Silence-Based Communication

Anand K. Dhulipala, Christina Fragouli, *Member, IEEE*, and Alon Orlitsky, *Fellow, IEEE*

***Abstract*—Communication complexity—the minimum amount of communication required—for computing a function of data held by several parties is studied. A communication model where silence is used to convey information is introduced. For this model the worst case and average-case complexities of symmetric functions are studied. For binary-input functions the average- and worst case complexities are determined and the protocols achieving them are described. For functions of nonbinary inputs one-round communication, where each party is restricted to communicate in consecutive stages, is considered and the extra amount of communication required by one- over multiple-round communication is analyzed. For the special case of ternary-input functions close lower and upper bounds on the worst case one-round complexity are provided and protocols achieving them are described. Protocols achieving the average-case one-round complexity for ternary-input functions are also described. These protocols can be generalized to inputs of arbitrary size.**

***Index Terms*—Communication complexity, function computation, multiple-round, one-round, sensor networks, symmetric functions.**

I. Introduction

In several applications, a system needs to compute a function whose value depends on data distributed among several parties. As the whole data is available to none of the parties, they need communicate with each other to accomplish their task. The least amount of communication required to compute the function is called its *communication complexity*.

The concept of communication complexity is relevant in many practical applications such as very large scale integration (VLSI) circuit design and sensor networks, where one wants to minimize the amount of energy used by the various system components. Since the components consume energy to communicate it is important to minimize the total amount of communication by decreasing the number of signals exchanged by them.

For example, consider a sensor network, a collection of many spatially distributed devices that monitor and relay data. Typically, the devices, called nodes, collect and communicate information to a central processor, called a satellite, that computes a predetermined function of the combined inputs it receives. The information collected by the sensors could be the temperature at various locations, and the function to be computed by the satellite could be the average or maximum temperature.

Often, the sensors are small, have tiny batteries, and are not easily accessible. It is therefore desirable to find communication protocols that minimize the number of transmissions and thereby conserve their energy. In this paper, we propose using silence as an energy-efficient means of communication.

To formalize silence, we consider *pulse communication* where at each time unit a node can either be silent, or emit an energy *pulse*. Unlike standard *bit communication*, where a node transmits either zero or one, the pulse itself conveys no information except for its existence. It can be thought of as a beam of light that either does or does not exist. In bit communication there is an implicit assumption that the cost of communicating the bit zero is the same as that of the bit one. Pulse communication can be thought of as associating zero cost with one of the bits. Since staying silent has zero communication cost, we seek to minimize the total number of pulses communicated, as opposed to minimizing the total number of bits.

Communication complexity was first analyzed for two-party communication for functions of continuous variables by Abelson [1], and for discrete variables by Yao [2]. See [3], [4] for an overview. Multiparty functions were considered, e.g., in [5]–[12] and in the context of sensor networks in [13]–[18]. In this paper, we study multiparty pulse communication where a party either sends a pulse or stays silent to communicate.

Pulse and bit communication models are clearly special cases of the more general model of communication with unequal-cost bits.

Communication with symbols of unequal cost was considered in [19]–[27]. However, these works generally considered block coding of probabilistic sources, rather than the single-instance function computation analyzed here.

Motivated by sensor networks, we consider multiparty pulse communication. We first study binary-input symmetric functions. We derive explicit expressions for the worst case and average-case pulse communication complexities and describe protocols that achieve them.

We next consider functions of nonbinary inputs. For these functions, we restrict ourselves to one-round communication. In one-round communication, each party is restricted to communicate in consecutive stages (see Example 1).

We first analyze the cost of restricting to *one-round* communication. We compare the one-round complexity to the unrestricted multiple-round complexity. For pulse communication, we show that one-round complexity is greater than the unre-

Manuscript received November 06, 2007; revised November 02, 2008. Current version published December 23, 2009. The work by A. K. Dhulipala and A. Orlitsky was supported by the National Science Foundation (NSF) under Grant 0514973. The work of C. Fragouli was supported by the Swiss National Foundation Award PP002–110483, and by the EU Project NetReFound FP6-IST-034413. The material in this paper was presented in part at the IEEE International Symposium on Information Theory (ISIT), Nice, France, July 2007.

A. K. Dhulipala is with the Fraud Analytics Group, FICO, San Diego, CA 922130 USA.

A Orlitsky is with the Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093 USA (e-mail: alon@ucsd.edu).

C. Fragouli is with the School of Computer and Communication Sciences, EPFL, Lausanne, CH-1015 Switzerland (e-mail: christina.fragouli@epfl.ch).

Communicated by I. Kontoyiannis, Associate Editor for Shannon Theory.

Digital Object Identifier 10.1109/TIT.2009.2034813

TABLE I
ORGANIZATION OF RESULTS

	Pulse	Bits
Worst-case	Section III-C1	Section III-B1
Average-case	Section III-C2	Section III-B2
Non-binary inputs		
	Pulse	Bits
Worst-case	Section VI-A	NA
Average-case	Section VI-B	NA
One- vs multi-round	Section V-A	Section V-B

stricted complexity by a factor of at most the size of the input alphabet. For bit communication, the complexities differ by a factor of the logarithm of the size of the input alphabet.

We next study the one-round complexity of ternary-input functions. We provide close lower and upper bounds to the worst case one-round complexity and describe a protocol that satisfies these bounds. We also describe a protocol that achieves the average-case complexity. These results can be generalized to nonbinary inputs.

For the more general case of communication with unequal-cost bits, we analyze and provide expressions for the communication complexity of binary-input symmetric functions. We also compare the one-round complexity to the unrestricted complexity for this model.

II. COMMUNICATION MODEL AND RESULTS

In this section, we start by defining various quantities and describing the communication model in detail. We then formally state our contributions.

Consider a system of n parties $P_1, \ldots, P_n$. Let the data held by party P_i be denoted by x_i and belong to a known discrete-input alphabet $\mathcal{X}$. Let the size of the input alphabet, $|\mathcal{X}|$, be m. Denote by $x^n = x_1, \ldots, x_n$ the joint data of the n parties. The goal of the system is to compute a known function f defined over the set of all the joint data $\mathcal{X}^n$.

The parties communicate according to a predetermined protocol Q which depends only on the function to be computed. The protocol operates in stages, where at each stage a single party transmits information. At each stage, the protocol determines whether or not f can be computed using the data communicated so far. If f can be computed, the protocol outputs the value and communication ends. Otherwise, the protocol decides which party communicates next. The identity of this party depends solely on the information communicated so far as this is the only knowledge available to the protocol. In the *bit-communication* model, the party chosen by the protocol sends a bit to convey certain information. In the *pulse-communication model*, the party remains silent or emits a pulse. What is sent depends only on the information communicated so far and the data held by the chosen party.

Many protocols of varying efficiencies potentially exist for each function. The communication complexity is the amount of communication required by the most efficient protocol. Formally, let $\mathcal{Q}_b$ be the set of unrestricted bit communication protocols to compute the function f. Define $c(Q, f(x^n))$ as the communication cost required to compute $f(x^n)$ using a protocol Q and $C(Q, f)$ as the amount of communication required by the protocol Q to compute $f(x^n)$ in the worst case, i.e.,

$$C(Q, f) = \max_{x^n \in \mathcal{X}^n} c(Q, f(x^n)).$$

$C_b(f)$, the unrestricted bit communication complexity of computing a function, is formally defined as

$$C_b(f) = \inf_{Q \in \mathcal{Q}_b} C(Q, f);$$

the minimum communication cost required to evaluate the function f in the worst case.

We likewise define average-case complexity. Let p denote any distribution over length-n strings drawn from $\mathcal{X}$. The average-case complexity is defined as

$$\hat{C}_b(f) = \inf_{Q \in \mathcal{Q}_b} \hat{C}(Q, f)$$

where

$$\hat{C}(Q, f) = \sum_{x^n \in \mathcal{X}^n} p(x^n) \cdot c(Q, x^n).$$

Example 1: Let two parties P_1 and P_2 each hold r-bit strings $a_1, \ldots, a_r$, and $b_1, \ldots, b_r$, respectively. The equality function f_{eq} is 1 iff $a^r = b^r$.

Consider a simple protocol Q_1 in which P_1 communicates a^r to P_2 using r bits. P_2 then checks whether $a^r = b^r$ or not and communicates the result to P_1 using 1 bit. Thus, Q_1 needs to communicate a total of $r + 1$ bits for every possible a^r and b^r.

Consider another protocol Q_2 in which P_1 communicates a_i to P_2. P_2 then checks whether $a_i = b_i$ or not and communicates the result to P_1. If $a_i \neq b_i$ then $a^r \neq b^r$ and communication stops. Otherwise, P_1 communicates a_{i+1} next, and so on. Thus, Q_2 needs to communicate $2 \cdot r$ bits in the worst case ($a^r = b^r$). On the other hand, if a^r and b^r are chosen randomly according a uniform distribution over $\{0, 1\}^r$, then Q_2 needs to communicate only 4 bits on average.

Note that Q_1 is the most efficient protocol in the worst case while Q_2 is the most efficient one on average. ★

In the preceding example there was an implicit assumption that the cost of communicating the bit 0 is the same as that of the bit 1. In contrast, in *pulse communication*, one of the bits, say 0, has zero communication cost. To visualize a bit having zero cost, one could imagine a party staying silent instead of sending a 0 (hence zero cost). Instead of sending a 1, the party sends, what we call, a *pulse*. In pulse communication, the objective is to minimize the total number of pulses communicated. This is equivalent to minimizing the number of 1's communicated in bit communication.

Formally, let $\mathcal{Q}_p$ be the set of unrestricted pulse communication protocols to compute the function f. $C_p(f)$, the unrestricted pulse communication complexity of computing a function, is defined as

$$C_p(f) = \inf_{Q \in \mathcal{Q}_p} C(Q, f),$$

the minimum communication cost required to evaluate the function f in the worst case. Similarly, the average-case complexity is defined as

$$\hat{C}_p(f) = \inf_{Q \in \mathcal{Q}_p} \hat{C}(Q, f).$$

The following example illustrates the difference between bit communication and pulse communication.

Example 2: Let n parties $P_1, \ldots, P_n$ each hold a bit $x_1, \ldots, x_n$ respectively. Define the binary OR function f over $\{0,1\}^n$ as

$$f_{\mathrm{OR}}(x^n) = \begin{cases} 0, & x_i = 0, 1 \le i \le n \\ 1, & \text{otherwise.} \end{cases}$$

The function f_{OR} neatly contrasts pulse communication with bit communication. Indeed, for bit communication, it is easy to see that if

$$x_i = 0 \ \forall i, \quad \text{i.e., } f(x^n) = 0$$

all the parties have to transmit their data. Otherwise, a party that does not communicate could potentially hold a 1 and hence the function cannot be computed accurately. Hence, n bits are required to compute the function in the worst case. In contrast, consider the pulse communication model in which a party sends a pulse if its data is 1 and stays silent otherwise. Once a party sends a pulse, it implies that its data is 1 and hence that $f_{\mathrm{OR}}(x^n) = 1$. If $f_{\mathrm{OR}}(x^n) = 1$ a single pulse is transmitted, and if $f_{\mathrm{OR}}(x^n) = 0$ no pulses are transmitted. Thus, bit communication needs n bits while pulse communication needs just one in the worst case. ★

In most of this paper, we study the communication complexity of *symmetric functions.* A function f is said to be symmetric if

$$f(x_1, x_2, \ldots, x_n) = f(x_{\sigma_1}, x_{\sigma_2}, \ldots, x_{\sigma_n})$$

where σ is any integer permutation. In other words, the value of the function depends only on the histogram of its arguments. Thus, a symmetric function f can be defined as a function of $\mu_0(x^n), \mu_1(x^n), \ldots, \mu_{m-1}(x^n)$, where $\mu_i(x^n)$ is the number of i's in x^n. These functions are of interest in applications such as sensor networks, where the data rather than the identity of the parties is important.

Finally, for functions with nonbinary inputs, we distinguish between one-round and multiple-round communication. A one-round protocol is a protocol with the least amount of interaction, where each party can communicate in only consecutive stages. In Example 1, the protocol Q_1 is a one-round protocol as P_1 and P_2 each communicate in consecutive stages (in fact, in a single stage each). In contrast, in unrestricted multiple-round communication, the parties communicate with each other over multiple and not necessarily consecutive stages. In the protocol Q_2 described in Example 1, P_1 and P_2 each communicate in potentially multiple nonconsecutive stages. Since one-round communication is a special case of multiple-round communication, clearly one-round complexity is never less than the multiple-round complexity. (Note that Q_2 is not the most efficient multiple-round protocol for the function f_{eq}).

In terms of notation, we use the superscript 1 to specify we refer to one-round communication complexity: for example, $C_p^1(f)$ denotes the one-round pulse communication complexity of computing a function as opposed to $C_p(f)$, the unrestricted pulse communication complexity.

A. Summary of Results

We first show in Section III that the pulse complexity $C_p(f)$ of a nonconstant symmetric binary-input function is

$$C_p(f) = n + 1 - I_{\max}$$

where $I_{\max}$ is the cardinality of the largest f-constant interval[1] of f. We also show that for bit communication

$$C_b(f) = n.$$

Generalizing pulse and bit communication, we next consider communication with bits of unequal cost. If c_0 is the communication cost of the bit 0 and c_1 that of 1, the unequal-cost communication complexity $C(f)$ is given by

$$C(f) = (n + 1 - I_{\max}) \cdot c_1 + (1 - I_{\max}) \cdot c_0.$$

Finally, we derive expressions for the minimum average case complexity for symmetric functions and describe protocols that achieve it.

We consider functions of non-binary inputs in Section IV. We describe one-round communication and contrast it to the general multiple-round communication using an example.

Though unrestricted multiple-round communication, in general, is more efficient than one-round communication, we restrict ourselves to one-round communication protocols as analyzing the multiple-round complexity is challenging. In Section V, we study the relation between one-round and multiple-round complexity for symmetric functions. For bit communication we show that

$$C_b^1(f) \le \log m \cdot C_b(f)$$

and that there exist functions such that

$$C_b^1(f) > \frac{\log m}{2} \cdot \left(1 - \frac{\log m}{n}\right) \cdot C_b(f).$$

For pulse communication, we show that

$$C_p^1(f) \le m \cdot C_p(f)$$

and that there exist symmetric functions such that

$$C_p^1(f) > (m - 1) \cdot C_p(f).$$

As mentioned above, we find analyzing the multiple-round complexity of functions of nonbinary inputs difficult. In Section VI, we study the one-round complexity of ternary-input symmetric functions. We provide close lower and upper bounds to the worst case pulse complexity and describe a protocol that satisfies these bounds. We also describe a protocol that achieves

[1] f-constant intervals are defined in Section III-A.

the average-case complexity. These results can be generalized to larger alphabets.

III. Binary Inputs

We consider symmetric functions of binary inputs. We study the communication complexity of the standard bit communication and the pulse communication model. We derive explicit expressions for the worst case and average-case complexity of these models and describe protocols which achieve them.

A. f-Constant Intervals

Let $\mathcal{X} = \{0, 1\}$ and let f be a symmetric function defined over $\mathcal{X}^n$. By definition, $f(x^n)$ depends only on $\mu_1(x^n)$, the frequency of 1 (or, equivalently, only on $\mu_0(x^n)$). Thus, f can be expressed as a function of $\mu_1(x^n)$ alone. For example, the binary OR function f_{OR} can be defined as follows:

$$f_{\mathrm{OR}}(x^n) = \begin{cases} 1, & \mu_1(x^n) \geq 1 \\ 0, & \mu_1(x^n) = 0. \end{cases}$$

Define an *interval* $\mathcal{I}$ as a collection of consecutive integers. For a given function f, an interval $\mathcal{I} \subseteq \{0, 1, \ldots, n\}$ is f-constant if f is constant over all x^n such that $\mu_1(x^n) \in \mathcal{I}$, i.e.,

$$f(x^n) = f_{\mathcal{I}}, \quad \mu_1(x^n) \in \mathcal{I}$$

for some constant $f_{\mathcal{I}}$. For example, f_{OR} has two f-constant intervals

$$\begin{aligned} f_{\mathrm{OR}}(x^n) &= 0, \quad \mu_1(x^n) \in \{0\} \\ f_{\mathrm{OR}}(x^n) &= 1, \quad \mu_1(x^n) \in \{1, 2, \ldots, n\}. \end{aligned}$$

Note that a subinterval of an f-constant interval is also an f-constant interval.

Let $r \geq 1$ be the number of maximal f-constant intervals

$$f(x^n) = f_{\mathcal{I}_j}, \quad \mu_1(x^n) \in \mathcal{I}_j \tag{1}$$

where

$$\begin{aligned} \mathcal{I}_j &= (n_j, \ldots, n_{j+1} - 1), \quad 0 \leq j \leq r - 1 \\ 0 &= n_0 < n_1 < \cdots < n_{r-1} < n_r = n + 1 \end{aligned}$$

such that no two adjacent f-constant intervals can be combined to yield a larger f-constant interval, i.e.,

$$f_{\mathcal{I}_j} \neq f_{\mathcal{I}_{j+1}}, \quad 0 \leq j < r - 1.$$

B. Bit Communication

In the standard bit communication model, the parties transmit bits to each other until the function can be computed. Since the data held by each party is binary, a party can communicate its data using 1 bit. Since there are n parties at most n bits are needed to compute the function.

1) Worst Case Complexity: If the function is constant, namely, it has the same value for all its inputs, the parties can compute the function without communicating with each other. Hence, the worst case complexity, $C_b(f)$, is zero. However, to compute nonconstant functions, all the parties need to transmit their data, and hence in the worst case send a bit each, i.e., $C_b(f) = n$.

Lemma 1: For every nonconstant function f defined over $\mathcal{X}^n$

$$C_b(f) = n.$$

Proof: Suppose that the parties communicate in some order as determined by the protocol. Let the data held by the jth party to communicate be denoted by x_j. Since f is nonconstant, there exists an f-constant interval $\mathcal{I} = (n_1, \ldots, n_2 - 1)$ such that

$$f(x^n, \mu_1(x^n) = n_1 - 1) \neq f(x^n, \mu_1(x^n) = n_1). \tag{2}$$

Consider the joint data x^n such that $x_j = 1, j < n_1$ and $x_j = 0, j \geq n_1$. Note that $\mu_1(x^n) = n_1 - 1$. For such data all the parties have to communicate to compute $f(x^n)$. Suppose m parties communicate their data. If $m < n_1 - 1$ one cannot know whether $\mu_1(x^n) < n_1 - 1$ or not. It follows from (2) that the value of the function cannot be computed. Suppose $n_1 - 1 \leq m < n$. Since $\mu_1(x^m) = n_1 - 1$, the data held by at least one of the parties that did not communicate could potentially be 1 and hence $\mu_1(x^n)$ could potentially be n_1. Thus, it again follows from (2) that f cannot be computed. Therefore, all the parties need to communicate their respective data to be able to compute the function. Hence

$$C_b(f) = n. \qquad \square$$

2) Average-Case Complexity: Assume that the data held by each party is randomly distributed according to a Bernoulli distribution such that p and $\overline{p} = 1 - p$ denote the probability the data held by a party is 1 and 0, respectively. We assume that the data held by each party is independent of the data held by the rest of the parties.

Since each party holds a binary number, communicating a single bit is sufficient to indicate the value of its data. As both bits 0 and 1 have the same communication cost, a party could send either bit, say 0, to communicate that its data is $x = 0$ and the other bit, 1, to communicate that $x = 1$. The parties stop communicating once the function can be computed. We call a protocol nonredundant if according to the protocol no party communicates more than once to convey the same information. It is easy to see that any nonredundant protocol that terminates once the function is computed has the same average-case complexity as any other such protocol.

Lemma 2: Any nonredundant protocol that terminates once the function is computed has the same average-case complexity given by

$$\begin{aligned} \hat{C}_b(f) = \sum_{i=0}^{n} & \binom{i + n - n_{\iota_i + 1}}{n_{\iota_i} - 1} \cdot p^{n_{\iota_i}} \cdot \overline{p}^{i + n - n_{\iota_i + 1} - n_{\iota_i} + 1} \\ & \cdot (i + n - n_{\iota_i + 1} + 1) \\ & + \binom{i + n - n_{\iota_i + 1}}{i} \cdot p^{i} \cdot \overline{p}^{n - n_{\iota_i + 1} - 1} \\ & \cdot (i + n - n_{\iota_i + 1} + 1). \end{aligned}$$

One such protocol is as follows.

Protocol: Suppose that the parties communicate in some predetermined order. Suppose that $i - 1$ parties have already communicated and let P_i be the ith party to communicate. If the function can be computed with information transmitted so far then the protocol outputs the value of the function and communication stops. Otherwise P_i

- sends a 0 if $x_i = 0$
- sends a 1 if $x_i = 1$.

Proof: Let the r f-constant intervals of the function, as defined in (1), be $\mathcal{I}_0, \mathcal{I}_1, \ldots, \mathcal{I}_{r-1}$ such that $\mathcal{I}_j = \{n_j, \ldots, n_{j+1} - 1\}$. Let $\imath_i$ denote the argument of the f-constant interval to which i belongs, i.e.,

$$\imath_i = j, \quad \text{if } i \in \mathcal{I}_j = \{n_j, \ldots, n_{j+1} - 1\}. \tag{3}$$

We first compute the complexity corresponding to individual f-constant intervals and then proceed as follows:

$$\begin{aligned}\hat{C}_b(f) &= \sum_{x^n \in \mathcal{X}^n} p(x^n) c(Q, f(x^n)) \\ &= \sum_j \sum_{x^n \in \mathcal{X}^n,\ \mu_1(x^n) \in \mathcal{I}_j} p(x^n) c(Q, f(x^n)) \\ &\stackrel{\text{def}}{=} \sum_j c(Q, f_{\mathcal{I}_j})\end{aligned}$$

where Q is the optimal protocol. Suppose that $\mu_1(x^n) \in \mathcal{I}_j = \{n_j, \ldots, n_{j+1} - 1\}$ and that a total of i parties communicate to compute $f(x^n)$. In other words, knowing x^i is enough to compute $f(x^n)$ but knowing x^{i-1} is not. Note that $i \geq n_j + n - n_{j+1} + 1$ for one to be sure that $\mu_1(x^i) \geq n_j$ and $\mu_1(x^i) \leq n_{j+1} - 1 \equiv \mu_0(x^i) \geq n - n_{j+1} + 1$. Therefore, one of the following two conditions must be satisfied. Either

$$x_i = 1, \quad \mu_1(x^{i-1}) = n_j - 1 \tag{4}$$

or

$$x_i = 0, \quad \mu_0(x^{i-1}) = n - n_{j+1}. \tag{5}$$

There are $\binom{i-1}{n_j - 1}$ n-tuples of data each with probability $p^{n_j} \cdot \overline{p}^{i-n_j}$ which satisfy (4). Similarly, there are $\binom{i-1}{n-n_{j+1}}$ n-tuples of data each with probability $p^{i-n+n_{j+1}-1} \cdot \overline{p}^{n-n_{j+1}+1}$ which satisfy (5). The parties send one bit each, a total of i bits to communicate with each other. Therefore

$$\begin{aligned}c(Q, f_{\mathcal{I}_j}) = &\sum_{i=n-(n_{j+1}-1-n_j)}^{n} \binom{i-1}{n_j - 1} \cdot p^{n_j} \cdot \overline{p}^{i-n_j} \cdot i \\ &+ \sum_{k=n-(n_{j+1}-1-n_j)}^{n} \binom{k-1}{n - n_{j+1}} \cdot p^{k-n+n_{j+1}-1} \\ &\cdot \overline{p}^{n-n_{j+1}+1} \cdot k.\end{aligned}$$

Rewriting the arguments of summation

$$\begin{aligned}c(Q, f_{\mathcal{I}_j}) = &\sum_{i=n_j}^{n_{j+1}-1} \binom{i+n-n_{j+1}}{n_j - 1} \cdot p^{n_j} \\ &\cdot \overline{p}^{i+n-n_{j+1}-n_j+1} \cdot (i + n - n_{j+1} + 1) \\ &+ \sum_{k=n_j}^{n_{j+1}-1} \binom{k+n-n_{j+1}}{n - n_{j+1}} \cdot p^k \\ &\cdot \overline{p}^{n-n_{j+1}-1} \cdot (k + n - n_{j+1} + 1) \\ = &\sum_{i=n_j}^{n_{j+1}-1} \binom{i+n-n_{j+1}}{n_j - 1} \cdot p^{n_j} \\ &\cdot \overline{p}^{i+n-n_{j+1}-n_j+1} \cdot (i + n - n_{j+1} + 1) \\ &+ \binom{i+n-n_{j+1}}{i} \cdot p^i \cdot \overline{p}^{n-n_{j+1}-1} \\ &\cdot (i + n - n_{j+1} + 1).\end{aligned}$$

Summing up over all the f-constant intervals

$$\begin{aligned}\hat{C}_b(f) &= \sum_j c(Q, f_{\mathcal{I}_j}) \\ &= \sum_j \sum_{i=n_j}^{n_{j+1}-1} \binom{i+n-n_{j+1}}{n_j - 1} \cdot p^{n_j} \\ &\quad \cdot \overline{p}^{i+n-n_{j+1}-n_j+1} \cdot (i + n - n_{j+1} + 1) \\ &\quad + \binom{i+n-n_{j+1}}{i} \cdot p^i \cdot \overline{p}^{n-n_{j+1}-1} \\ &\quad \cdot (i + n - n_{j+1} + 1) \\ &= \sum_{i=0}^{n} \binom{i+n-n_{\imath_i+1}}{n_{\imath_i} - 1} \cdot p^{n_{\imath_i}} \\ &\quad \cdot \overline{p}^{i+n-n_{\imath_i+1}-n_{\imath_i}+1} \cdot (i + n - n_{\imath_i+1} + 1) \\ &\quad + \binom{i+n-n_{\imath_i+1}}{i} \cdot p^i \cdot \overline{p}^{n-n_{\imath_i+1}-1} \\ &\quad \cdot (i + n - n_{\imath_i+1} + 1)\end{aligned}$$

where $\imath_i$ is the argument of the f-constant interval to which i belongs as defined in (3).

C. Pulse Communication

In pulse communication, a party either transmits a pulse or remains silent to communicate its data. As in the case of bit communication, a party could completely reveal the data it holds in a single stage.

1) Worst Case Complexity: Since sending a pulse entails a unit cost and staying silent does not, each party makes judicious use of a pulse by sending a pulse when the data it holds is likely to help the most to determine the value of function and stay silent if it holds less helpful data.

For example, as seen for the function f_{OR} in Example 2, knowing that a party holds a 1 determines the value of the function immediately while knowing a party holds a 0 does not help determine the value of the function unless each of the parties also holds a 0. Therefore, to compute $f_{\text{OR}}(x^n)$ a party sends a pulse if its data is $x = 1$ and stays silent otherwise.

The worst case complexity for symmetric functions is given by the following theorem.

Theorem 1: Let f be a symmetric function defined over $\mathcal{X}^n$ with f-constant intervals $\mathcal{I}_0, \mathcal{I}_2, \ldots, \mathcal{I}_{r-1}$ as defined in (1). The worst case pulse complexity of f is

$$C_p(f) = n + 1 - I_{\max} \tag{6}$$

where $I_{\max} = \max_j |\mathcal{I}_j|$, is the cardinality of the largest f-constant interval.

Proof: To prove the theorem we use a pictorial representation of f. Let the horizontal y-axis represent $\mu_1(x^n)$ and let the vertical z-axis represent $\mu_0(x^n)$. Since f is symmetric $f(x^n)$ depends only on $\mu_0(x^n)$ and $\mu_1(x^n)$. Note that $\mu_0(x^n) + \mu_1(x^n) = n$. The $n + 1$ possible values of the function corresponding to each possible pair $(\mu_0(x^n), \mu_1(x^n))$

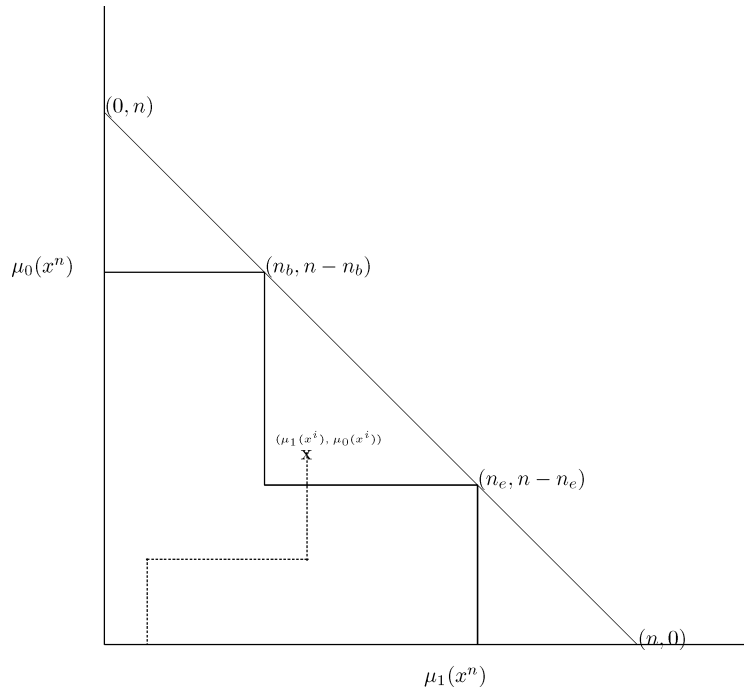

Fig. 1. Determining the value of the binary-input symmetric function with three constant intervals.

are indicated on the line $y + z = n$. Each f-constant interval $\mathcal{I}_j = (n_j, \ldots, n_{j+1} - 1)$ corresponds to an isosceles triangle given by the intersection of $y + z \leq n$, $z \geq n_j$, and $z \leq n_{j+1} - 1$. We first describe a protocol that requires at most $n + 1 - \max_j |I_j|$ pulses to compute f, thus showing that (6) can be achieved. Suppose the parties communicate according to a given protocol, which we will describe shortly. Let the data held by the jth party to communicate be denoted by $x_j \in \mathcal{X}$. Suppose i parties have already communicated. Let $\mu_1(x^i)$ and $\mu_0(x^i)$, respectively, denote the number of 1's and 0's communicated by these i parties. This corresponds to the point with coordinates $(\mu_1(x^i), \mu_0(x^i))$ in the Fig. 1. By definition, we have $\mu_0(x^i) + \mu_1(x^i) = i$.

Note that whenever a party communicates that the data it holds is 1, the corresponding point shifts by one unit along the y axis. Similarly, when a party communicates that the data it holds is 0, the corresponding point moves along the x axis by one unit. Therefore, if a point $(\mu_1(x^i), \mu_0(x^i))$ lies within an isosceles triangle corresponding to some f-constant interval $\mathcal{I}$, then the eventual point $(\mu_1(x^n), \mu_0(x^n))$ will also correspond to the same interval, *i.e.,* $\mu_1(x^n) \in \mathcal{I}$.

Thus, to uniquely determine the value of the function, the parties do not need to know the exact values of $(\mu_0(x^n), \mu_1(x^n))$. It is sufficient to know the identity of the f-constant interval to whose triangle the data corresponds. The following protocol minimizes the amount of communication required to achieve this.

Protocol: Let $\mathcal{I}_{\max} = (n_b, \ldots, n_e)$ be the longest f-constant interval such that $I_{\max} = |\mathcal{I}_{\max}| = n_e - n_b + 1$. The triangle corresponding to this interval is given by the intersection of $y + z = n$, $y \geq n_b$, and $y \leq n_e$ ($\equiv z \geq n - n_e$). The base (the lower left vertex) of the triangle is given by $(n_b, n - n_e)$. P_i, the ith party to communicate, sends a pulse if

- $x_i = 1$ until
  — $i = n$, i.e., all the parties communicate, or
  — $i < n$ but $\mu_1(x^i) \leq n_b$
- $x_i = 0$ and $\mu_1(x^i) \geq n_b$ until
  — $i = n$,
  — $i < n$ but $\mu_0(x^i) \leq n - n_e$

Communication ends when either

- $i = n$ or
- $\mu_1(x^i) \geq n_b$ and $\mu_0(x^i) \geq n - n_e$.

If $\mu_0(x^i) \geq n - n_e$ and $\mu_1(x^i) \geq n_b$ then the point $(\mu_1(x^i), \mu_0(x^i))$ belongs to the triangle that corresponds to $\mathcal{I}$. Therefore, $f(x^n)$ can be computed. In this scenario, the parties together send a total of at most $n_b + n - n_e = n + 1 - I_{\max}$ pulses. If $i = n$ when communication stops, x^n is exactly known and hence $f(x^n)$ can be computed. Again, at most $n_b + n - n_e$ pulses are communicated by the parties. Therefore, the above described protocol achieves (6).

To show that no protocol can do better than (6), consider the joint data x^n such that $\mu_1(x^n) \in \mathcal{I}_{\max}$, the largest f-constant interval. For such data, to compute $f(x^n)$, one needs to know the value of at least $n + 1 - I_{\max}$ parties. Suppose the values of at most $n - I_{\max}$ parties are known. Since no f-constant interval is longer than $I_{\max}$, the data held by the parties that did not communicate, which are at least $I_{\max}$ in number, could be such that $\mu_1(x^n)$ could potentially belong to different f-constant intervals. Hence, the protocol would not be able to learn which f-constant interval $\mu_1(x^n)$ belongs to and therefore the function cannot be computed. Thus, any protocol would require at least $n + 1 - I_{\max}$ parties to communicate their data. This would require at least $n + 1 - I_{\max}$ pulses in the worst case.

We conclude that the worst case complexity of symmetric functions of binary inputs is given by

$$C_p(f) = n + 1 - I_{\max}. \qquad \square$$

Example 3: For $0 \le j \le n+1$, the greater than or equal to j function, defined as

$$f_{\ge j}(x^n) = \begin{cases} 1, & \mu_1(x^n) \ge j \\ 0, & \mu_1(x^n) < j \end{cases}$$

is 1 iff the data held by at least j of the n parties is 1. For example, $f_{\ge 0}$ is the constant-1 function, $f_{\ge 1}$ is the binary OR function, $f_{\ge n}$ is the binary AND function, and $f_{\ge n+1}$ is the constant-0 function.

For $j = 0$ or $n+1$, $f_{\ge j}$ has a single f-constant interval $\{0, 1, \ldots, n\}$. Hence, $I_{\max} = n+1$ and $C^{pulse}(f_{\ge j}) = 0$. As expected, no communication is required as these values of j correspond to constant functions.

For $1 \le j \le n$, $f_{\ge j}$ has two f-constant intervals $\{0, 1, \ldots, j-1\}$ and $\{j, \ldots, n\}$. Hence, $I_{\max} = \max(j, n+1-j)$ and $C^{pulse}(f_{\ge j}) = \min(j, n+1-j)$. To achieve this complexity, the parties could communicate according to a protocol which proceeds as follows. Suppose $j \le (n+1)/2$. The parties, in some predetermined order, communicate a pulse to indicate that the data they hold is 1. Communication ends once the function can be computed. If $\mu_1(x^n) \ge j$, j parties communicated a pulse to indicate that they each hold a 1 and hence that $f_{\ge j} = 1$. Otherwise, only $\mu_1(x^n) < j$ parties sent a pulse each to indicate a 1 and hence that $f_{\ge j} = 0$. Thus, the communication complexity of the function is j pulses. If $j > (n+1)/2$, the parties could communicate according to a similar protocol by sending a pulse to indicate a 0 instead of 1. ★

2) Average-Case Complexity: Again, let the data held by each party be randomly distributed according to the Bernoulli distribution, where p and $\overline{p} = 1 - p$ denote the probability that the data held by a party is 1 and 0, respectively. We assume that the data of each party is distributed identical to and independent of the data of the rest of the parties. Let the r f-constant intervals of the function, as defined in (1), be $\mathcal{I}_0, \mathcal{I}_1, \ldots, \mathcal{I}_{r-1}$ such that $\mathcal{I}_j = \{n_j, \ldots, n_{j+1} - 1\}$.

Theorem 2: If $p \le 1/2$ the average-case pulse complexity of f is given by

$$\hat{C}_p(f) = \sum_{i=0}^{n} \binom{i+n-n_{\imath_i+1}}{n_{\imath_i}-1} \cdot p^{n_{\imath_i}} \cdot \overline{p}^{\,i+n-n_{\imath_i+1}-n_{\imath_i}+1} \cdot n_{\imath_i} + \binom{i+n-n_{\imath_i+1}}{i} \cdot p^{i} \cdot \overline{p}^{\,n-n_{\imath_i+1}-1} \cdot i$$

where $\imath_i$ is the argument of the f-constant interval to which i belongs to, as defined in (3). Moreover, the following protocol achieves the lowest average-case complexity.

Protocol: Let $p \le 1/2$. Suppose that $i-1$ parties have already communicated and let P_i, the ith party to communicate, hold the data x_i. The optimal communication strategy is as follows:

- if the function can be computed with information transmitted so far then the protocol outputs the value of the function and communication stops;
- else
  — if $x_i = 1$ P_i sends a pulse or
  — if $x_i = 0$ P_i stays silent.

If $p > 1/2$, the optimal protocol is identical to the above protocol except that P_i now sends a pulse if $x_i = 0$ and stays silent otherwise.

Proof: We first show that the above protocol is optimal using proof by contradiction, and then provide the expression for the complexity. Consider an optimal protocol Q according to which the parties transmit in some order. Suppose that $i-1$ parties have already communicated. Let x^{i-1} denote the data communicated by these parties. Suppose that P_i, the ith party to communicate sends a pulse if $x_i = 0$ and stays silent if $x_i = 1$, in opposition to the protocol in the above theorem. We show that by inverting the communication strategy for P_i at this stage, i.e., by having P_i send a pulse if $x_i = 1$ and stay silent if $x_i = 0$, and leaving the rest of the protocol as it is, one can obtain a new protocol $\acute{Q}$ with lower communication complexity thus showing that Q is not optimal.

Let $S(x^{i-1}) \subseteq \mathcal{X}^n$ denote the set of joint data such that x^{i-1} are the first $i-1$ bits of the n-bit string. Clearly, $|S(x^{i-1})| = 2^{n+1-i}$. The average-case communication complexity of the protocol Q can be expressed as

$$\begin{aligned} C_p(Q, f) &= \sum_{x^n \in \mathcal{X}^n} p(x^n) c(Q, f(x^n)) \\ &= \sum_{x^n \in S(x^{i-1})} p(x^n) c(Q, f(x^n)) \\ &\quad + \sum_{x^n \notin S(x^{i-1})} p(x^n) c(Q, f(x^n)). \end{aligned}$$

Consider a modified protocol $\acute{Q}$ which differs from Q only when the data held by the first $i-1$ parties is x^{i-1}. In this case, the ith party now sends a pulse if $x_i = 1$ and remains silent if $x_i = 0$. Hence

$$\begin{aligned} C_p(Q, f) - C_p(\acute{Q}, f) &= \sum_{x^n \in S(x^{i-1})} p(x^n) c(Q, f(x^n)) \\ &\quad - \sum_{x^n \in S(x^{i-1})} p(x^n) c(\acute{Q}, f(x^n)). \end{aligned}$$

Note that $c(Q, f(x^n)) = c(\acute{Q}, f(x^n)) + 1$ if $x_i = 0$ and $c(Q, f(x^n)) = c(\acute{Q}, f(x^n)) - 1$ if $x_i = 1$. Therefore

$$\begin{aligned} C - p(Q, f) - C_p(\acute{Q}, f) &= \sum_{x^n \in S(x^{i-1}), x_i = 1} p(x^n)(c(Q, f(x^n)) - c(\acute{Q}, f(x^n))) \\ &\quad + \sum_{x^n \in S(x^{i-1}), x_i = 0} p(x^n)(c(Q, f(x^n)) - c(\acute{Q}, f(x^n))) \\ &= \sum_{x^n \in S(x^{i-1}), x_i = 1} p(x^n)(-1) + \sum_{x^n \in S(x^{i-1}), x_i = 0} p(x^n)(1) \\ &= p(x^{i-1}0) - p(x^{i-1}1) \\ &= p(x^{i-1})(\overline{p} - p) \\ &\ge 0. \end{aligned}$$

We conclude that $\acute{Q}$ has a lower communication complexity than Q, and thus Q is not optimal. As a result, the assumption that in an optimal protocol, a party sends a pulse to communicate 1 (or the more probable input in general) is false. Hence, under an optimal protocol, a party would send a pulse if the data it holds is the less probable input and stay silent otherwise.

The derivation of the expression for the average-case pulse complexity follows along the same lines as that in the case of bit communication, the only difference being that we count the average number of pulses (or equivalently the number of 1's if $p \leq 1/2$) communicated instead of the average number of bits communicated. □

Example 4: Consider the binary OR function f_{OR}. Suppose that the probability of observing 1 is less than half, i.e., $p \leq 1/2$. Applying Theorem 2, under the optimal protocol, the ith party P_i sends a pulse if $x_i = 1$ and remains silent otherwise. Communication ends once the function can be computed. The first party that sends a pulse and indicates that the data it holds is a 1 determines the value of the function since $f_{\mathrm{OR}} = 1$ iff $\mu_1(x^n) \geq 1$. Thus, to compute the function, a single pulse is required when $\mu_1(x^n) \geq 1$ and no pulses are required when $\mu_1(x^n) = 0$. This implies that

$$\begin{aligned}\hat{C}_p(f_{\mathrm{OR}}) &= p(\mu_1(x^n) \geq 1) \cdot 1 \\ &= 1 - p(\mu_1(x^n) = 0) = 1 - \overline{p}^n.\end{aligned}$$

Consider f_{OR} again, and now suppose that $p > 1/2$. Applying Theorem 2, the party P_i sends a pulse if $x_i = 0$ and remains silent if $x_i = 1$. The first party that stays silent and indicates that the data it holds is 0 also determines the value of the function. Thus, if P_j is the first party that holds a 1 (and stays silent), $j - 1$ pulses, one for each of the preceding parties, are needed to compute the function. Therefore

$$\begin{aligned}\hat{C}_p(f_{\mathrm{OR}}) &= \sum_{j=0}^{n-1} p \cdot \overline{p}^j \cdot j + \overline{p}^n \cdot n \\ &= \sum_{j=0}^{n-1} p \cdot \overline{p}^j \cdot (j+1) - \sum_{j=0}^{n-1} p \cdot \overline{p}^j + \overline{p}^n \cdot n \\ &= \sum_{j=0}^{\infty} p \cdot \overline{p}^j \cdot (j+1) \\ &\quad - \sum_{j=n}^{\infty} p \cdot \overline{p}^j \cdot (j+1) - (1 - \overline{p}^n) + \overline{p}^n \cdot n.\end{aligned}$$

Using the fact that $\sum_{j=0}^{\infty} p \cdot \overline{p}^j \cdot (j+1)$, the expected number of coin tosses to get a "head" (or a 1) is $\frac{1}{1-\overline{p}}$

$$\begin{aligned}\hat{C}_p(f_{\mathrm{OR}}) &= \frac{1}{1-\overline{p}} - \sum_{i=0}^{\infty} p \cdot \overline{p}^{i+n} \cdot (i+n+1) \\ &\quad - (1 - \overline{p}^n) + \overline{p}^n \cdot n \\ &= \frac{1}{1-\overline{p}} - \overline{p}^n \left(\frac{1}{1-\overline{p}} + n \right) - 1 + \overline{p}^n + \overline{p}^n \cdot n \\ &= (1 - \overline{p}^n) \frac{\overline{p}}{1-\overline{p}}.\end{aligned}$$

It is interesting to note that when $p > 1/2$ to minimize the average-case complexity it is optimal to send a pulse to communicate a 0 in spite of potentially having to send up to n pulses (when $\mu_1(x^n) = 0$). Recall that the worst case complexity, as noted in the previous section, is just 1 pulse, which is achieved by sending a pulse to communicate a 1. ★

D. Unequal-Cost Bit Communication

Suppose that each party can communicate the data it holds using one of two bits 0 and 1 with respective communication costs c_0 and c_1. If $c_0 = c_1$ this is equivalent to binary communication. If $c_0 = 0$, it corresponds to pulse communication. In the rest of the section we assume that $0 < c_0 \leq c_1$.

1) Worst Case Complexity:

Theorem 3: Let f be a symmetric function defined over $\mathcal{X}^n$ with f-constant intervals $\mathcal{I}_0, \mathcal{I}_2, \ldots, \mathcal{I}_{r-1}$ as defined in (1). The worst case pulse complexity of f is

$$C = (n + 1 - I_{\max}) \cdot c_1 + (I_{\max} - 1) \cdot c_0 \tag{7}$$

where $I_{\max} = \max_j |\mathcal{I}_j|$ is the cardinality of the largest f-constant interval.

Proof: The proof, which is similar to that of Thorem 1 is provided in Appendix A. Note that Theorem 1 is a special case of Theorem 3.

2) Average-Case Complexity: Suppose that the data held by each party is randomly distributed according to the Bernoulli distribution, independent of and identical to that of the data held by the rest of the parties. Let p and $\overline{p} = 1 - p$ denote probabilities that the data held by a party is 1 and 0, respectively. Let also the r f-constant intervals of the function, as defined in (1), be $\mathcal{I}_0, \mathcal{I}_1, \ldots, \mathcal{I}_{r-1}$ such that $\mathcal{I}_j = \{n_j, \ldots, n_{j+1} - 1\}$.

Lemma 3: If $p \leq 1/2$ the average-case pulse complexity of f is given by

$$\begin{aligned}\hat{C}(f) = \sum_{i=0}^{n} &\binom{i + n - n_{\iota_i+1}}{i} \cdot p^i \cdot \overline{p}^{n - n_{\iota_i+1} - 1} \\ &\cdot (i \cdot c_1 + .(n - n_{\iota_i+1} - 1) \cdot c_0) \\ &+ \binom{i + n - n_{\iota_i+1}}{n_{\iota_i} - 1} \cdot p^{n_{\iota_i}} \cdot \overline{p}^{i+n-n_{\iota_i+1}-n_{\iota_i}+1} \\ &fi \cdot (n_{\iota_i} \cdot c_1 + (i + n - n_{\iota_i+1} - n_{\iota_i} + 1) \cdot c_0).\end{aligned}$$

This complexity is achieved by the following protocol.

Protocol: Suppose that the parties communicate in some predetermined order. Suppose that $i - 1$ parties have already communicated and let P_i be the ith party to communicate:

- if the function can be computed with the information transmitted so far then the protocol outputs the value of the function and communication stops;
- else
  — if $x_i = 1$ P_i sends 1
  — if $x_i = 0$ P_i sends 0.

If $p > 1/2$, the optimal protocol is identical to the above protocol except that P_i now sends a pulse if $x_i = 0$ and stays silent otherwise.

Proof: The proof is almost identical to that in the case of pulse communication and is omitted.

IV. Nonbinary Inputs

In this section, we consider functions of nonbinary inputs and focus on pulse communication. We introduce one-round communication and compare it to multiple-round communication. We describe some properties of optimal protocols and lay the groundwork for the subsequent sections.

A. Unary Questions

Let the data held by each party be one of m values, i.e., let $\mathcal{X} = \{0, 1, \ldots, m-1\}$. The n parties communicate in some particular order as determined by a protocol. When a party communicates, it can either send a pulse or stay silent, thus differentiating between two distinct events. A protocol can be viewed as asking a series of "yes/no" questions such that a party sends a pulse if the answer to a question is, for example, "yes" and stays silent otherwise. Let a party P hold the data $x \in \mathcal{X}$. Suppose that a protocol asks the question "Does $x \in S \subset \mathcal{X}$." The following lemma shows that there exists an optimal protocol such that all the questions are unary, i.e., $|S| = 1$.

Lemma 4: There exists an optimal protocol where all the questions are unary.

Proof: Suppose an optimal protocol asks the question "Does $x \in S \subset \mathcal{X}$" where $S = \{j, k\}$ for some $j, k \in \mathcal{X}$, i.e., it costs one pulse to know that $x \in \{j, k\}$. Consider a different communication strategy such that P sends a pulse if $x = j$, stays silent in the current stage and sends a pulse in a later stage if $x = k$, and stays silent in both stages if $x \notin \{j, k\}$. Thus, the party sends one pulse to reveal that $x = j$ or that $x = k$ instead of sending one pulse to reveal only that $x \in \{j, k\}$. Therefore, a party could communicate more information by sending a pulse to reveal the exact value of the data, instead of revealing that the data could be one of various different values. It follows that optimal pulse communication can be viewed as asking a series of questions such as "Is $x = j$." □

Note that, as in the case of binary inputs, even for nonbinary inputs a single pulse is sufficient to communicate the value of the data held by a party. Indeed, as shown in Fig. 2, a party that has a value x can use $m-1$ stages to specify what this value is: in the ith stage the party sends a pulse if $x = i$ and stays silent otherwise. Thus, the party spends one pulse if $x \in \{1, 2, \ldots, m-1\}$ and stays silent in all the $m-1$ stages if $x = 0$. Note that, in the worst case, a party needs to use $m-1$ stages to communicate its value, incurring an increased delay as compared to the case of binary communication, where $\log m$ stages would be sufficient.

Example 5: Consider the maximum function defined as $f_{\max}(x^n) = \max_i x_i$. Since the function is symmetric it can be described as follows:

$$f_{\max}(x_1^n) = i, \quad \text{if } \mu_i(x^n) > 0 \text{ and } \mu_j(x^n) = 0, \forall j > i.$$

Since $f_{\max}$ is not a constant function, $C_p(f_{\max}) \geq 1$. It is easy to see that this can be achieved with an equality using the following protocol.

Protocol: Suppose the parties communicate in some order. The ith party sends a pulse if $x_i = m-1$. If one of the parties sends a pulse then $\mu_{m-1}(x^n) \geq 1$ and hence $f_{\max}(x^n) = m-1$ and communication ends. Suppose n stages have passed and none of the parties has sent a pulse, namely $\mu_{m-1}(x^n) = 0$. The parties communicate again and the ith party sends a pulse if $x_i = m-2$. If one of the parties sends a pulse, $\mu_{m-2}(x^n) \geq 1$, $\mu_{m-1}(x^n) = 0$, hence, $f_{\max}(x^n) = m-2$ and communication ends. If none of the parties sends a pulse then the ith party now sends a pulse if $x_i = m-3$, and so on. If after $n \cdot (m-1)$ stages none of the parties sends a pulse, this implies that $\mu_j(x^n) = 0$, $\forall j > 0$ and therefore $f_{\max}(x^n) = 0$.

Note that this protocol needs at most one pulse to evaluate $f_{\max}(x^n)$. Thus, the worst case complexity of $f_{\max}$ is one. ★

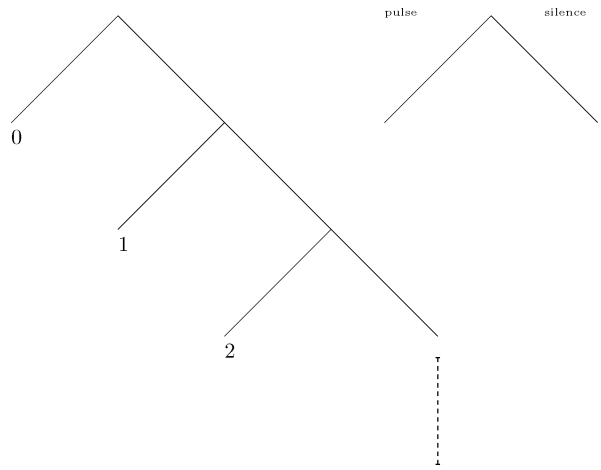

Fig. 2. Communication using a single pulse.

B. One-Round Communication

In general, and as seen in the case of $f_{\max}$ in Example 5, the optimal communication protocol that minimizes the total amount of communication may involve the parties interacting with each other by communicating in multiple (but not necessarily consecutive) stages. This unrestricted multiple-round communication might be challenging to implement in practice. In the following, we focus our attention on one-round communication.

In one-round communication a party can communicate only in consecutive stages and cannot alternate with another party. Suppose the party P_1 communicates in a certain stage. In the next stage P_1 could communicate again. But if a different party, say P_2, communicates instead, then P_1 cannot communicate again in the future. Recall that we use $C_p^1(f)$ and $C_p(f)$ to respectively denote the one-round and the unrestricted multiple-round pulse complexities of f.

Example 6: Consider the function $f_{\max}$ again. The one-round communication complexity of $f_{\max}$ is

$$C_p^1(f_{\max}) = m - 1.$$

For the sake of simplicity we prove this result for ternary functions, i.e., $m = 3$. The proof can be generalized to any m. Suppose that the parties communicate in some order and that each party communicates in up to two consecutive stages. Consider x^n such that $x_1 \neq 2$. As mentioned earlier, a party sends a pulse for two of the inputs and stays silent for one. Therefore, in

the worst case, the first party sends a pulse to communicate its data. Clearly, the function cannot be computed with the knowledge of x_1 alone and thus at least one of the subsequent parties sends a pulse in the worst case. Therefore $C_p^1(f_{\max}) \geq 2$. The following protocol shows that this can be achieved with equality.

Protocol: If none of the previous $i-1$ parties held either 2 or 1, P_i communicates x_i as follows:

- sends a pulse in the first stage if $x_i = 1$;
- stays silent in the first stage and sends a pulse in the second stage if $x = 2$;
- stays silent in both stages if $x = 0$.

If one of the previous parties held a 2, the function can be computed ($f_{\max} = 2$). If none of them held a 2 but one of them held a 1, P_i now communicates whether $x \stackrel{?}{=} 2$ as follows:

- sends a pulse if $x_i = 2$;
- stays silent otherwise.

If x^n is such that the first 1 appears before the first 2 appears, two pulses are required to evaluate $f_{\max}$. If x^n is such that the first 2 appears before the first 1 appears, one pulse is required and if x^n has no 1's or 2's, no pulses are required to evaluate $f_{\max}$. Thus, the complexity of this protocol is 2. Hence, $C_p^1(f_{\max}) = 2$. In Section VI, we prove this result again using different techniques. ★

In the rest of the paper, we restrict ourselves to one-round communication. We first study how restricting communication to one-round effects the communication complexity in Section V. In Section VI, we study the one-round complexity of ternary-input functions.

V. One-Round Versus Multiple-Round

In this section, we compare the one-round and multiple-round complexity of bit and pulse communication.

A. Pulse Communication

We first consider pulse communication. We derive an upper bound on the ratio of one-round and multiple-round complexity and describe a function which approaches this bound.

Theorem 4: For all $m \geq 2$

$$(m-1) \cdot C_p(f) \stackrel{\exists}{\leq} C_p^1(f) \stackrel{\forall}{\leq} m \cdot C_p(f) \tag{8}$$

where the upper bound holds for all symmetric functions and the lower bound for some symmetric functions.

Proof: We first prove the upper bound by showing that given any symmetric function f of multiple-round complexity $C_p(f)$, one could derive a one-round protocol Q^1 such that $C(Q^1, f) \leq m \cdot C_p(f)$.

In one-round communication, a party communicates in a single round (over potentially multiple but consecutive stages). To know the exact value of the data held by a party using a single pulse a protocol could ask the following successive questions:

$$x_1 = 0? \stackrel{no}{\rightarrow} x_1 = 1? \stackrel{\text{no}}{\rightarrow} \ldots x_1 = m-2'? \stackrel{\text{no}}{\rightarrow} x_1 = m-1?$$

such that a party sends a pulse when it answers "yes" to a question. Note that the last question is redundant since if $x \notin \{0, 1, \ldots, m-2\}$ then $x = m-1$. But we include the question for the sake of simplicity. The one-round protocol Q^1 asks such questions until it learns whether at least $C_p(f)$ each of 0's, 1's, $\ldots, m-1$'s exist or not. If at some stage the protocol learns that there are at least $C_p(f)$ of say, 0's, then in the future the protocol will not ask the question "Is $x = 0$?"

When Q^1 terminates, for any given $j \in \mathcal{X}$, it learns either that $\mu_j(x^n) \geq C_p(f)$ or the exact value $\mu_j(x^n) < C_p(f)$. Define $L(x^n) \subseteq \mathcal{X}$ such that

$$\mu_j(x^n) \begin{cases} < C_p(f), & j \in L(x^n) \\ \geq C_p(f), & j \notin L(x^n). \end{cases}$$

Q^1 spends $\mu_j(x^n) < C_p(f)$ pulses for each $j \in L(x^n)$ and $C_p(f)$ pulses for each $j \notin L(x^n)$, a total of at most $m \cdot C_p(f)$ pulses, to gather information about $\mu_j(x^n)$ for each j. We show that the information gathered by Q^1 is sufficient to answer all the questions asked by any optimal multiple-round protocol. Consider any optimal multiple-round protocol Q, i.e., $C(Q, f) = C_p(f)$. As described in Section IV-A, Q can be viewed as a series of "yes/no" questions of the form "Is $x = j$" for some $j \in \mathcal{X}$. A party sends a pulse if the answer is "yes" and stays silent otherwise. Since $C(Q, f) = C_p(f)$, at most $C_p(f)$ questions are answered "yes." In particular, for any given $j \in \mathcal{X}$ if $C_p(f)$ nodes answer "yes" to the question whether they hold a j, then the protocol must terminate.

We next show that the information gathered by Q_1 is sufficient to correctly answer all the questions asked by Q. For each $j \in L(x^n)$ the first $\mu_j(x^n)$ questions of the form "Is $x = j$" asked by Q are answered "yes" and the rest of such questions are answered "no." These are clearly the correct answers to these questions. For each $j \notin L(x^n)$ all questions of the form "Is $x = j$" asked by Q are answered "yes." It is clear from the above observation that Q poses at most $C_p(f)$ such questions and are all answered correctly.

Thus, the information gathered by Q^1 is sufficient to answer all the questions posed by Q. By definition, Q can compute $f(x^n)$ upon termination. Hence, Q^1 can also compute f. Therefore

$$C_p^1(f) \leq C(Q^1, f) \leq m \cdot C_p(f).$$

This completes the proof for the upper bound. □

The lower bound $(m-1) \cdot C_p(f) \leq C_p^1(f)$ holds trivially when $m = 2$, i.e., when the data held by the parties is binary. This is because for binary-inputs one-round communication is equivalent to multiple-round communication as any given party can communicate its data in a single stage. For $m > 2$, the following slightly stronger bound holds.

Lemma 5: For all $m > 2$, there exists a symmetric function f such that

$$(m-1) \cdot C_p(f) \stackrel{\exists}{<} C_p^1(f).$$

Proof: A function which achieves the bound is described as follows. Let r be an integer such that $\frac{m\cdot(m-1)}{2} < r < \frac{n}{m}$. Consider a function defined as follows. If $1 \leq \mu_0(x^n) \leq m-1$

$$f = \begin{cases} \mu_0(x^n)\cdot n + r, & \mu_{\mu_0(x^n)}(x^n) \geq r - \mu_0(x^n) \\ \mu_0(x^n)\cdot n + \mu_{\mu_0(x^n)}(x^n), & \text{else} \end{cases}$$

else if $\mu_0(x^n) \geq m$ or $\mu_0(x^n) = 0$

$$f = \begin{cases} \mu_0(x^n), & m \leq \mu_0(x^n) \leq r \\ r, & \text{otherwise.} \end{cases}$$

In other words, the value of the function depends on the number of parties that hold a 0 as their data, i.e., on $\mu_0(x^n)$, and on the number of parties that hold $\mu_0(x^n)$ as their data i.e., on $\mu_{\mu_0(x^n)}(x^n)$.

Multiple-round communication: Consider a multiple-round protocol Q under which the parties communicate in some order. First the parties communicate so that $\mu_0(x^n)$, the number of parties which hold 0 as their data can be determined. Once $\mu_0(x^n)$ is known, the parties, if needed, then communicate again to determine $\mu_{\mu_0(x^n)}(x^n)$. Note that if $\mu_0(x^n) \geq r$, to compute $f(x^n) = r$, the protocol needs to know if at least r parties hold 0 as their data or not which would need r pulses in the worst case. Therefore

$$C_p(f) \geq r. \tag{9}$$

The following protocol achieves this with an equality.

The protocol first learns $\mu_0(x^n)$ by having the parties send a pulse if their data is "0" and stay silent otherwise until r parties send a pulse. This involves sending $\min(r, \mu_0(x^n))$ pulses. If $1 \leq \mu_0(x^n) < m$ the protocol learns $\mu_{\mu_0(x^n)}(x^n)$ by having the parties send a pulse if their data is "$\mu_0(x^n)$" and stay silent otherwise until $r - \mu_0(x^n)$ parties send a pulse. This involves sending $\min(r - \mu_0(x^n), \mu_{\mu_0(x^n)}(x^n))$ pulses. Thus, the protocol needs r pulses in the worst case when $\mu_0(x^n) \geq r$. Therefore, $C_p(f) \leq r$.

Hence, it follows that $C_p(f) = r$.

One-round communication: We now show that an optimal one-round protocol needs more than $(m-1)\cdot r$ pulses to compute $f(x^n)$ in the worst case.

Consider x^n such that $x_i = 0$ for $n - r + 1 \leq i \leq n$ and $x_i \neq 0$ for $1 \leq i \leq n - r$. Since a one-round protocol would not be able to determine that $\mu_0(x^n) = r$ until all the n parties communicate, it is clear that for such x^n, in the worst case, for each $1 \leq i \leq m-1$ the first $r - i$ parties which have i as their data need to communicate their respective values by sending a pulse each. Further, the last r parties communicate their data using a total of r pulses in the worst case. Therefore

$$C_p^1(f) \geq \sum_{i=1}^{m-1}(r-i) + r = m\cdot r - \frac{(m-1)\cdot m}{2}.$$

Since $r > \frac{(m-1)\cdot m}{2}$, it follows that

$$(m-1)\cdot C_p(f) < C_p^1(f).$$

This completes the proof for the lower bound. □

1) Asymmetric Functions: In Theorem 4 we showed that the one-round complexity of symmetric functions is at most m times and could be as high as $m-1$ times the multiple-round complexity. The following lemma shows that for asymmetric functions the bounds are of the order of n.

Lemma 6: For asymmetric functions

$$\frac{n}{4}C_p(f) \overset{\exists}{<} C_p^1(f) \overset{\forall}{\leq} n.$$

Proof: Since each party can communicate its data using at most one pulse, the upper bound holds trivially. To prove the lower bound consider the following function. Let $\mathcal{X} = \{0,1,2\}$ and let an even number n of parties be indexed from 1 to n. For a given joint data x^n, let $j(x^n)$ be the smallest j such that the jth party holds a 0. If no such party exists $j(x^n)$ is defined to be ∞, i.e.,

$$j(x^n) \overset{\text{def}}{=} \begin{cases} \min\{j : x_j = 0\}, & \text{if } \exists j \text{ such that } x_j = 0 \\ \infty, & \text{if } \forall j\ x_j \neq 0. \end{cases}$$

Define

$$f(x^n) \overset{\text{def}}{=} \begin{cases} (\infty,\infty), & \text{if } j(x^n) = \infty \\ (j,1), & \text{if } x_{n+1-j(x^n)} = 1 \\ (j,0), & \text{if } x_{n+1-j(x^n)} \neq 1 \end{cases}$$

which indicates whether the $(n+1-j(x^n))$th party holds a 1.

It is easy to see that $C_p(f) \leq 2$. One way to achieve this is as follows. The parties communicate in the order they are indexed. The parties send a pulse if the data they hold is 0 until the first party sends a pulse. If none of the parties send a pulse then $j(x^n) = \infty$ and $f(x^n) = (\infty,\infty)$ and communication stops. If a party sends a pulse then $j(x^n)$ is the index of that party. The $(n+1-j(x^n))$th party can then send a pulse if its data is 1 and stay silent otherwise. Thus, 2 pulses are sufficient to compute f.

On the other hand, it can be shown that $C_p^1(f) \geq \frac{n}{2}$. Consider x^n such that $x_i \neq 0\ \forall i$. Clearly, $j(x^n)$ needs to be determined to compute $f(x^n)$, and $j(x^n)$, for the x^n we chose, can be determined iff all the parties communicate. For such x^n it can be shown that for each pair of the parties (P_i, P_{n+1-i}) at least one of the two parties must send a pulse.

Suppose neither of the two parties P_i, P_{n+1-i} has communicated yet. Since none of the parties holds a zero, $j(x^n)$ could potentially be either i or $n+1-i$. Between the two parties suppose P_i communicates first. It is not enough for the protocol to know whether $x_i = 0$. Since $x_i \neq 0$ and j could be $n+1-i$, and since P_i can communicate in only consecutive stages in one-round communication, the protocol also needs to know whether $x_i = 1$. This needs one pulse in the worst case. For example, after P_i stays silent in the first stage to communicate that $x_i \neq 0$, in the second stage P_i could either send a pulse if $x_i = 1$ and stay silent otherwise (or *vice versa*). Therefore, if $x_i = 1$ (or, equivalently, if $x_i = 0$) P_i sends a pulse.

If P_{n+1-i} communicates before P_i a similar argument holds. Thus, in either case, for each pair of parties (P_i, P_{n+1-i}), one of the two needs to send a pulse in the worst case. Since there are $\frac{n}{2}$ such distinct pairs, $C_p^1(f) \geq \frac{n}{2}$. This proves the lower bound of Lemma 6. □

B. Bit Communication

We next compare the one-round and multiple-round complexity for bit communication.

Theorem 5: For all $m \geq 2$, the one- and multiple-round bit complexities satisfy

$$\frac{\log m}{2} \cdot (1 - \frac{\log m}{n}) \cdot C_b(f) \overset{\exists}{<} C_b^1(f) \overset{\forall}{\leq} \log m \cdot C_b(f) \quad (10)$$

where the upper bound holds for all symmetric functions and the lower bound holds for some symmetric functions.

Proof: Please refer to Appendix B for proof.

C. Unequal-Cost Bits

We finally compare the one-round and multiple-round complexity in the case of communication with unequal-cost bits.

Let c_0 and c_1 be the communication costs of the bits 0 and 1, respectively. In one-round communication each party describes the data it holds in a single stage. We can use a prefix-free encoder $E(\mathcal{X}) : \mathcal{X} \rightarrow \{0,1\}^*$ to represent such a description, where $\{0,1\}^* = \bigcup_i \{0,1\}^i$ denotes the set of all possible binary strings. For each $x \in \mathcal{X}$, let $c^{E(\mathcal{X})}(x)$ denote the cost of describing x using E. This is equal to the sum of the costs of the individual bits in $E(x)$, the description of x. The cost of the encoder is defined as $c^{E(\mathcal{X})} = \max_x c^{E(\mathcal{X})}(x)$, the maximum cost incurred by an $x \in \mathcal{X}$ using the encoder $E(\mathcal{X})$.

Example 7: Let $\mathcal{X} = \{0,1,2\}$ and let $E(0) = 1$, $E(1) = 00$, and $E(2) = 01$. Therefore, $c^{E(\mathcal{X})}(0) = c_1$, $c^{E(\mathcal{X})}(1) = 2 \cdot c_0$, and $c^{E(\mathcal{X})}(2) = c_0 + c_1$. The cost of the encoder is $c^{E(\mathcal{X})} = c_0 + c_1$. Note that $0 < c_0 \leq c_1$. ★

$c^{\mathcal{X}}$, the cost of communicating $\mathcal{X}$ is defined as the minimum cost of communicating $x \in \mathcal{X}$ in the worst case using any prefix free encoder, i.e.,

$$c^{\mathcal{X}} = \min_E c^{E(\mathcal{X})}, \quad (11)$$

where the minimization is over all possible prefix-free binary encoders.

Theorem 6: Let $C_u(f), C_u^1(f)$ denote the multiple-round and one-round complexity of computing f using unequal-cost bits and let $c_1 \geq c_0 > 0$. The complexities satisfy the following bounds:

$$\frac{c^{\mathcal{X}} - c_1 + c_0}{c_0 + c_1} \cdot (1 - \frac{\log m}{n}) \cdot C_u(f) \overset{\exists}{<} C_u^1(f) \overset{\forall}{\leq} \frac{c^{\mathcal{X}}}{c_0} \cdot C_u(f). \quad (12)$$

Proof: The proof is along the same lines as in the case of bit communication.

VI. Ternary Inputs

In this section, we study the pulse complexity of symmetric functions of ternary inputs. We prove bounds on the one-round pulse complexity of symmetric functions and describe communication protocols achieving them. We also describe an optimal protocol to achieve the lowest average-case one-round complexity. These results can be generalized to functions of larger-alphabet inputs. We start by introducing the notion of dependency, which will be used in the bounds we will develop.

Dependency: As the parties communicate with each other to compute a function f, the system knows more and more about the data and the value of the function. After a certain stage, a party may need to communicate only partial information about its data to compute the function and the ternary-input function behaves like a binary-input function. We define *dependency* to denote the mathematical representation of such a transition.

For example, consider the ternary maximum function $f_{\max}$ defined in Example 5. The very first party to communicate has to reveal the exact value it holds. But suppose after i parties have communicated it is known that at least one of the parties holds a 1. Thus, $f_{\max}(x^n) \geq 1$. Therefore, in the future, a party only needs to communicate whether the data it holds is 2 or not. If the data is not 2, it does not matter whether it is 1 or 0. In other words, the function behaves like a binary-input function, the inputs being 2 and $\overline{2} = \{0,1\}$.

Formally, f is said to be (m_0, m_1, m_2) 2-dependent if $\forall x$ such that $\mu_0(x^n) \geq m_0$ and $\mu_1(x^n) \geq m_1$

$$f(x^n) = g_2(\mu_2(x^n))$$

where $g_2(\mu_2(x^n))$ is determined by $\min(\mu_2(x^n), m_2)$. In other words, once the parties communicate at least m_0 0's and m_1 1's, the function behaves like a binary-input function. And once the parties communicate m_2 2's too, the function behaves like a constant function and communication stops.

For example, the function $f_{\max}$ is $(0,1,1)$ 2-dependent. Note that if $\mu_1(x^n) \geq 1$

$$f_{\min}(x^n) = g_2(\mu_2(x^n))$$
$$g_2(i) = \begin{cases} 1, & i = 0 \\ 2, & i \geq 1. \end{cases}$$

We similarly define a function to be (m_0, m_1, m_2) 0-dependent if $\forall x^n$ such that $\mu_1(x^n) \geq m_1$ and $\mu_2(x^n) \geq m_2$

$$f(x^n) = g_0(\mu_0(x^n))$$

where $g_0(\mu_0(x^n))$ is determined by $\min(\mu_0(x^n), m_0)$, and define f to be (m_0, m_1, m_2) 1-dependent if $\forall x$ such that $\mu_0(x^n) \geq m_0$ and $\mu_2(x^n) \geq m_2$

$$f(x^n) = g_1(\mu_1(x^n))$$

where $g_1(\mu_1(x^n))$ is determined by $\min(\mu_1(x^n), m_1)$. For example, $f_{\max}$ is both $(0,0,1)$ 0-dependent and 1-dependent. If a function f is (m_0, m_1, m_2) 0-, 1- and 2-dependent, f is said to be (m_0, m_1, m_2) $\mathcal{X}$-dependent.

A. Worst Case Complexity: Upper Bound

Let (m_0^0, m_1^0, m_2^0) be a 3-tuple with the smallest sum $m_0^0 + m_1^0 + m_2^0$ such that f is (m_0^0, m_1^0, m_2^0) 0-dependent. Similarly, let (m_0^1, m_1^1, m_2^1), and (m_0^2, m_1^2, m_2^2) be the respective 3-tuples with the smallest sums such that f is (m_0^1, m_1^1, m_2^1) 1-dependent and (m_0^2, m_1^2, m_2^2) 2-dependent. The following theorem provides an upper bound on the worst case complexity of ternary-input symmetric functions.

Theorem 7: The worst case pulse complexity of a ternary-input symmetric function f is upper-bounded as

$$C_p^1(f) \leq \max_r m_0^r + \max_r m_1^r + \max_r m_2^r. \tag{13}$$

Proof: Please refer to Appendix C.

Example 8: Consider the ternary maximum function $f_{\max}$ defined earlier. It can be verified that $(0,1,1)$ is the smallest 3-tuple such that $f_{\max}$ is $(0,1,1)$ 2-dependent. Similarly, $(0,0,1)$ is the smallest 3-tuple such that $f_{\max}$ is $(0,0,1)$ 0- and 1-dependent, i.e.,

$$(m_0^0, m_1^0, m_2^0) = (0,0,1)$$
$$(m_0^1, m_1^1, m_2^1) = (0,0,1)$$
$$(m_0^2, m_1^2, m_2^2) = (0,1,1).$$

Applying Theorem 7

$$C_p^1(f_{\max}) \leq \max(1+1,1,1) = 2.$$

This concurs with the result shown Section IV that $C_p^1(f_{\max}) = m-1$. Note that for ternary-input functions $m = 3$. ★

For a given function f, let (m_0, m_1, m_2) be a 3-tuple with the smallest sum such that f is (m_0, m_1, m_2) 0-, 1-, and 2-dependent. Since f is also $(\max_r m_0^r, \max_r m_1^r, \max_r m_2^r)$ 0-, 1-, and 2-dependent it follows that $m_0 + m_1 + m_2 \leq \max_r m_0^r + \max_r m_1^r + \max_r m_2^r$. The following lemma gives a slightly tighter upper bound on $C_p^1(f)$.

Lemma 7: The worst case one-round pulse complexity of a function f is upper-bounded as

$$C_p^1(f) \leq m_0 + m_1 + m_2. \tag{14}$$

A protocol that achieves this bound is identical to the one that achieves the other upper bound the only difference being $\max_r m_i^r$ is replaced by m_i for $0 \leq i \leq 2$.

B. Worst Case Complexity: Lower Bound

The following theorem provides a lower bound on the complexity of ternary-input function.

Theorem 8: The worst case one-round pulse complexity of a ternary-input symmetric function f is lower-bounded as

$$C_p^1(f) \geq \max_i (m_0^i + m_1^i + m_2^i). \tag{15}$$

Proof: Please refer to Appendix D.

For the ternary maximum function, it follows from Theorems 7 and 8 that

$$2 \leq C_p^1(f_{\max}) \leq 2.$$

The following is an example for which the bounds given by the theorems are not as tight.

Example 9: Consider a function f defined as follows:

$$f(x^n) = \begin{cases} 1, & \text{if } \exists k \neq j \text{ s. t. } \mu_j(x^n) \geq r \text{ and } \mu_k(x^n) \geq r \\ 0, & \text{otherwise} \end{cases}$$

for some integer r. It can be shown that f is $(0,r,r)$ 0-dependent, $(r,0,r)$ 1-dependent, and $(r,r,0)$ 2-dependent, the 3-tuples being those with the smallest possible sum. It follows from Theorems 7 and 8 that

$$2 \cdot r \leq C_p^1(f) \leq 3 \cdot r.$$

It can be shown that for the above function $C_p^1(f) = 3 \cdot r - 1$. ★

C. Average-Case Complexity

Suppose that the data held by each party is randomly distributed according to some probability distribution, independent of and identical to that of the data held by the rest of the parties. Let p_0, p_1, and p_2, respectively, denote the probability that the data held by a party is 0, 1, and 2.

Theorem 9: The following protocol achieves the lowest average-case complexity for one-round pulse communication and ternary-input symmetric functions.

Protocol: Without loss of generality, assume that $p_0 \leq p_1 \leq p_2$. If the probabilities satisfy a different inequality the protocol can be easily modified by replacing 0 with the input with the lowest probability, 2 with the input the largest probability, and 1 with the remaining input. Suppose that the parties communicate in some order and that $0 \leq i$ parties have already communicated. Let $\mu_0(x^i)$, $\mu_1(x^i)$, and $\mu_2(x^i)$, respectively, be the number of 0's, 1's and 2's communicated so far, such that the function cannot yet be evaluated with this information. Let P_{i+1} be the next party to communicate.

Case 1: Suppose $\mu_0(x^i)$, $\mu_1(x^i)$, and $\mu_2(x^i)$ are such that in the future the parties need not differentiate between two particular inputs, say 0 and 2. Thus, the value of the function depends only on the number of 1's in the future. In other words

$$f(x^n) = g(\mu_1(x^n))$$

for some function g such that

$$\mu_j(x^n) \geq \mu_j(x^i), \quad j \in \{0,2\}.$$

The function now behaves as binary-input function and communication proceeds as described in Section III.

Case 2: If $\mu_0(x^i)$, $\mu_1(x^i)$, and $\mu_2(x^i)$ do not satisfy the conditions in Case 1, then the parties need to communicate the exact value of the data they hold. To minimize the communication complexity P_{i+1}

- stays silent to communicate the most probable input, i.e., if $x_{i+1} = 2$;
- sends a pulse to communicate the least probable input, i.e., if $x_{i+1} = 0$; and
- stays silent first and then sends a pulse to communicate the third input, i.e., if $x_{i+1} = 1$.

Proof: The proof of the theorem, based on proof by contradiction, is similar to that of functions of binary inputs in Theorem 2. □

VII. Conclusion

We introduce and formalize the use of silence to convey information in multiple-party communication, through the use of pulses that convey no information apart from their existence or absence. We study, in the framework of communication complexity, the average and worst case complexity required for pulse as opposed to bit communication for a number of cases

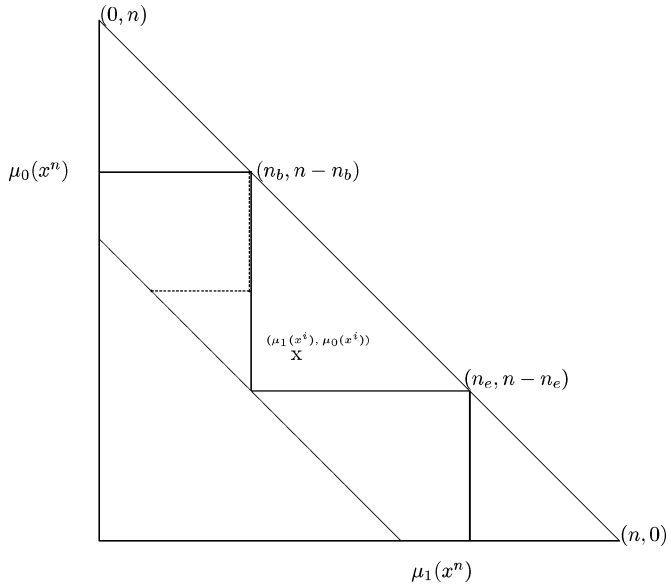

Fig. 3. Symmetric functions—f-constant intervals.

and propose protocols to achieve the optimal performance in each case. For nonbinary input functions, we also introduce the notion of one-round versus multiple-round communication that allows to capture practical considerations, and compare pulse and bit communication for each model. Finally, we extend our results to the general case of unequal cost bits, bit and pulse communication being the two extreme cases of equal cost and zero–one cost, respectively.

Appendix A
Proof of Theorem 3

Case 1: $c_0 = c_1$. This is equivalent to bit communication. Hence, (7) is clearly true.

Case 2: $c_0 = 0$. Equation (7) is again clearly true since this is equivalent to pulse communication.

Case 3: $0 < c_0 < c_1$. To show that (7) can be achieved consider the same protocol described in the proof of Theorem 1, the only difference being that a party sends the bit 1 instead of a pulse and sends the bit 0 instead of staying silent. Following the arguments described there it can be seen that

$$C \leq (n + 1 - I_{\max}) \cdot c_1 + (1 - I_{\max}) \cdot c_0.$$

To show that the complexity of any protocol is at least that in (7) consider Fig. 3. For a description of the figure see Section III-C. Consider any protocol Q. Suppose i parties have already communicated. Let the point with coordinates $(\mu_0(x^i), \mu_1(x^i))$ represent the data communicated so far. Whenever a party communicates a 1, the point moves vertically by one unit and whenever a party communicates a 0, the point moves horizontally by one unit. Once the point lies within an isosceles triangle corresponding to an f-constant interval, the function can be evaluated. Since the length of the largest f-constant interval is $I_{\max}$, knowing the data held by $n - I_{\max}$ is not sufficient to compute $f(x^n)$. As shown in the figure, the point representing the data communicated by $n - I_{\max}$ parties lies on the line $y + z = n - I_{\max}$ and outside the isosceles triangles.

Clearly, for any protocol Q and whatever the order in which the parties might communicates, there exists $x^{n-I_{\max}}$ such that the $n - I_{\max}$ parties all send a 1 to communicate their data and thus contribute $(n - I_{\max}) \cdot c_1$ to the complexity. At this stage the point $(\mu_0(x^{n-I_{\max}}), \mu_0(x^{n-I_{\max}}))$ lies on the line $y + z = n - I_{\max}$. Further, as shown in the figure, for any given point $(\mu_0(x^{n-I_{\max}}), \mu_0(x^{n-I_{\max}}))$ there exists $x^n_{n+1-I_{\max}}$ such that point $(\mu_0(x^n), \mu_1(x^n))$ is on the border of 2 f-constant intervals. For such data, all the parties need to communicate in the worst case. For example, if $\mu_1(x^{n-1}) = n_b - 1$, P_n, which holds x_n, has to communicate its data. Otherwise, the system will not be able to learn if $\mu_1(x^n) \in \mathcal{I}_{\max}$ (which is the case if $x_n = 1$) or not. To learn $x^n_{n+1-I_{\max}}$, any protocol would need at least $(I_{\max} - 1)c_0 + c_1$.

Thus, any protocol would have to learn the data of at least $n - I_{\max}$ parties which would cost $(n - I_{\max}) \cdot c_1$ in the worst case, and for any $x_1^{n-I_{\max}}$ there exists $x^n_{n+1-I_{\max}}$ such that it costs $(I_{\max} - 1) \cdot c_0 + c_1$ to compute f. Therefore

$$C_b(f) \geq (n + 1 - I_{\max}) \cdot c_1 + (I_{\max} - 1) \cdot c_0.$$

This completes the proof. □

Appendix B
Proof of Theorem 5

We first prove the upper bound. Let Q be any multiple-round communication protocol. Suppose that the protocol needs $c(Q, f(x^n))$ bits to compute $f(x^n)$. Since a single bit is communicated in each stage, the protocol computes $f(x^n)$ in $c(Q, f(x^n))$ stages and a total of $c(Q, f(x^n))$, not necessarily distinct, parties communicate. We now describe a one-round protocol Q^1 which needs at most $\log m \cdot c(Q, f(x^n))$ bits to compute $f(x^n)$.

Consider a protocol Q^1 that proceeds as follows. The parties under the protocol Q^1 communicate in the same order as they would under the protocol Q but instead of communicating a bit in a single stage they now send $\log m$ bits in $\log m$ consecutive stages to communicate the exact value of the data they hold. If the party chosen has already communicated, it would not communicate anymore. Since Q can compute $f(x^n)$, so can Q^1. Clearly, Q^1 is a one-round protocol and it computes $f(x^n)$ in at most $c(Q, f(x^n))$ stages. Thus, $c(Q^1, x^n) \leq \log m \cdot c(Q, f(x^n))$ for all x^n. Therefore, $C(Q^1, f)$ is higher than $C(Q, f)$ at most by a factor of $\log m$, i.e.,

$$C(Q^1, f) \leq \log m \cdot C(Q, f). \tag{16}$$

Thus, given any unrestricted communication protocol Q, including the optimal protocol, one could find a one-round protocol Q^1 such that (16) holds. Therefore, there exits a one-round protocol Q^1 such that

$$C(Q^1, f) \leq \log m \cdot C_b(f).$$

Hence

$$C_b^1(f) \leq \log m \cdot C_b(f).$$

This completes the proof for the upper bound.

We next prove the lower bound of Theorem 5 by describing a function which achieves it.

Let the input set $\mathcal{X}$ be of size $m = 2^k$ for some positive integer k. Let $b(x) \stackrel{\text{def}}{=} b_1(x)b_2(x)\cdots b_k(x)$ denote the binary representation of $x \in \mathcal{X}$ where $b_i(x)$ is the ith bit in the representation. For example, for $k = 4$, $b(5) = 0101$. Let the string x^n denote the data held by the n parties and let l_i denote the number of parties which have 1 as their ith bits

$$l_i = |\{x_j : b_i(x_j) = 1, 1 \le j \le n\}|$$

where $1 \le i \le k$. Consider a function over $\mathcal{X}^n$ defined as follows:

$$f(x^n) = \begin{cases} l_1 \cdot n + l_{l_1}, & 2 \le l_1 \le k \\ l_1 & \text{otherwise.} \end{cases}$$

In other words, the value of the functions depends on l_1, the number of parties with 1 as their first bit, and on l_{l_1}, the number of parties with 1 as their l_1th bit.

Multiple-round communication: Consider a multiple-round protocol Q under which the parties communicate in some order. First, the parties communicate so that l_1, the number of parties which have 1 as the first bit of their binary representation, can be determined. Once l_1 is known, the parties then communicate again if necessary to determine l_{l_1}.

To learn l_1, it is sufficient if each party P_i which holds the data x_i sends its first bit, $b_1(x_i)$. This would last n stages and a total of n bits, one by each party, are communicated. If $l_1 > k$ or $l_1 \le 1$, $f(x^n) = l_1$, the function can be evaluated and communication ends. If $2 \le l_1 \le k$, the parties now learn l_{l_1} by communicating $b_{l_1}(x_i)$, their l_{l_1}th bit. This would last another n stages and cost another n bits. Thus, in the worst case, $2 \cdot n$ bits are needed to compute $f(x^n)$. Therefore

$$C_b(f) \le C(Q, f) = 2 \cdot n. \tag{17}$$

One-round communication: Consider x^n such that $b_1(x_i) = 0$ for $1 \le i \le n-k$ and $b_1(x_i) = 1$ for $n-k < i \le n$. Since a one round protocol would not be able to determine l_1 until all n parties communicate and since each party gets only one chance to communicate, it is clear that for such x^n the ith party, for $1 \le i \le n-k$, needs to transmit x_i completely. Since x_i can be one of m distinct values, the party needs to send $\log m = k$ bits in the worst case. Thus, the first $n-k$ parties send k bits each in the worst case. Therefore

$$C_b^1(f) > (n-k) \cdot \log m. \tag{18}$$

Hence, $C_b^1(f) > \frac{\log m}{2} \cdot (1 - \frac{\log m}{n}) \cdot C_b(f)$. This completes the proof for Theorem 5. □

Appendix C
Proof of Theorem 7

To prove the theorem, we will provide a protocol that achieves this complexity.

Protocol: Suppose that the parties communicate in some order. Suppose that i parties have already communicated the data they respectively hold. Let $\mu_0(x^i), \mu_1(x^i)$, and $\mu_2(x^i)$, respectively denote the number of 0's, 1's, and 2's communicated by the first i parties. Note that $\mu_0(x^i) + \mu_1(x^i) + \mu_2(x^i) = i$. If

$$\mu_j(x^i) \le \max_r m_j^r, \quad 0 \le j \le 2 \tag{19}$$

the party P_{i+1} which holds x_{i+1} communicates as follows:

- transmits a pulse if $x_{i+1} = 1$;
- stays silent in the first stage and transmits a pulse in the second stage if $x_{i+1} = 2$;
- stays silent in both stages if $x_{i+1} = 0$.

If one of $\mu_j(x^i)$, say for $j = 1$, is at least $\max_r m_j^r$, i.e.,

$$\mu_0(x^i) \le \max_r m_0^r, \ \mu_2(x^i) \le \max_r m_2^r$$

and

$$\mu_1(x^i) \ge \max_r m_1^r$$

the party P_{i+1} communicates as follows:

- transmits a pulse in the first stage if $x_{i+1} = 0$;
- stays silent in the first stage and transmits a pulse in the second stage if $x_{i+1} = 2$;
- stays silent in both stages if $x_{i+1} = 1$.

If $\mu_j(x^i)$ is at least $\max_r m_j^r$, for two different values of j, say for $j = 1, 2$, i.e.,

$$\mu_1(x^i) \ge \max_r m_1^r, \ \mu_2(x^i) \ge \max_r m_2^r$$

and

$$\mu_0(x^i) \le \max_r m_0^r$$

the party P_{i+1} communicates as follows:

- transmits a pulse if $x_{i+1} = 0$;
- stays silent $x_{i+1} \in \{1, 2\}$.

If

$$\mu_j(x^i) \ge \max_r m_j^r, \quad 0 \le j \le 2$$

communication stops and the protocol outputs the value of the function.

Proof: At the outset, all three inequalities in (19) hold and communication proceeds according to the protocol until one of the three inequalities is satisfied with an equality for some j, say, $j = 1$. Suppose i_1 parties have communicated so far. These parties transmit a total of at most

$$\sum_{j \ne 1} \mu_j(x^{i_1}) + \max_r m_1^r$$

pulses.

In the future, the parties expend a pulse only if the data they hold is 0 or 2 as described in the protocol. Suppose $i_2 \ge i_1$ parties have communicated such that only one of the three inequalities in (19) holds, say for $j = 0$. At this stage

$$\mu_1(x^{i_3}) \ge \max_r m_1^r \ge m_1^0$$
$$\mu_2(x^{i_3}) \ge \max_r m_2^r \ge m_2^0.$$

Note that f is (m_0^0, m_1^0, m_2^0) 0-dependent. By definition, the value of the function, given that there are at least m_1^0 1's and m_2^0 2's, depends only on the number of 0's. The function now

behaves like a binary-input function with inputs 0 and $\overline{0}$. Therefore, in the future, the protocol does not differentiate between 1 and 2. At this stage, the parties have transmitted a total of at most

$$\mu_0(x^{i_3}) + \max_r m_1^r + \max_r m_2^r$$

where $\mu_0(x^{i_3}) \leq \max_r m_0^r$.

Communication proceeds as described in the protocol. Suppose $i_3 \geq i_2$ have communicated such that

$$\mu_j(x^i) \geq \max_r m_j^r, \quad 0 \leq j \leq 2.$$

If such an i_3 does not exist, all n parties communicate. In either case, the parties send a pulse to communicate at most $\max_r m_0^r$ 0's and by the definition of 0-dependency, the function can be computed. Thus, the protocol uses at most

$$\max_r m_0^r + \max_r m_1^r + \max_r m_2^r$$

pulses to compute the function. □

Appendix D
Proof of Theorem 8

The outline of the proof is as follows. We first show that $C_p^1(f) \geq (m_0^0 + m_1^0 + m_2^0)$. We do this by considering the data x^n for which the first 0 appears after a certain stage where the parties no longer need to differentiate between 1's and 2's. We show that a party can do this at a stage only if the number of 1's and 2's communicated so far correspond to a 0-dependency 3-tuple. The function, beyond this stage, behaves like a binary-input function for which the optimal protocol is already known. If the protocol sends a pulse each for $\acute{m}_0$ 0's, $\acute{m}_1$ 1's, and $\acute{m}_0$ 2's, we show that f is $(\acute{m}_0, \acute{m}_1, \acute{m}_2)$ 0-dependent. Thus, any protocol and hence the optimal protocol, needs a total of $\acute{m}_0 + \acute{m}_1 + \acute{m}_2 \geq m_0^0 + m_1^0 + m_2^0$ pulses. The same argument holds for 1- and 2-dependency 3-tuples thus proving the theorem.

Let x^i represent the substring formed by the first i elements of x^n. Let $\mu_0(x^i)$, $\mu_1(x^i)$, and $\mu_2(x^i)$ denote the number of 0's, 1's, and 2's in x^i. Suppose that the data x^n is such that the first 0 appears after the stage where the parties no longer need to differentiate between 1's and 2's. In other words, there exists $1 \leq i \leq n$ for which $\mu_0(x^i) = 0$ and f is $(\acute{m}_0, \mu_1(x^i), \mu_2(x^i))$ 0-dependent for some $\acute{m}_0$. Let k be the largest such i, i.e., either $k = n$ or $x_{k+1} = 0$.

Let Q^1 be any communication protocol. At the outset, each party needs to communicate the exact value of the data it holds. Suppose $0 \leq i \leq k$ parties have already communicated such that a party still needs to differentiate between 1's and 2's. Since a party could stay silent for only one of 0, 1, and 2, it needs to expend a pulse to communicate either 1 or 2. Let $x_{i+1} \in \{1, 2\}$ be such that the party P_{i+1} spends a pulse to communicate x_{i+1} when communicating according to the protocol Q^1. Suppose that after j parties have communicated, the parties no longer need to differentiate between 1 and 2. The j parties send j pulses to communicate each of the $\mu_1(x^j)$ 1's and $\mu_2(x^j)$ 2's.

We claim that this must correspond to some 0-dependency 3-tuple, i.e., $\exists\ m_0(x^j)$ such that f is $(m_0(x^j), \mu_1(x^j), \mu_2(x^j))$ 0-dependent. Suppose this is not true. Then, by definition, there exists x^n such that $f(x^n)$ depends on $\mu_0(x^n)$, $\mu_1(x^n)$, and $\mu_2(x^n)$ and not just on $\mu_0(x^n)$ and the knowledge that $\mu_1(x^n) \geq \mu_1(x^j)$ and $\mu_2(x^n) \geq \mu_2(x^j)$ i.e., there exists $\acute{m}_0$, $\acute{m}_1$, and $\acute{m}_2$ such that $\acute{m}_0 + \acute{m}_1 + \acute{m}_2 = n - 1$, $\acute{m}_1 \geq \mu_1(x^j)$, $\acute{m}_2 \geq \mu_2(x^j)$, and

$$\begin{aligned} &f(x^n : \mu_1(x^n) = \acute{m}_1, \mu_2(x^n) = \acute{m}_2 + 1) \\ &\neq f(x^n : \mu_1(x^n) = \acute{m}_1 + 1, \mu_2(x^n) = \acute{m}_2). \end{aligned}$$

Therefore, if the protocol stops differentiating between 1's and 2's after the first j parties, then there exists x^n such that knowing $\mu_0(x^n)$ and that $\mu_1(x^n) \geq \mu_1(x^j)$ and $\mu_2(x^n) \geq \mu_2(x^j)$ alone is not sufficient to compute the function. Hence, the protocol should not have stopped differentiating between 1's and 2's yet.

If no such $\acute{m}_0, \acute{m}_1$, and $\acute{m}_2$ exist it follows that $f(x^n)$ depends only of $\mu_0(x^n)$ as long as $\mu_1(x^n) \geq \mu_1(x^j)$ and $\mu_2(x^n) \geq \mu_2(x^j)$. This contradicts our assumption that there does not exist an $m_0(x^n)$ such that f is $(m_0(x_1^j), \mu_1(x^j), \mu_2(x^j))$ 0-dependent. Therefore, a party could stop differentiating between 1's and 2's iff the number of 1's and 2's communicated so far corresponds to a 0-dependency 3-tuple. Otherwise, there exists x^n such that the protocol would not be able to compute $f(x^n)$.

If the protocol need not differentiate between 1's and 2's, the function now behaves as a binary-input function which we denote by $g_{f(x^j)}$. This is a function over $\{0, \overline{0}\}^{n-j}$ such that $f(x^n) = g_{f(x^j)}(\acute{x}_{j+1}^n)$, where

$$\acute{x_i} = \begin{cases} 0, & x_i = 0 \\ \overline{0}, & x_i \in \{1, 2\} \end{cases}$$

for $j < i \leq n$.

Let $\mathcal{I}_{\max}(x^j) = (n_b(x^j), \ldots, n_e(x^j))$ be the largest f-constant interval of $g_{f(x_1^j)}$. Communication now proceeds according to the protocol for binary-input functions as described in Section III-C. A party sends a pulse to communicate a 0 until $n_b(x_1^j)$ 0's are communicated. Once $n_b(x_1^j)$ 0's are communicated a party would send a pulse to communicate a $\overline{0}$ until $(n - j) + 1 - n_e(x_1^j)$ $\overline{0}$'s are communicated. Let x_{j+1}^n be such that

$$x_i = \begin{cases} 0, & j + 1 \leq i \leq j + n_b(x_1^j) \\ 1, & j + n_b(x_1^j) + 1 \leq i \leq n. \end{cases}$$

Note that $x_i = 1 \equiv \acute{x_i} = \overline{0}$. When communication ends

$$j + n_b(x_1^j) + (n - j) + 1 - n_e(x_1^j) = n + 1 + n_b(x_1^j) - n_e(x_1^j)$$

parties would have communicated, with parties sending a pulse each to convey $\mu_1(x^j)$ 1's , $\mu_2(x^j)$ 2's, $n_b(x_1^j)$ 0's, and $n - j + 1 - n_e(x_1^j)$ $\overline{0}$'s in that order.

Since there exists $m_0(x^j)$ such that f is $(m_0(x_1^j), \mu_1(x^j), \mu_2(x^j))$ 0-dependent, it follows from the definition of 0-dependency that f is also

$(n_b(x_1^j), \mu_1(x^j)+l, \mu_2(x^j)+(n-j)+1-n_e(x_1^j)-l)$ 0-dependent for $0 \le l \le (n-j)+1-n_e(x_1^j)$. Hence

$$\begin{aligned} C_p^1(Q^1, f) &\ge c(Q^1, f(x^n)) \\ &= n_b(x^j) + \mu_1(x^j) + k + \mu_2(x^j) \\ &\quad + (n-j) + 1 - n_e(x_1^j) - k \\ &\ge m_0^0 + m_1^0 + m_2^0. \end{aligned}$$

The last inequality holds because (m_0^0, m_1^0, m_2^0) is a 3-tuple with the smallest sum such that f is (m_0^0, m_1^0, m_2^0) 0-dependent.

Similar arguments holds for 1- and 2-dependency, leading to

$$C_p^1(f) \ge \max_i (m_0^i + m_1^i + m_2^i).$$

This completes the proof of Theorem 8. □

REFERENCES

[1] H. Abelson, "Lower bounds on information transfer in distributed computations," in *Proc. 19th Ann. Symp. Foundations of Computer Science*, Ann Arbor, MI, Oct. 1978, pp. 151–158.

[2] A. C. Yao, "Some complexity questions related to distributive computing," in *Proc. 11th Annu. ACM Symp. Theory of Computing*, Atlanta, GA, Apr. 1979, pp. 209–213.

[3] A. Orlitsky and A. El Gamal, "Communication complexity," in *Complexity in Information Theory*, Y. Abu Mustafa, Ed. New York: Springer-Verlag, 1986, pp. 16–61.

[4] E. Kushilevitz and N. Nisan, *Communication Complexity*. Cambridge, U.K.: Cambridge Univ. Press, 1996.

[5] P. Duris, Z. Galil, and G. Schnitger, "Lower bounds on communication complexity," *Inf. Comput.*, vol. 73, no. 1, pp. 1–22, Apr. 1987.

[6] D. Dolev and T. Feder, "Multiparty communication complexity," in *Proc. 30th IEEE Symp. Foundations of Computer Science*, Triangle Park, NC, Oct./Nov. 1989, pp. 428–433.

[7] L. Babai, N. Nisan, and M. Szegedy, "Multiparty protocols, pseudorandaom generators for LOGSPACE, and time-space trade-offs," *J. Comput. Syst. Sci.*, vol. 45, no. 2, pp. 204–232, 1992.

[8] R. L. Graham, B. Rothschild, and J. H. Spencer, *Ramsey Theory*, 2nd ed. New York: Wiley, 1990.

[9] A. K. Chandra, M. L. Furst, and R. J. Lipton, "Multi-part protocols," in *Proc. 15th ACM Symp. Theory of Computing*, Boston, MA, May 1983, pp. 94–99.

[10] R. K. Chung and P. Tetali, "Communication complexity and quasi randomness," *SIAM J. Discr. Math.*, vol. 6, no. 1, pp. 110–123, 1993.

[11] V. Grolmusz, "The BNS Lower bound for multi-party protocols is nearly optimal," *Inf. Comput.*, vol. 112, no. 1, pp. 51–54, 1994.

[12] L. Babai, P. Kimmel, and S. V. Lokam, "Simultaneous messages vs. communication," in *Proc. 12th Symp. Theoretical Aspects of Computer Science*, Munich, Germany, Mar. 1995.

[13] Y. C. Cheng and T. G. Robertazzi, "Distributed computation with communication delay," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 24, no. 6, pp. 700–172, Nov. 1988.

[14] Y. Xu and H. Qi, "Distributed computing paradigms for collaborative signal and information processing in sensor networks," *J. Parallel Distrib. Comput.*, vol. 64, pp. 945–959, 2004.

[15] S. Tatikonda and K. Ramachandran, "Coding and routing complexity in sensor networks," in *Proc. IEEE Int. Symp. Information Theory*, Chicago, IL, Jun./Jul. 2004, p. 117.

[16] 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.

[17] L. Ying, R. Srikanth, and G. E. Dullerud, "Distributed symmetric function computation in noisy wireless sensor networks with binary data," in *Proc. 4th Int. Symp. Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Boston, MA, Apr. 2006, pp. 1–9.

[18] T. Tyrakowski and Z. Palka, "On the communication complexity of Bar-Yehuda, Goldreich and Itai's randomized broadcasting algorithm," *J. Discr. Algorithms*, vol. 5, pp. 323–329, 2007.

[19] N. M. Blachman, "Minimum-cost encoding of information," *IRE Trans. Information Theory*, vol. PGTI-3, pp. 139–149, 1954.

[20] D. M. Choy and C. K. Wong, "Bounds for optimal α — β binary trees," *BIT*, vol. 17, pp. 1–15, 1997.

[21] N. Cot, "A linear-time ordering procedure with applications to variable length encoding," in *Proc. 8th Princeton Conf. Information Sciences and Systems*, Princeton, NJ, 1974, pp. 460–467.

[22] Y. Perl, M. R. Garey, and S. Even, "Efficient generation of optimal prefix code: Equiprobable words using unequal cost letters," *J. ACM*, vol. 22, pp. 202–214, 1975.

[23] L. E. Stanfel, "Tree structuring for optimal searching," *J. ACM*, vol. 17, pp. 508–517, 1970.

[24] B. Varn, "Optimal variable length codes (arbitrary symbol cost and equal codeword probability)," *Inf. Contr.*, vol. 19, pp. 289–301, 1971.

[25] R. M. Karp, "Minimum-redundancy coding for the discrete noiseless channel," *IRE Trans. Inf. Theory*, vol. IT-7, no. 1, pp. 27–39, Jan. 1961.

[26] S. Verdú, "On channel capacity per unit cost," *IEEE Trans. Inf. Theory*, vol. 36, no. 5, pp. 1019–1030, Sep. 1990.

[27] S. Savari and A. Naheta, "Bounds on the expected cost of one-to-one codes," in *Proc. IEEE Int. Symp. Information Theory*, Chicago, IL, Jun. 2004, p. 92.

[28] T. Cover and J. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

Anand K. Dhulipala received the B.Tech. degree in electrical engineering from Indian Institute of Technology (IIT) Madras and the M.S. and Ph.D. degrees in electrical engineering from the University of California, San Diego.

He currently works as an Analytic Science Scientist in the Fraud Analytic Group at FICO, San Diego. CA.

Christina Fragouli (S'98– M'04) received the B.S. degree in electrical engineering from the National Technical University of Athens, Athens, Greece, in 1996 and the M.Sc. and Ph.D. degrees in electrical engineering from the University of California, Los Angeles, in 1998 and 2000, respectively.

She has worked at the Information Sciences Center, AT&T Labs, Florham Park, NJ, and the National University of Athens. She also visited Bell Laboratories, Murray Hill, NJ, and DIMACS, Rutgers University, Piscataway, NJ. From 2006 to 2007, she was an FNS Assistant Professor in the School of Computer and Communication Sciences, EPFL, Switzerland and is now Assistant Professor there. Her research interests are in network information flow theory and algorithms, network coding, and connections between communications and computer science.

Prof. Fragouli served as an Editor for IEEE COMMUNICATIONS LETTERS, and is currently serving as an Editor for IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE TRANSACTIONS ON INFORMATION THEORY. She received the Fulbright Fellowship for her graduate studies, the Outstanding Ph.D. Student Award 2000–2001, UCLA, Electrical Engineering Department, the Zonta award 2008 in Switzerland, and the ERC Starting Grant from the European Research Council in 2009.

Alon Orlitsky (S'83–M'84–SM'00–F'06) received the B.Sc. degrees in mathematics and electrical engineering from Ben Gurion University, Be'er-Sheva, Israel, in 1980 and 1981, and the M.Sc. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1982 and 1986.

From 1986 to 1996, he was with the Communications Analysis Research Department of Bell Laboratories, Murray Hill, NJ. He spent the following year as a quantitative analyst at D.E. Shaw and Company, an investment firm in New York City. In 1997, he joined the University of California, San Diego (UCSD), where he is currently a Professor of Electrical and Computer Engineering and of Computer Science and Engineering, and directs the Information Theory and Applications Center. His research concerns information theory, statistical modeling, machine learning, and speech recognition.

Prof. Orlitsky is a recipient of the 1981 ITT International Fellowship and the 1992 IEEE W.R.G. Baker Paper Award, a corecipient of the 2006 Information Theory Society Paper Award, and holds the Qualcomm Chair for Information Theory and its Applications at UCSD.