paper.

- (i)  $h_i$  calculates the target divisible function  $f(\cdot)$  with the aggregated information from its own cell and its children.
- (ii)  $h_i$  encodes the result in the last step into a codeword with size  $O(\max(N \log r, \log n))$  and transmits it to its parent node.

## B. Analysis

We analyze the performance of the intra-cell protocol first. It's easy to check that the complexity of the intra-cell protocol in the  $i$ th cell for one instance is  $O(\max(Nm, \log n_i)n_i \frac{1}{N})$ . Since  $n_i = \Theta(\log n)$  and there are  $\Theta(\frac{n}{\log n})$  cells, the total complexity is  $O(\frac{n}{N} \max(Nm, \log \log n))$ . As to the error probability, we have the following result, the proof of which is given in Appendix II.

*Proposition 1:* The cell head  $h_i$  correctly retrieves all the information,  $mNn_i$  bits in total, with the probability at least  $1 - O(\frac{2}{n^2})$ .

By the union bound, the probability that all the cell heads acquire the correct information in their cells is at least  $1 - O(\frac{2}{n \log n})$ .

For the inter-cell protocol there are  $\Theta(\frac{n}{\log n})$  nodes in total on the spanning tree composed of the cell heads. Since each node only transmits once the complexity of the inter-cell protocol for one instance is  $O(\frac{1}{N} \max(N \log r, \log n) \frac{n}{\log n})^5$ .

We now consider the error probability for the inter-cell protocol. A codeword can be correctly decoded by its parent node with probability at least

$$\begin{aligned} \Pr \quad & \text{(a codeword is decoded correctly)} \\ & \geq 1 - e^{-\gamma \max(N \log r, \log n)}, \end{aligned} \quad (6)$$

for some  $\gamma > 1$ , according to Corollary 1.

By the union bound

$$\begin{aligned} \Pr \quad & \text{(all the codewords are decoded correctly)} \\ & \geq 1 - \frac{n}{\log n} e^{-\gamma \max(N \log r, \log n)} \\ & \geq 1 - \frac{1}{n^{\gamma-1} \log n}. \end{aligned} \quad (7)$$

Combining the analysis for both the intra-cell and inter-cell protocols we reach the conclusion in Theorem 1.

## C. Discussion

Since our general protocol can compute any divisible function, it is interesting to examine its efficiency and potential bottlenecks. Depending on different relations between the system parameters, we differentiate a few cases as below:

1) *One shot, constant-size data case:* In this scenario  $N = O(1)$  and  $m = O(1)$ . From Theorem 1, we can see in such a case the complexity becomes  $O(n \log \log n + \max(\log r, \log n) \frac{n}{\log n})$ .  $r$  plays an important role here and provides us some insights for the protocol design.

- $r = O(n^{\log \log n})$ : in this case the complexity of our general protocol becomes  $O(n \log \log n)$ , which provides

<sup>5</sup> $m$  may play a role here through  $r$ .

the best upper bound we know in literature on some individual functions. For example, our result for the histogram computation ( $r = O(n^{|\chi|})$ ) is the same as what is achieved in [9]. For the parity function ( $r = O(1)$ ), the best known protocol is given in [6] for a noisy *broadcast* network with the complexity of  $O(n \log \log n)$ . Our result shows that actually this result is achievable for a noisy *multi-hop* network as well. In addition, when  $r = O(n)$  the complexity of the inter-cell protocol is  $O(n)$ . Obviously for all non-degenerate functions,  $\Omega(n)$  transmissions is needed, even in a *noiseless, broadcast* network. Thus, any effort to improve the inter-cell is not necessary and the bottleneck lies in the *intra-cell* protocol. In Section VI we'll show how to improve the efficiency of the intra-cell protocol for some particular type-threshold functions.

- $r = \Omega(n^{\log \log n})$ : In this case it is easily seen that the complexity of our proposed protocol is  $O(\log r \frac{n}{\log n})$ . This reveals an interesting point that, when the function range  $r$  is large enough the computation bottleneck lies in the *inter-cell* protocol.

The identity function ( $r = |\chi|^n$ ) belongs to this category, and our general protocol can compute it with the complexity of  $O(\frac{n^2}{\log n})$ . We give a more efficient inter-cell protocol for computing the identify function in the next section.

2) *Block computation*: This scenario corresponds to  $mN = \Omega(1)$ . Intuitively when either  $m$  or  $N$  is large enough, the computation efficiency will be improved through block coding. By examining the complexity expression of our protocol in (3), we find that this intuition is only true for the intra-cell protocol. In particular, the first term of the complexity expression suggests that when  $Nm = \Omega(\log \log n)$ , a tight bound  $O(nm)$  for the intra-cell protocol is achieved. For the inter-cell protocol, the conclusion can not be made without the information of  $r$ . We can see from the second term of the complexity expression that, if  $r = \Omega(n)$  block computation can not help for the inter-cell propagation in our general protocol; the computation of the identity function belong to this case. In contrast, for the max function with constant  $m$ , block computation helps both for the intra-cell processing and inter-cell propagation.

To summarize, our general protocol is actually quite efficient; for example, it admits the best known results about the computation of the histogram and parity function in literature. On the other hand, we also identify the scenarios for potentially further improvement. In the following two sections, we discuss two such improvements.

## V. IMPROVEMENT FOR THE IDENTITY FUNCTION

Identity function attracts much attention in the studies of distributed computing since it is the mother function of other other functions. In the one-shot, constant-size data case, it is known that in a noisy broadcast network with  $n$  nodes the (order) optimal complexity for identity function is  $\theta(n \log \log n)$  [12], which is achieved by the intra-cell protocol above. Therefore we focus on the improvement of the inter-cell protocol

for the identity function in this section. As we discussed in the last section, this is also the bottleneck of the computing task.

We consider the one shot case with constant  $m$ . In the new inter-cell protocol each cell head codes its aggregated information, obtained from the intra-cell protocol, into a codeword with size  $O(\log n)$ . Each codeword is transported to the sink node through the shortest path routing policy (see Fig. 3 for an example) and appropriate scheduling. The decode-and-forward scheme is adopted for the information's relay. The total number of hops  $N_h$  can be computed as:

$$\begin{aligned} N_h &= \sum_{i=1}^{\Theta(\sqrt{\frac{n}{\log n}})} 8i \cdot i \\ &= \Theta\left(\frac{n}{\log n} \sqrt{\frac{n}{\log n}}\right), \end{aligned} \quad (8)$$

due to the fact that there are  $8i$  nodes in the  $i$ th layer which are  $i$  hops away from the sink node (and the sink node lies in layer 0). Therefore, communication complexity of this inter-cell protocol is  $\Theta(\log n N_h) = \Theta(n \sqrt{\frac{n}{\log n}})$ .

It's easy to check that in each hop a codeword can be correctly decoded by the receiver with probability at least

$$\begin{aligned} Pr \quad &(a \text{ codeword is decoded correctly}) \\ &\geq 1 - e^{-\gamma \log n}, \end{aligned} \quad (9)$$

for some  $\gamma > 3/2$ , according to Corollary 1.

By the union bound

$$\begin{aligned} Pr \quad &(all \text{ the codewords are decoded correctly}) \\ &\geq 1 - \frac{n}{\log n} \sqrt{\frac{n}{\log n}} e^{-\gamma \log n} \\ &\geq 1 - \frac{1}{n^{\gamma-3/2} (\log n)^{3/2}} \\ &= 1 - o(1). \end{aligned} \quad (10)$$

*Remark 2*: This inter-cell protocol improves the computing efficiency for the identity function, but not necessarily for other functions. As we discussed in IV-C, for the  $r = O(n^{\log \log n})$  case, the bottleneck lies in the intra-cell protocol, so there is no need to improve the inter-cell protocol. For the other case, this new protocol outperforms the one presented in the last section only when  $r = \omega(c\sqrt{n/\log n})$  for some constant  $c$ .

## VI. IMPROVEMENT FOR RESTRICTED TYPE-THRESHOLD FUNCTIONS

In this section we'll show that the intra-cell protocol could be further improved for some restricted type-threshold functions  $f$ .

*Type-Threshold Function* [4]: A function  $f(\cdot)$  is said to be type-threshold if there exists a nonnegative  $|\chi|$ -dimensional vector  $\theta$ , called the threshold vector, such that  $f(\underline{x}) = f'(\tau(\underline{x})) = f'(\min(\tau(\underline{x}), \theta))$ , for all  $\underline{x} \in \mathcal{X}^n$ , where  $\tau(\underline{x})$  is the histogram<sup>6</sup> of  $\underline{x}$  and  $\min$  signifies element-wise minimum.

<sup>6</sup>The histogram of the vector  $\underline{x} \in \mathcal{X}^n$  is defined as  $\tau(\underline{x}) = [\tau_1(\underline{x}), \tau_2(\underline{x}), \dots, \tau_{|\chi|}(\underline{x})]$ , where  $\tau_i(\underline{x}) := |\{j : x_j = i\}|$ , the number of occurrence of  $i$  in  $\underline{x}$

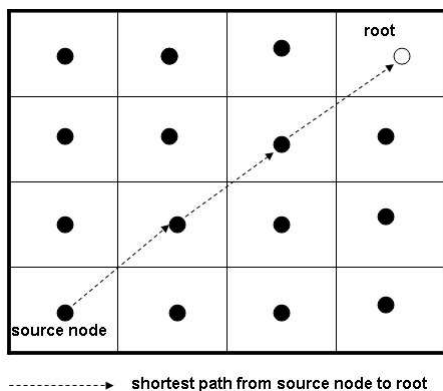


Fig. 3: shortest path routing scheme

*Remark 3:* Type-threshold functions is a subset of symmetric functions, whose values are solely determined by the histograms of the input data. For example, the max function is a type threshold function with a threshold vector  $[1, 1, \dots, 1]$ ; we can find the maximum among the inputs by searching the first non-zero position from right in the vector  $\min(\tau(\underline{x}), \theta)$ . Note that this definition implies that the value of the type-threshold function can be determined by a subset  $S$  of the total inputs, i.e.,  $f(\underline{x}) = f(\underline{x}_S)$ . For example, as to the max function,  $S$  contains the single index  $i$  such that  $x_i$  is the maximum of the input. Of course, this subset is typically input-dependent and not known a priori. However, it is possible to design efficient protocols that lead to the discovery of this subset. To this purpose, we further impose the restriction that the size of  $S$  is bounded above by a known value  $K (= o(\log n))$ . For example  $|S| = 1$  for the max function and the indicator function, and  $|S| = K$  for the function that computes the  $K$  largest values (which need not be distinct). However, the function that computes the  $K$ th largest value (required to be strictly smaller than the  $(K - 1)$ th one) does not satisfy our condition, even though it is also a type-threshold function; the reason is that the size of its defining subset can not be pre-determined but depends on the input instead.

It's not difficult to check that for this class of functions, a tight bound  $\Theta(mn)$  in communication complexity is achieved by our general protocol (Theorem 1) in most block computation scenarios. Therefore we focus on the one shot case ( $N = O(1)$ ) with constant  $m$ , for which the general protocol gives the complexity of  $O(n \log \log n)$ . As we discussed in Sec.IV-C, the bottleneck of computation for this type of functions lies in the intra-cell protocol. Inspired by the study in [7], we design a more efficient intra-cell protocol for this class of functions with complexity  $O(n \log \log K)$  and a fault tolerance  $Q$ , which is an arbitrary small constant. In the following, we take the function, finding the  $K$  largest values, to describe our design for concreteness. It's straightforward to extend it to computing other functions.

#### A. The intra-cell protocol

In cell  $i$ , we divide  $n_i$  nodes into  $\frac{n_i}{4K}$  groups, each of size  $4K$ . A cell head  $h_i$  is designated and the protocol is described recursively (on the size of the network) as below:

- (1) The cell head  $h_i$  collects all the measurements in its cell. We differentiate two scenarios based on the value of  $K$ :
  - $K = O(1)$ : in this case each node encodes its integer into a codeword with size  $O(\max(m, \log \frac{K}{Q}))$  and transmits it to  $h_i$ ;
  - $K = \Omega(1)$ : each group is further divided into subgroups of size  $\log 4K$ . And then the intra-cell protocol discussed in Sec. IV is applied in each subgroup, with  $n_i$  there replaced with  $\log 4K$ .
- (2) For each group,  $h_i$  compares the  $4K$  values and determines the indices of the  $K$  largest values. And then  $h_i$  assigns '1' to nodes holding the  $K$  largest values and '0' to others. These  $n$  assigned bits are concatenated into a word, encoded with an error correcting code of size  $O(n_i)$  and broadcasted.
- (3) Let  $J$  be the set of winning nodes in (2). The protocol proceeds by executing the protocol on  $J$  (of size  $n_i/4$ ) for 3 independent times. In each time, the head node  $h_i$  obtains from the protocol output  $K$  indices corresponding to the  $K$  (nominally) largest values (the result from one execution of the protocol on  $n_i/4$  nodes); it takes the majority of the three outputs as the result for the protocol on cell  $i$  (or arbitrarily decides if there is no majority).
- (4)  $h_i$  encodes the  $K$  indices found in (3) into a codeword with size  $O(K \log n)$  and broadcasts it. The nodes selected encode their data into codewords with size  $O(\log n)$  and send them to  $h_i$ . The protocol ends up with  $h_i$  knowing the  $K$  largest values.

*Remark 4:* The recursion happens on the third step, where we obtain a set of reduced size (by a factor of 4), and apply this very protocol to the winning set for three times. To complete the description of the protocol, consider the execution of this protocol on a group of  $n_i = 4K$  nodes. First, step 1 above is executed. Then  $h_i$  compares the  $4K$  values and determines the indices of the  $K$  (nominal) largest values.

This protocol is non-oblivious and can be turned into an oblivious one by the 'helper' idea in [7] without degrading the performance.

#### B. Analysis

*Proposition 2:* With  $O(n \log \log K)$  bits transmitted, the  $K$  largest values are selected by cell heads with probability  $1 - O(1)$ .

The proof is given by the Appendix III.

The inter-cell protocol is the same as the one in the general protocol and the complexity is  $O(\max(K, \log n) \frac{n}{\log n}) = O(n)$ .

Combining the results for the intra-cell protocol and the inter-cell protocol, we get Theorem 3.

## VII. CONCLUSION AND FURTHER WORK

In this paper we first designed a general protocol for computing any divisible functions reliably with high probability in a noisy multi-hop network, with communication complexity  $O(\frac{n}{N} \max(Nm, \log \log n) + \max(N \log |R(f, n)|, \log n) \frac{n}{N \log n})$ . The general protocol performs efficiently for some functions but inefficiently for others. After analyzing its bottleneck in different scenarios, we endeavor to propose more efficient protocols for the identity function and some restricted type-threshold functions.

In our future work we'll devote our effort on the tight lower bound for the distributed function computation in the noisy multi-hop networks.

### APPENDIX I SELECTION OF $T$

In the  $T^2$ -TDMA cell scheduling scheme each time slot is assigned to one of  $T^2$  cells in a  $T \times T$  supper cell and the concurrently transmitting cells are at least  $T$  cells away from each other. Selection of  $T$  depends on the transmission model. In the protocol model,  $T$  is set to  $\lceil 2 + 2\sqrt{2}(1 + \Delta) \rceil$  to guarantee that there is only one transmitter within a distance  $(1 + \Delta)r_n$  for each receiver.

In the physical model, we assume the transmission power  $P = P_0(c_n)^\alpha$ , where  $P_0$  is a constant.  $T$  is chosen as follows. Suppose node  $X_i$  transmits to node  $X_{R(i)}$ , which is at most  $2\sqrt{2}c_n$  distance away. There exist 8 first tie interfering signals at least  $(T - 1)c_n$  away from the receiver, and 16 second tie ones at least  $(2T - 1)c_n$  away. In general there are  $8k$   $k$ th tie interfering signals with a distance at least  $(Tk - 1)c_n$ . The SINR of the receiver  $X_{R(i)}$  is thus given by:

$$\begin{aligned} \text{SINR} &= \frac{\frac{P}{|X_i - X_{R(i)}|^\alpha}}{N_0 + \sum_{k \neq i}^{k \in T} \frac{P}{|X_k - X_{R(i)}|^\alpha}} \quad (11) \\ &\geq \frac{\frac{P}{(2\sqrt{2}c_n)^\alpha}}{N_0 + \sum_{k \neq i}^{\infty} \frac{8k}{[(kT-1)c_n]^\alpha} P} \\ &= \frac{\frac{P_0}{(2\sqrt{2})^\alpha}}{N_0 + \frac{8P_0}{T^\alpha} \sum_{k \neq i}^{\infty} \frac{k}{[(k-1/T)]^\alpha}} \\ &> \beta \end{aligned}$$

where we select a sufficiently large  $T$  to satisfy the last inequality, which is feasible since  $\sum_{k \neq i}^{\infty} \frac{k}{[(k-1/T)]^\alpha}$  converges when  $\alpha > 2$ .

### APPENDIX II PROOF OF PROPOSITION 1

We start the proof by denoting  $U$  the set of nodes in cell  $i$ , which don't decode all the codewords in its cell correctly. According to Lemma 2, assume that each  $mN$ -bit integer is encoded by a codeword  $v \in \Gamma_n$  with length  $C_1 \max(mN, \log n_i)$ . A receiver gets a noisy copy  $v'$  of  $v$  and decodes it as  $w \in \Gamma_n$  which minimizes  $d(v', w)$ . The decoding error of any  $mN$ -bit integer at any node is bounded

above by  $\exp(-2(\frac{1}{2}\gamma - \epsilon)^2 C_1 \max(mN, \log n_i))$  according to Lemma 1. Summing over the  $n_i$  codewords and for a node  $I$ ,

$$\begin{aligned} \Pr[I \in U] &\leq n_i \exp(-2(\frac{1}{2}\gamma - \epsilon)^2 C_1 \max(mN, \log n_i)) \\ &\leq n_i \exp(-2(\frac{1}{2}\gamma - \epsilon)^2 C_1 \log n_i). \end{aligned}$$

Therefore

$$\Pr[|U| \geq \epsilon n_i] \leq w^{\epsilon n_i} 2^{n_i} < 4^{-n_i} \text{ for } w < (\frac{1}{8})^{1/\epsilon},$$

where  $w = n_i \exp(-2(\frac{1}{2}\gamma - \epsilon)^2 C_1 \log n_i)$ . The last inequality can be achieved with sufficiently large  $C_1$ .

In phase 2, we encode the  $mNn_i$ -bit word by a codeword  $v_2 \in \Gamma'_n$  with length  $C_2 mNn_i$ . The decoding error at the head node is upper bounded by  $\Pr(d(v'_2, v_2) \geq \frac{1}{2}\gamma C_2 mNn_i) = \Pr(d(v'_2, v_2) \geq 3\epsilon C_2 mNn_i)$  for  $\gamma = 6\epsilon$ . The decoding error occurs as the blocks of bits from some node  $i \in U$  may be erroneous or bits may be corrupted by noise during transmission. Therefore, we have

$$\begin{aligned} \Pr(d(v'_2, v_2) \geq 3\epsilon C_2 mNn_i) &\leq \Pr[C_2 mN|U| \geq C_2 mN\epsilon n_i] + \Pr[N_e \geq 2mNC_2\epsilon n_i] \\ &\leq 4^{-n_i} + \exp(-2mNC_2\epsilon^2 n_i) \\ &< 2 \cdot 4^{-n_i} \text{ for } C_2 \geq 1/\epsilon^2 \\ &= O(1/n^2), \end{aligned}$$

where  $N_e$  is the number of bits corrupted by noise.

### APPENDIX III PROOF OF PROPOSITION 2

*Proof:* We analyze the complexity first. For the case  $K = \Omega(1)$ , in each group (of size  $4K$ ), each node transmits  $O(\log \log K)$  bits.  $K$  nodes are chosen into the next round. Therefore, in cell  $i$  the number of bits transmitted from the nodes to  $h_i$  during the recursion is:

$$\begin{aligned} C_n &= O(\log \log K)n_i + 3C_{(K \frac{n_i}{4K})} \quad (12) \\ &= O(\log \log K)(n_i + \frac{3n_i}{4} + \frac{9n_i}{16} + \dots) \\ &\leq 4O(n_i \log \log K), \end{aligned}$$

where  $n_i = \Theta(\log n)$  is the number of nodes in cell  $i$ . The complexity of the index-broadcasting in step (2) is  $O(n_i + n_i/4 + n_i/16 + \dots) = O(n_i)$  and the complexity in step (4) is  $O(K \log n_i)$ . Therefore the communication complexity in cell  $i$  is  $O(n_i \log \log K)$ . Since there are  $\Theta(n/\log n)$  cells, the total complexity of the intra-cell protocol is  $O(n \log \log K)$ .

For  $K = O(1)$  case, each node transmits  $O(\max(m, \log \frac{K}{Q}))$  bits in each group. Then following the same proof above we conclude that the complexity of the intra-cell protocol is  $O(n \max(m, \log \frac{K}{Q})) = O(n)$  (since  $K, Q, m$  are all constants).

Now, we analyze the error probability. For ease of description, the nodes holding the  $K$  largest values are called target nodes. Note that the target nodes could be distributed in at most  $K$  groups, located in at most  $K$  cells.

An error may occur if either of the following two cases happens:

- not all the target nodes, are selected by the cell heads (corresponding to the first three steps);
- the cell heads receive incorrect information from the selected nodes (corresponding to step (4)).

To analyze the first error, we define the following three events:

$e_1$ : the cell heads receive wrong data from at least one group containing one or some target nodes at step 1;

$e_2$ : the nodes receive incorrect indices broadcasted by the cell heads at step 2;

$e_3$ : an error happen during one recursion at step 3, which includes the operation in step 1 and 2.

Essentially we would like to show that  $p(e_3)$  remains bounded during recursion. However,  $e_2$  leads to error propagation in the recursion and renders the analysis intractable. Therefore, we first assume  $P(e_2) = 0$ , i.e., all the nodes receive the indices broadcasted by the cell heads in step (2) correctly. The information from a group (of  $4K$  nodes) is received incorrectly with probability at most  $\frac{1}{K^\gamma \log K}$  ( $\frac{Q^\gamma}{K^\gamma}$ ) for  $K = \Omega(1)$  ( $K=O(1)$ ) case. Therefore the probability of  $e_1$  is at most  $\frac{1}{K^{\gamma-1} \log K} < Q^2$  ( $\frac{Q^\gamma}{K^{\gamma-1}} < Q^2$ ), for certain  $\gamma$ , due to the fact that the  $K$  target nodes come from at most  $K$  groups. For the probability of  $e_3$ , it's easy to check  $P(e_3) < Q$  when  $n = 4K$ . Assume for  $n'/4$  nodes  $P(e_3) < Q$ . Executing the protocol on the  $n'/4$  nodes three times results in an error probability of at most  $3Q^2$ . By induction, for  $n'$  nodes the probability of  $e_3$  is  $Q^2 + 3Q^2 < Q$  for small  $Q$ . Therefore  $P(e_3)$  remains bounded during recursion.

Let us go back to examine the probability of  $e_2$ . There are at most  $O(\log n)$  recursions in total since the number of nodes in each recursion is reduced exponentially. In each recursion, the probability that not all the nodes receive the broadcasted indices correctly is  $\frac{n}{e^{\gamma n}}$ ,  $\gamma > 0$ . Therefore, nodes may receive wrong indices during the protocol's execution with error probability at most  $\frac{n \log n}{e^{\gamma n}}$ . Thus the probability of the first error is bounded above by  $Q + \frac{n \log n}{e^{\gamma n}}$ .

To check the probability of the second error,  $\Theta(K \frac{n}{\log n})$  nodes in total are selected after execution of the intra-cell protocol and each selected node transmits its data to its corresponding cell head correctly with possibility at least  $1 - \frac{1}{n^2}$  according to lemma 2. By the union bound the error probability of the second error is at most  $\frac{K}{n \log n}$ . Therefore, the total error probability for the intra-cell protocol is  $Q + \frac{n \log n}{e^{\gamma n}} + \frac{K}{n \log n} = O(1)$ .

□

## REFERENCES

- [1] E. Kushilevitz and N. Nisan, *Communication Complexity*, Cambridge Univ. Press, 1997.
- [2] N. A. Lynch, *Distributed Algorithms*, San Francisco, CA: Morgan Kaufmann, 1997.
- [3] H. Attiya and J. Welch, *Distributed Computing: Fundamentals, Simulations, and Advanced Topics*, second edition, Wiley-Interscience, 2004.
- [4] A. Giridhar and P. R. Kumar, "Computing and communicating functions over sensor network," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, Apr. 2005.
- [5] A. El Gamal, Open problem presented in the 1984 Workshop on Specific Problems in Communication and Computation, sponsored by Bell Communication Research, 1984.

- [6] R. Gallager, "Finding parity in a simple broadcast network," *IEEE Trans. Info. Theory*, vol. 34, no. 2, pp. 176-80, Mar. 1988.
- [7] I. Newman, "Computing in fault tolerance broadcast networks," *IEEE Annual Conference on Computational Complexity (CCC)*, 2004.
- [8] N. Khude, A. Kumar, and A. Karnik, "Time and energy complexity of distributed computation in wireless sensor networks," in *Proc. IEEE INFOCOM*, 2005.
- [9] L. Ying, R. Srikant, and G. E. Dullerud, "Distributed symmetric function computation in noisy wireless sensor network," *IEEE Trans. Infor. Theory*, 2007.
- [10] S. Rajagopalan and L. Schulman, "A coding theorem for distributed computation," *Proc. 26th Annual ACM Symp. Theory of Comp.*, 1994.
- [11] Y. Kanoria and D. Manjunath, "On distributed computation in noisy random planar networks," *Proc. IEEE International Symposium on Information Theory*, Nice, France, June 2007.
- [12] N. Goyal, G. Kindler and M. Saks, "Lower bounds for the noisy broadcast problem," *IEEE Symposium on Foundations of Computer Science*, 2005.
- [13] A. Yao, "On the complexity of communication under noise," invited talk in the 5th ISTCS Conference, 1997.
- [14] E. Kushilevitz and Y. Mansour, "Computation in noisy radio networks," *ACM-SIAM Symp. Discrete Algorithms*, 1998.
- [15] U. Feige and J. Kilian, "Finding OR in a noisy broadcast network," *Information Processing Letters*, vol. 73, no. 1-2, 2000.
- [16] C. Li, H. Dai and H. Li, "Finding the  $K$  largest metrics in a noisy broadcast network," *Allerton Conference on Communication, Control and Computing*, 2008.
- [17] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Infor. Theory*, vol. 46, no. 2, Mar. 2000.
- [18] Feng Xue and P. R. Kumar, *Scaling Laws for Ad Hoc Wireless Networks: An Information Theoretic Approach*, Delft, The Netherlands, 2006.
- [19] P. Gupta and P. Kumar, "Critical power for asymptotic connectivity in wireless networks," *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W. H. Fleming*, W. McEneaney, G. Yin, and Q. Zhang, Eds. Boston, MA: Birkhauser, 1998
- [20] F. Xue and P. Kumar, "The number of neighbors needed for connectivity of wireless networks," *Wireless network*, vol. 10, no. 2, Mar. 2004.
- [21] D. Mosk-Aoyama and D. Shah. "Computing separable functions via gossip," *ACM Principles of Distributed Computing*, Sep. 2007.
- [22] S. Subramanian, P. Gupta and S. Shakkottai, "Scaling Bounds for Function Computation over Large Networks," *Proc. IEEE International Symposium on Information Theory*, Nice, France, June 2007.
- [23] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *J. Amer. Stat. Assoc.*, vol. 58, pp. 13-30, 1963.
- [24] J. H. van Lint, *Introduction to coding theory*, 3rd Edition, Springer-Verlog, 1999.