Amortized Communication Complexity

(Preliminary Version)

Tomàs Feder
Bellcore

Eyal Kushilevitz*
Computer Science Department
Technion - Israel Institute of Technology

Moni Naor†
IBM Research Division
Almaden Research Center

Abstract

In this work we study the *direct sum* problem with respect to communication complexity: Consider a function $f : D \rightarrow \{0,1\}$, where $D \subseteq \{0,1\}^n \times \{0,1\}^n$. Can the communication complexity of simultaneously computing f on ℓ instances $(x_1, y_1), \ldots, (x_\ell, y_\ell)$ be smaller than the communication complexity of computing f on the ℓ instances, separately ?

Let the *amortized* communication complexity of f be the communication complexity of simultaneously computing f on ℓ instances, divided by ℓ. We study the properties of the amortized communication complexity. We show that the amortized communication complexity of a function can be smaller than its communication complexity. More precisely, we present, both in the *deterministic* and the *randomized* models, functions with communication complexity $\Theta(\log n)$ and amortized communication complexity $O(1)$.

We also give a general lower bound on the amortized communication complexity of any function f in terms of its communication complexity $C(f)$: for every function f the amortized communication complexity of f is $\Omega\left(\sqrt{C(f)} - \log n\right)$.

*Research was supported in part by US-Israel BSF grant 88-00282. e-mail: eyalk@techunix.bitnet
†e-mail naor@ibm.com

1 Introduction

A very basic question in the theory of computation is the *direct-sum* question: Can the cost of solving ℓ independent instances of a problem simultaneously be smaller than the cost of independently solving the ℓ problems, say, sequentially? In this work we study the direct-sum question in the context of communication complexity. This question was recently raised by Karchmer, Raz and Wigderson [6] as part of a new approach for proving lower bounds on Boolean circuits using communication complexity arguments (as in [7, 17]). For a general survey on communication complexity, see [10]. Different scenarios where the direct sum question was investigated are [3, 5, 16, 18].

Let f be a Boolean function defined over a domain $D \subseteq \{0,1\}^n \times \{0,1\}^n$. Let $f^{(\ell)} : D^\ell \rightarrow \{0,1\}^\ell$ be the extension of f to ℓ instances. Denote by $C(f)$, the communication complexity of f, and by $\overline{C}(f)$, the amortized communication complexity of f. Namely,

$$\overline{C}(f) = \limsup_{\ell \rightarrow \infty} \frac{1}{\ell} C(f^{(\ell)}).$$

Clearly, $\overline{C}(f) \leq C(f)$ for every function f. It was observed in [6] that when functions over *full domain* ($\{0,1\}^n \times \{0,1\}^n$) are considered, an upper bound on $\overline{C}(f)$ which is significantly smaller than $C(f)$, implies that the *rank* lower-bound on $C(f)$ [11] is not tight. This is because the rank of the matrix represents $f^{(\ell)}$ equals the rank of the matrix represents f, to the power of ℓ.

We present a function f (defined over partial domain), such that $C(f) = \Theta(\log n)$ and $\overline{C}(f) =$

$O(1)$. This proves that computing a function f on ℓ instances simultaneously may be easier than computing f on the ℓ instances separately. In [6], it was conjectured that $\overline{C}(f)$ can not be smaller than $C(f)$ by more than an *additive* factor of $O(\log n)$. We prove two weaker versions of this conjecture:

- If *one-way* communication protocols are considered then *any* function f with a domain $D \subseteq \{0,1\}^n \times \{0,1\}^n$ satisfy $\overline{C}_1(f) \geq C_1(f) - \log n - O(1)$.
- For *general* protocols, any function f over $\{0,1\}^n \times \{0,1\}^n$ satisfy $\overline{C}(f) \geq \sqrt{C(f)/2} - \log n - O(1)$. (The lower bound holds also for functions defined over *partial* domain under stronger requirements for computing $f^{(\ell)}$.)

The proof of the first lower bound is via a reduction to an appropriate graph-coloring problem, and then applying the results of Linial and Vazirani [9] on the chromatic number of product graphs. The lower bound for general protocols is achieved by considering *non-deterministic* protocols and proving that $\overline{C}_N(f) \geq C_N(f) - \log n - O(1)$, and then applying the result of Aho, Ullman and Yannakakis [1] which relates nondeterministic communication complexity and deterministic communication complexity.

We also study the direct-sum question with respect to *randomized* protocols. The trivial upper bound on $C_R(f^{(\ell)})$ in this case is $O(\ell \cdot \log \ell \cdot C_R(f))$ (the $\log \ell$ factor seems to be needed, since we are required to have a "good" probability of success on *all* ℓ instances simultaneously). For explicit functions we can do much better: We consider the *identity* function (i.e., $ID : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ defined by $ID(x,y) = 1$ iff $x = y$). It is well known that $C_R(ID) = \Theta(\log n)$ [21]. We prove that $\overline{C}_R(ID) = O(1)$.

Organization: In section 2 the notions of amortized communication complexity are defined. In section 3 we exhibit a function whose amortized communication complexity is smaller than its communication complexity. In section 4 we discuss the special case of one-way communication protocols. In section 5 we prove our general lower bound on the amortized communication complexity. Finally, in section 6 we present a function whose amortized communication complexity is smaller than its communication complexity, when randomized protocols are considered.

2 Preliminaries

In this section we give formal definitions for the various notions of communication complexity. Two parties P_1 and P_2 wish to compute the value of a Boolean function f defined on a domain $D \subseteq \{0,1\}^n \times \{0,1\}^n$ (i.e., f is not necessarily defined for every $(x,y) \in \{0,1\}^n \times \{0,1\}^n$. On the other hand, it is convenient to assume that every x is possible and every y is possible). The party P_1 is given a n-bit input x and the party P_2 is given a n-bit input y, such that $(x,y) \in D$. The parties exchange messages, where each message sent by a party P_i depends on its input, and the messages it received in previous rounds. The last message in the protocol contains the value of $f(x,y)$. The concatenation of all the messages exchanged in the protocol $\mathcal{F}$ on input (x,y) is denoted $\mathcal{F}(x,y)$. The communication complexity of the protocol $\mathcal{F}$, denoted $C(\mathcal{F})$, is the maximum $|\mathcal{F}(x,y)|$ over all $(x,y) \in D$. The communication complexity of the function f, denoted $C(f)$, is the minimum of $C(\mathcal{F})$, over all protocols $\mathcal{F}$ computing f.

Given a function f, we define the function $f^{(\ell)} : D^\ell \to \{0,1\}^\ell$ as follows:

$$\begin{aligned} & f^{(\ell)}((x_1,y_1),(x_2,y_2),\ldots,(x_\ell,y_\ell)) \\ = \; & (f(x_1,y_1), f(x_2,y_2), \ldots, f(x_\ell,y_\ell)) \end{aligned}$$

The *amortized communication complexity* of the function f is defined as

$$\overline{C}(f) = \limsup_{\ell \to \infty} \frac{1}{\ell} C(f^{(\ell)}).$$

Similar definitions holds for *nondeterministic* protocols. In such a protocol the parties are required to compute the correct value of $f(x,y)$ in some computation on (x,y) but are allowed to output "fail". $C_N(\mathcal{F}), C_N(f)$ and $\overline{C}_N(f)$ are defined with respect to nondeterministic protocols.

We also consider *randomized* protocols, in which each of the parties has, in addition to its input, a string of random coins (the random strings of the two parties are independent). For such protocols we require that the last message in the protocol is equal to $f(x,y)$ only with probability $\geq \frac{3}{4}$. For such protocols the notions of $C_R(\mathcal{F}), C_R(f)$ and $\overline{C}_R(f)$ are defined in a similar way. Note that when computing $f^{(\ell)}$ we require that with probability at least $3/4$ *all* ℓ instances be correct simultaneously.

It is also useful to consider a variant of the randomized model in which both parties have access to a *public* random string. The quantities $C_{pub}(f)$ and $\overline{C_{pub}}(f)$ are defined in a similar way.

3 A Function With a Low Amortized Complexity

In this section we prove that amortized communication complexity can be substantially lower than the corresponding communication complexity. We present a function f, defined over a partial domain of $\{0,1\}^n \times \{0,1\}^n$, such that $C(f) = \Theta(\log n)$, while $\overline{C}(f) = O(1)$.

We start with the definition of f: Let $M = \{0,1,2,\ldots,m-1\}$. Let $t \geq 2$, be a parameter. The input of P_1 is S, a subset of M of size t (the length of this input is $n = t \cdot \log m$ bits). The input of P_2 is $x \in S$ (the length of this input is $\log m < n$ bits). The parties wish to compute the rank of x in the subset S. Orlitsky [15] showed that the communication complexity of this function is $C(f) = \Theta(\log t + \log\log m)$.

The protocols we present make use of the following set of hash-functions suggested by Fredman, Komlòs and Szemerèdi [4]: Let $p \simeq t^2 \log m$ be a prime. Define

$$\begin{aligned} H &= \{h : M \to \{0,1,\ldots,2t^2-1\} \mid\mid h(x) \\ &= (ax \bmod p) \bmod 2t^2;\ 1 \leq a \leq p-1\} \end{aligned}$$

We say that $h \in H$ is *good* with respect to a set $S \subset M$ if h is $1-1$ with respect to the elements of S. It is known [4] that for every S of size t at least $\frac{1}{2}$ of the functions in H are good.

We start by presenting the following protocol from [15] that meets the lower bound for computing f on a *single* instance (S,x). This protocol has the advantage that an appropriate generalization of it gives the amortized result.

- P_1 finds a function $h \in H$ which is good with respect to S. It sends its name ($O(\log t + \log\log m)$ bits) to P_2.
- P_2 computes $h(x)$ and sends this value ($O(\log t)$ bits) to P_1.
- Since h is good with respect to S, then the value $h(x)$ determines $x \in S$. Now P_1 can compute the value $f(S,x)$, and sends it to P_2 ($O(\log t)$ bits).

We now show how to generalize the protocol in order to efficiently compute the values $f(S_1,x_1), f(S_2,x_2),\ldots,f(S_\ell,x_\ell)$ simultaneously. The main idea is the following: since for every S at least half of the hash-functions are good, then given $S_1,\ldots,S_\ell$ we can construct a set $L = \{h_1,h_2,\ldots,h_{\log\ell}\}$ of hash-functions such that the function h_1 is good with respect to at least half of the S_i's, the function h_2 is good with respect to at least half of the remaining S_i's (those for which h_1 was bad), and so on. Thus, for every S_i at least one of the $\log\ell$ functions in L, denoted $h_{j(i)}$, is good. (To prove that such a set of functions exists - consider a matrix with the S_i's as rows, and the hash functions in H as columns. The (S,h) entry is 1 if h is good with respect to S and 0 otherwise. Using the fact that for every subset S, at least half of the functions in H are good, the proof now is straightforward). The following protocol computes f on ℓ instances simultaneously:

- P_1 finds a set L of $\log\ell$ hash functions as above, and sends the names of functions in L ($O(\log\ell \cdot (\log t + \log\log m))$ bits) to P_2. In addition, for every $1 \leq i \leq \ell$, it sends the index $j(i)$. This requires only $O(\ell)$ bits, since h_1 is good for about $\frac{1}{2}$ of the sets, h_2 is good for about $\frac{1}{4}$ of the sets etc. By using, say Huffman coding, we get the desired result.
- P_2 computes $h_{j(i)}(x_i)$, for every i, and sends it to P_1 ($O(\ell \cdot \log t)$ bits).

- Since $h_{j(i)}$ is good with respect to S_i, the party P_1 knows the value of x_i for every $1 \leq i \leq \ell$ and thus can compute $f(S_1, x_1), \ldots, f(S_\ell, x_\ell)$.

All together the protocol takes $O(\ell \cdot \log t + \log \ell \cdot (\log t + \log\log m))$ bits. Take, for example, $t = 2$ and recall that in this case the length of the input satisfies $n = 2\log m$, we get that the number of bits exchanged in this protocol is $O(\ell + \log \ell \cdot \log n)$. Therefore we have a function with $C(f) = \Theta(\log n)$, and $\overline{C}(f) = \limsup_{\ell\to\infty} \frac{1}{\ell} C(f^{(\ell)}) = O(1)$.

4 One-Way Communication

In this section we deal with one-way communication protocols. We show that if we restrict the discussion to the computation of (partial) functions using one-way protocols (i.e., P_1 sends a message to P_2 and P_2 has to compute the value of the function) then we can still "save" bits by computing f on many instances simultaneously. In fact, the function f of the previous section yields such an example: take $t = 2$ and assume that $S_i = \{y_i^1, y_i^2\}$ where $0 \leq y_1^i < y_2^i \leq m-1$. As stated before, $C(f) = \Theta(\log n)$ (and clearly $C_1(f) \geq C(f)$). On the other hand, a slight modification of the previous protocol gives $\overline{C}_1(f) = O(1)$: P_1 sends together with the list L of hash functions also $h_{j(i)}(y_1^i)$ and $h_{j(i)}(y_2^i)$ for $1 \leq i \leq \ell$. Now P_2 can decide whether $x_i = y_i^1$ or $x_i = y_i^2$.

On the other hand, we can prove that for every function f no more than $\log n$ bits can be saved: $\overline{C}_1(f) \geq C_1(f) - \log n - O(1)$. We start with a simple theorem, which claims that if f is of full domain (i.e., $D = \{0,1\}^n \times \{0,1\}^n$) then $\overline{C}_1(f) \cong C_1(f)$.

Theorem 1: Let f be a function defined on $\{0,1\}^n \times \{0,1\}^n$. Then $\overline{C}_1(f) \cong C_1(f)$.

Proof: Define the following relation on the inputs of P_1: $x_1 \sim x_2$ if $f(x_1, y) = f(x_2, y)$, for every y. Clearly $\sim$ is an equivalence relation. Denote by $Class(f)$ the number of equivalence classes of the $\sim$ relation. It can be easily verified that $C_1(f)$ is exactly $\lceil \log Class(f) \rceil$. This is true, since P_1 can send on input x the index of equivalence class for which x belongs. From this information P_2 can compute $f(x,y)$. On the other hand, if for two inputs in different equivalence classes P_1 sends the same string then clearly the protocol is wrong for at least one of them (on some y). Using similar arguments, one can prove that for computing $f^{(\ell)}$ the party P_1 must use $Class(f^{(\ell)}) = Class(f)^\ell$ different strings. As this number of strings is enough, the theorem follows. □

The following theorem claims that this result cannot be extended to functions defined over partial domain. The key point is that for partial functions $\sim$ is not necessarily an equivalence relation.

Theorem 2: Let f be a function defined over $D \subseteq \{0,1\}^n \times \{0,1\}^n$. Then $C_1(f^{(2)}) \geq 2C_1(f) - \log n - O(1)$.

Proof: The idea of the proof is to reduce the problem of the one-way communication complexity of a function to an appropriate graph-coloring problem,[1] and then to use results of Linial and Vazirani [9] on this problem.

We construct a graph $G_f = (V, E)$ as follows: The vertices are all the possible x's. There is an edge between x_1 and x_2 if $x_1 \not\sim x_2$. Namely, if there exists y such that (x_1, y) and (x_2, y) are both in D and $f(x_1, y) \neq f(x_2, y)$.

The number of different messages used by the optimal one-way communication protocol is exactly the *chromatic number* of G_f (denoted $\chi(G_f)$): If we have a legal coloring of G_f then this coloring defines a one-way communication protocol for computing f - P_1 sends the color c of its input x. This color together with P_2's input y determine $f(x,y)$ (if there is x' colored by c such that (x', y) is also in D and $f(x,y) \neq f(x',y)$, then the coloring is illegal). On the other hand, every protocol induces a legal coloring of G_f where the color of every x is the message P_1 sends on it.

[1] Similar reductions appear in [15, 19]. In these works the two parties have an input $(x,y) \in D$ and P_1 has to transmit its information x to P_2. This problem corresponds in our setting to the problem of computing the specific function $f(x,y) = x$ over the domain D.

Now, we define the *product* operation on graphs: Given $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ the vertices set of the product $G_1 \times G_2$ is $V_1 \times V_2$. The edge set includes all the edges $((v_1, v_2), (u_1, u_2))$ such that $(v_1, u_1) \in E_1$ *or* $(v_2, u_2) \in E_2$. (In the terminology of [9] this is called *inclusive-product*). It is easy to verify that $G_{f^{(2)}} = G_f \times G_f$.

Using this reduction to the graph-coloring problem we can now prove the theorem: it is enough to prove that for every f the chromatic number satisfy $\chi(G_{f^{(2)}}) \geq \frac{\chi^2(G_f)}{cn}$, for some constant c. This is proved in [9, Theorem 1]. □

If we consider the above proof, and [9, Theorem 1], the following corollary follows:

Corollary 3: Let f be a function defined over $D_f \subseteq \{0,1\}^n \times \{0,1\}^n$, and let g be a function defined over $D_g \subseteq \{0,1\}^n \times \{0,1\}^n$. Let $f \times g$ be the function defined over $D_f \times D_g$ in the obvious way. Then, $C_1(f \times g) \geq C_1(f) + C_1(g) - \log n - O(1)$.

The last theorem also implies that $C_1(f^{(\ell)}) \geq \ell C_1(f) - \ell \log n - O(\ell)$. Therefore, we have

Corollary 4: Let f be a function defined over $D \subseteq \{0,1\}^n \times \{0,1\}^n$. Then $C_1(f) \geq \overline{C}_1(f) \geq C_1(f) - \log n - O(1)$.

For additional examples of functions with $\overline{C}_1(f)$ significantly smaller than $C_1(f)$, we show that for every graph G with 2^n vertices there exists a partial function f such that $G = G_f$. Label the vertices of G by strings in $\{0,1\}^n$ and define a function f as follows: for every x, $f(x,x) = 1$. For every edge $(x,y) \in E$ define $f(x,y) = 0$. All the undefined pairs are not in the domain D. It can be easily verified that $G = G_f$. This implies that from every graph G with 2^n vertices, such that $\chi(G \times G) \cong \frac{\chi^2(G)}{cn}$, we can construct a partial function f such that $C_1(f^{(2)}) \cong 2C_1(f) - \log n - O(1)$. Examples of such graphs are given in [9, Theorem 2].

5 Lower Bound for General Protocols

In order to prove lower bounds on $\overline{C}(f)$ for a specific function f, we can use traditional techniques. For example, consider the *identity* function (i.e., $ID(x,y) = 1$ iff $x = y$). It is easy to verify that $\overline{C}(ID) = C(ID) = n$ (as in [21]). In this section we give a *general* lower bound on $\overline{C}(f)$ in terms of $C(f)$.

Our lower bound holds for functions defined over full domain. For extending the lower bound to functions defined over *partial* domain we have to give an appropriate (stronger) definition of protocols that compute $f^{(\ell)}$. If f is defined over $D \subseteq \{0,1\}^n \times \{0,1\}^n$, then $f^{(\ell)}$ is defined over D^ℓ. This implies that a protocol for computing $f^{(\ell)}$ is required to be correct only in case that *all* ℓ instances (x_i, y_i) belong to D. However, considering the protocol presented in Section 3, one can verify that it has a stronger property: on *any* input vector in $(\{0,1\}^n \times \{0,1\}^n)^\ell$, the protocol outputs a vector $(z_1, \ldots, z_\ell)$ such that if $(x_i, y_i) \in D$ then $z_i = f(x_i, y_i)$ (and if $(x_i, y_i) \notin D$ then there is no requirement with respect to z_i). We say that such a protocol computes $f^{(\ell)}$ in the *strong sense*. In this section all protocols are assumed to be strong.

We start with some definitions and notations that are used in the proof. Given a function $f : D \to \{0,1\}$ we denote by M_f the matrix representing this function. That is, each row of M_f corresponds to an input x of P_1, and each column corresponds to an input y of P_2. The entry (x,y) contains $f(x,y)$ if $(x,y) \in D$, and $\perp$ otherwise. Similarly, $M_{f^{(\ell)}}$ is defined as a matrix with entries which are ℓ-tuples with coordinates in $\{0,1,\perp\}$. A *monochromatic rectangle* of M_f is a set $R = R_x \times R_y$ such that all the non-$\perp$ entries in R have the same value. For *strong* protocols that compute $f^{(\ell)}$ we call a rectangle R of $M_{f^{(\ell)}}$ monochromatic if we can associate with R an output vector $z_R = (z_1, \ldots, z_\ell)$, in a way that for every input $(x_1, \ldots, x_\ell), (y_1, \ldots, y_\ell) \in R$ and for every $1 \leq i \leq \ell$, if $(x_i, y_i) \in D$ then $f(x_i, y_i) = z_i$. We denote by $N(f)$ the minimal number of monochromatic rectangles needed

to cover (possibly with overlaps) all the non-$\perp$ entries of M_f. Since any nondeterministic protocol for computing f induces such a cover, $\log N(f) \leq C_N(f)$.

The lower-bound on $\overline{C}(f)$ is given by first proving a lower bound on $N(f^{(\ell)})$ in terms of $N(f)$, and then applying known relations between deterministic and nondeterministic communication complexity [1]. The next theorem claims that $N(f^{(2)})$ cannot be much smaller than $N^2(f)$.

Theorem 5: Let $f : D \rightarrow \{0,1\}$, where $D \subseteq \{0,1\}^n \times \{0,1\}^n$. Then, for some constant c,

$$N(f^{(2)}) \geq \frac{N^2(f)}{c \cdot n}$$

Proof: Consider an optimal cover of $M_{f^{(2)}}$, with $k = N(f^{(2)})$ monochromatic rectangles, denoted by $R_1, R_2, \ldots, R_k$. We show how to cover M_f with m monochromatic rectangles, where $m^2 \leq c \cdot n \cdot N(f^{(2)})$ for some constant c. This implies that $N^2(f) \leq c \cdot n \cdot N(f^{(2)})$.

Consider the following matrix A (this is *not* $M_{f^{(2)}}$): each row of A corresponds to a legal input $(x_1, y_1) \in D$ and each column to a legal $(x_2, y_2) \in D$. Every entry $((x_1, y_1), (x_2, y_2))$ of A contains an element t in $\{1, 2, \ldots k\}$ such that $((x_1, x_2), (y_1, y_2))$ belongs to R_t. (If $((x_1, x_2), (y_1, y_2))$ belongs to more than one rectangle, then we choose one of them arbitrarily). The proof of [9, Theorem 1] provides the following claim:

Claim 1: Let A be a $\ell \times \ell$ matrix whose entries assume k values. Let k_1 be the minimal size of a set $T \subseteq \{1, 2, \ldots k\}$ that covers all the rows of A. That is, for every row i there exists a column j such that the value $A_{i,j}$ belongs to T. Similarly, let k_2 be the minimal size of a set that covers all the columns. Then $k_1 \cdot k_2 \leq c' \cdot \log \ell \cdot k$.

Apply this claim to the matrix A described above, and assume without loss of generality that $k_1 \leq k_2$; we get that $k_1^2 \leq c \cdot n \cdot k$. Let T be a set of k_1 values that covers the rows. We now prove that this implies that M_f can be covered with k_1 monochromatic rectangles.

Associate with every non-$\perp$ entry (x, y) in M_f an element of T that appears in the row (x, y) of A (if there is more than one possibility, then choose one arbitrarily). Now we extend this to rectangles in the obvious way. Namely, for every $t \in T$ the rectangle R'_t includes every (x, y) with value t, and if (x, y) and (x', y') are in R'_t then also (x', y) and (x, y') are in R'_t.

Clearly, these are k_1 rectangles and they cover M_f. What we still have to prove is that these rectangles are monochromatic. Thus, we must show that if (x, y) and (x', y') were given the value t, then $f(x, y) = f(x', y')$ and furthermore, that if f is defined on (x, y') or (x', y) then it gets the same value. By the construction, if (x, y) and (x', y') both have the value t, then there exist x_2, y_2, x'_2 and y'_2 such that both $((x, x_2), (y, y_2))$ and $((x', x'_2), (y', y'_2))$ belong to R_t. Since R_t is monochromatic, then we can associate with R_t a vector (z_1, z_2) with whom all the non-$\perp$ coordinates "agree". This implies that $f(x, y) = f(x', y')$. In addition, since R_t is a rectangle then it also contains $((x, x_2), (y', y'_2))$ and $((x', x'_2), (y, y_2))$ which implies that for both (x, y') and (x', y) if f is defined on them then it has the same value.

To conclude, we can cover M_f with no more than $\sqrt{c \cdot n \cdot N(f^{(2)})}$ monochromatic rectangles, which completes the proof of the theorem. $\square$

Again, the above theorem (using [9]) can be generalized to prove the following:

Corollary 6: Let f be a function defined over $D_f \subseteq \{0,1\}^n \times \{0,1\}^n$, and let g be a function defined over $D_g \subseteq \{0,1\}^n \times \{0,1\}^n$. Then,

$$N(f \times g) \geq \frac{N(f) \cdot N(g)}{c \cdot n}$$

It follows that $N(f^{(\ell)}) \geq \frac{N^\ell(f)}{(cn)^\ell}$. By [1], we have that $C(f) \leq 2 \log^2 N(f)$. All together, we get

$$\begin{aligned} C(f^{(\ell)}) &\geq \log N(f^{(\ell)}) \\ &\geq \ell \log N(f) - \ell \log n - O(\ell) \\ &\geq \ell \cdot \left(\sqrt{C(f)/2} - \log n - O(1) \right) \end{aligned}$$

Thus, we have the desired lower bound:

Corollary 7: Let $f : D \to \{0,1\}$, where $D \subseteq \{0,1\}^n \times \{0,1\}^n$. Then, $C(f) \geq \overline{C}(f) \geq \sqrt{C(f)/2} - \log n - O(1)$.

6 A Function With Low Amortized Randomized Complexity

In this section we consider amortized *randomized* communication complexity. Clearly, $\overline{C}_R(f) \leq \overline{C}(f) \leq n$. However, unlike the deterministic case, we do not know whether $\overline{C}_R(f) \leq C_R(f)$ for all functions f. The best we can say is that $\frac{1}{\ell} \cdot \overline{C}_R(f^{(\ell)})$ is $O(C_R(f) \cdot \log \ell)$ (the $\log \ell$ factor seems to be needed, since we require the protocols for computing $f^{(\ell)}$ to be correct with high probability on *all* ℓ instances simultaneously). For specific functions we can do much better. We show that the amortized complexity of the *identity* function, with respect to randomized protocols, is $\overline{C}_R(ID) = O(1)$. (It is well known that $C_R(ID) = \Theta(\log n)$ [21].)

For simplifying the presentation of the protocols we assume that the two parties have a way of agreeing on a random string with no cost in communication. This can be thought as protocols in the *public-coins* model. After presenting the protocols we describe how the parties can agree on such strings while preserving both the communication complexity and the correctness of the protocols.

The following protocol computes the identity function on a single pair of inputs:

- The parties agree on a random string $b \in \{0,1\}^n$.
- P_1 computes $\langle b, x \rangle$, the inner product of b and $x \pmod 2$, and P_2 computes $\langle b, y \rangle$.
- The parties exchange the bits $\langle b, x \rangle$ and $\langle b, y \rangle$. If the bits are equal they output "equal" ($x = y$), otherwise they output "not-equal" ($x \neq y$).

The number of bits exchanged in the protocol is $O(1)$. If $x = y$ it is always correct, while if $x \neq y$ it is correct with probability $\frac{1}{2}$ (which can be improved to any other constant advantage while preserving the $O(1)$ complexity).

Suppose now that P_1 and P_2 wish to compute the identity function on

$$(x_1, y_1), (x_2, y_2), \ldots, (x_\ell, y_\ell).$$

Consider the protocol where P_1 and P_2 amortize the first step in the above protocol while exchanging the bits $\langle b, x_i \rangle$ and $\langle b, y_i \rangle$, for all $1 \leq i \leq \ell$. Such a protocol gives a "good" success probability for computing each of the $f(x_i, y_i)$ separately, while what we want is a "good" probability of computing f on all ℓ instances simultaneously. A possible idea is to decrease the error probability on each (x_i, y_i) to $\frac{1}{poly(\ell)}$ by choosing $k = O(\log \ell)$ vectors b_i's. Formally,

Protocol *multi_compare*:

1. The parties agree on k random strings $b_1, b_2, \ldots, b_k \in \{0,1\}^n$.
2. For $i = 1, 2, \ldots, k$:
   (a) P_1 computes
   $$u_i = \langle b_i, x_1 \rangle, \langle b_i, x_2 \rangle, \ldots, \langle b_i, x_\ell \rangle$$
   P_2 computes
   $$v_i = \langle b_i, y_1 \rangle, \langle b_i, y_2 \rangle, \ldots, \langle b_i, y_\ell \rangle$$
   (b) The parties exchange the vectors u_i and v_i (each of them is an ℓ-bit string).
3. The output for the j-th pair (x_j, y_j) is "equal" ($x_j = y_j$) if and only if for every i the j-th bits of u_i and v_i are equal.

The only problem with this protocol is that if $k = O(\log \ell)$ then the communication complexity of this protocol is $O(\ell \log \ell)$. This complexity is more than what we aim for.

The main idea is the following: even if a vector b_i does not recognize all the pairs such that $x_j \neq y_j$, we expect that it does recognize a constant fraction of them. If at each time that the parties recognize such a pair, they replace it by $x_j = y_j = 0^n$ (a string of zeros), then the expected Hamming distance between the vectors

u_i and v_i in the above protocol decreases from round to round. We use this property by presenting a procedure $exchange(u, v)$ that enables the parties to exchange u_i and v_i (step (2b)) in a cost that depends on the Hamming distance between the vectors; Namely, the smaller the Hamming distance, the lower the communication complexity. This will give us the desired complexity.

We start with a simple case where the parties P_1 and P_2 receive, in addition to the input vectors $u, v \in \{0,1\}^\ell$ respectively, a bound d such that u and v are promised to be at Hamming distance at most d. The following *deterministic* protocol $exchange_d(u, v)$ enables each party to learn the value of the other party, by exchanging $O(\log \binom{\ell}{d})$ bits (we assume that $d \leq \ell/4$, otherwise the parties simply exchange their inputs). The protocol is due to Brandman, El-Gamal and Orlitsky (in [14]), Witsenhausen and Wyner [20] and Karchmer and Wigderson [8]:

Protocol $exchange_d(u, v)$**:**

- The parties consider the graph with 2^ℓ nodes corresponding to the strings in $\{0,1\}^\ell$ and edges between nodes which are at Hamming distance at most $2d$. The parties fix a coloring of the graph. (An effective coloring can be constructed using linear error correcting codes such as BCH.)

- P_1 sends P_2 the color of u and P_2 sends the color of v under the coloring. Since the Hamming distance between u and v is bounded by d and since there is at most one member of every color class at distance d from v (as we have a legal coloring of vectors with Hamming distance $\leq 2d$) then P_2 can identify u. Similarly, P_1 can identify v.

The degree of every node in this graph is less than $2d \cdot \binom{\ell}{2d}$. Therefore there exists a coloring of the graph with that many colors. Since the communication in this protocol consists of names of colors then $O(\log \binom{\ell}{2d}) = O(\log \binom{\ell}{d})$ bits are communicated.

The above protocol assumes that we have an upper bound on the Hamming distance between u and v. If u and v do not satisfy this assumption, then the protocol may fail. The next (randomized) protocol generalizes the above protocol for the case that such a bound d is not known, but the expected number of bits exchanged is still only $O(\log \binom{\ell}{\Delta})$ bits, where Δ is the actual Hamming distance between u and v.

Protocol $exchange(u, v)$**:**

1. The parties agree on k random "test strings" $c_1, c_2, \ldots, c_k \in \{0,1\}^\ell$.
2. For $d = 2^1, 2^2, 2^3, \ldots, 2^{\log \ell}$
   (a) P_1 and P_2 engage in $exchange_d(u, v)$. Denote the output of P_1 by v' and the output of P_2 by u'.
   (b) *Test step:* P_1 and P_2 test whether $u' = u$ by comparing the inner product of the "test strings" $c_1, c_2, \ldots c_k$ with u and u'; this is done in a bit by bit manner, quitting early if they discover an error and going to the next d. If all the k bits are equal the protocol terminates (i.e., the parties assume that d is correct, and therefore $u' = u$ and $v' = v$).

The number of bits required in step (2a) is $O(\log \binom{\ell}{d})$ if $d \leq \ell/4$ and $O(\ell)$ otherwise. If $u' \neq u$ then the *expected* number of bits exchanged in the test step is $O(1)$. If $u' = u$ then the number of bits exchanged in the test step is k, however this happens only once. Therefore, the expected number of bits communicated is $O(k + \sum_{i=1}^{\log \Delta} \log \binom{\ell}{2^i})$ which is $O(k + \log \binom{\ell}{\Delta})$ if $\Delta \leq \frac{\ell}{2}$ and $O(k + \log \binom{\ell}{\ell/2})$ otherwise. The error probability in each round is bounded by 2^{-k} and therefore the total error probability is bounded by $\log \ell \cdot 2^{-k}$.

To obtain our protocol, we have to replace step (2b) of the protocol *multi_compare*, by the procedure $exchange(u, v)$ described above. (As mentioned before, after the parties identify that $x_j \neq y_j$, they set $x_j = y_j = 0^n$.)

The analysis of the protocol is as follows: let d_i be the random variable which is the number of indices $1 \leq j \leq \ell$ such that $x_j \neq y_j$ but $\langle b_1, x_j \rangle = \langle b_1, y_j \rangle, \ldots, \langle b_{i-1}, x_j \rangle = \langle b_{i-1}, y_j \rangle$. I.e.

d_i is a bound on the distance between u_i and v_i. The expected number of bits exchanged is thus $O(\sum_{i=1}^{k}\left(k+\log\binom{\ell}{d_i}\right))$. The expected value of d_i is bounded by $\ell \cdot c^{-i}$ for some fixed constant $c > 1$. Thus the expected number of bits exchanged is $O(\sum_{i=1}^{k}\left(k+\log\binom{\ell}{\ell c^{-i}}\right))$ which is $O(k^2+\ell)$. If k is $O(\log \ell)$, then the expected number of bits communicated is $O(\ell)$. For the correctness, if $x_j \neq y_j$ then only with probability 2^{-k} for all i we have $\langle b_i, x_j\rangle = \langle b_i, y_j\rangle$. Therefore the probability that for some j, all the i's satisfy $\langle b_i, x_j\rangle = \langle b_i, y_j\rangle$ is bounded by $\frac{\ell}{2^k}$. In addition there is a probability of $\frac{\log \ell}{2^k}$ to fail each time we invoke protocol $exchange(u, v)$. Thus the total error probability is bounded by $\frac{\ell + k\log \ell}{2^k}$. Therefore, $k = O(\log \ell)$ is enough for achieving the needed advantage. (We can take k to be higher, i.e. $O(\sqrt{\ell})$, and thus significantly improve the success probability while still have $O(\ell)$ communication bits.)

The above protocols assume that the parties have a way to agree on random strings (i.e., the b_i's and c_i's) with no cost in communication. Here we describe how this is implemented while preserving the correctness and the communication complexity of the protocols. We start with a description of how to agree on a single string b_i: A collection of vectors $B_m \subset \{0,1\}^m$ is called *ε-biased* if

1. The number of strings in B_m is $poly(m)$ (and thus each of them can be represented by $O(\log m)$ bits).
2. Every $x \in \{0,1\}^m$ satisfy $Pr_{b\in B_m}(\langle b, x\rangle = 0) = \frac{1}{2} \pm \varepsilon$.

In [12] the existence and construction of such sets are shown. For our purposes it is sufficient to take ε to be say $1/4$. Fix B_n an ε-biased probability space. P_1 selects $b_i \in B_n$ by choosing $\log |B_n|$ random bits and sends them to P_2. Clearly, if $x = y$ then $\langle b_i, x\rangle = \langle b_i, y\rangle$ while if $x \neq y$ then $\langle b_i, x\rangle \neq \langle b_i, y\rangle$ with probability at least $1/4$.

When we need to pick k strings $b_1, b_2, \ldots, b_k$ we can do that as described above using $O(k \cdot \log n)$ bits. This can be improved to $O(k+\log n)$ by sampling the b_i's via a random walk in an expander à la Ajtai, Komlòs and Szemerèdi [2] (in such a case the b_i's are not independent): The elements of B_n are mapped to nodes of a constant degree expander G. Then, a random walk of length k in G is generated, and the vectors $b_1, b_2, \ldots, b_k$ are the vectors corresponding to the nodes of the walk. The number of bits required to specify the walk is $O(\log |B_n| + k)$ which is $O(\log n + k)$. (See [12] for details.) As before P_1 selects the random bits and sends them to P_2, so that they both agree on the same sequence. If $x \neq y$ then the probability that $\langle b_i, x\rangle = \langle b_i, y\rangle$ for all $1 \leq i \leq k$ goes down exponentially in k.

The strings $c_1, c_2, \ldots, c_k$ are selected similarly in B_ℓ using $O(k + \log \ell)$ bits. Note however that step (1) in protocol $exchange(u, v)$ should not be repeated, i.e. $c_1, c_2, \ldots c_k$ are chosen once and for all at the beginning of the protocol *multi_compare*.

To conclude, we have a randomized protocol for computing the ID function on ℓ instances with constant advantage and expected complexity of $O(\ell+\log n)$, which is $O(\ell)$, for ℓ sufficiently large. With a "small" additional error the protocol can be converted to a protocol that uses $O(\ell)$ bits in the *worst case*. To conclude, we proved

Theorem 8: $\overline{C}_R(ID) = O(1)$.

Remark: Newman [13] has considered the public-coins model. He showed that $C_R(f) = O(C_{pub}(f) + \log n)$, which in particular implies

$$C_R(f^{(\ell)}) = O(C_{pub}(f^{(\ell)}) + \log \ell n).$$

Clearly,

$$C_R(f^{(\ell)}) \geq C_{pub}(f^{(\ell)}) = O(\ell \cdot \log \ell \cdot C_{pub}(f)).$$

All together we have the following:

Theorem 9: Let $f : D \to \{0,1\}$. Then

1. $\overline{C}_R(f) = \Theta(\overline{C_{pub}}(f))$.
2. For every sufficiently large ℓ, $\frac{1}{\ell} \cdot C_R(f^{(\ell)}) = O(\log \ell \cdot C_{pub}(f))$.

7 Acknowledgments

We are grateful to Noam Nisan for many joyful discussions concerning the subject of this paper. In particular, Noam provided an alternative proof for the results of section 5 and clarified the issues concerning the relationship between the public and private coin models.

We thank Mauricio Karchmer and Avi Wigderson for raising the question and for helpful discussions, and Amos Beimel, Benny Chor, Alon Orlitsky and Steve Ponzio for many interesting comments on earlier versions of this paper.

References

[1] Aho A., J. Ullman, and M. Yannakakis, "On Notions of Information Transfer in VLSI Circuits", *Proc. of 15th STOC*, 1983, pp. 133-139.

[2] Ajtai M., J. Komlòs, and E. Szemerèdi, "Deterministic Simulation in LOGSPACE", *Proc. of 19th STOC*, 1987, pp. 132-140.

[3] Bshouty, N. H., "On The Extended Direct Sum Conjecture", *Proc. of 21th STOC*, 1989, pp. 177-185.

[4] Fredman M., J. Komlòs, and E. Szemerèdi, "Storing A Sparse Table with O(1) Access Time", *JACM*, Vol 31, 1984, pp. 538–544.

[5] Galibati G., and M. J. Fischer, "On The Complexity of 2-Output Boolean Networks", *TCS*, Vol 16, 1981, pp. 177–185.

[6] Karchmer M., R. Raz, and A. Wigderson, "On Proving Super-Logarithmic Depth Lower Bounds via the Direct Sum in Communication Complexity", *Proc. of 6th IEEE Structure in Complexity Theory*, 1991, pp. 299–304.

[7] Karchmer M., and A. Wigderson, "Monotone Circuits for Connectivity Require Super-Logarithmic Depth", *Proc. of 20th STOC*, 1988, pp. 539-550.

[8] Karchmer M., and A. Wigderson, private communication.

[9] Linial N., and U. Vazirani, "Graph Products and Chromatic Numbers", *Proc. of 30th FOCS*, 1989, pp. 124-128.

[10] Lovász, L., "Communication Complexity: A Survey", in *Paths, Flows, and VLSI Layout*, edited by B. H. Korte, Springer Verlag, Berlin New York, 1990.

[11] Mehlhorn, K., and E. Schmidt, "Las-Vegas is better than Determinism in VLSI and Distributed Computing", *Proc. of 14th STOC*, pp. 330-337, 1982.

[12] Naor J., and M. Naor, "Small-Bias Probability Spaces: Efficient Constructions and Applications", *Proc. of 22nd STOC*, 1990, pp. 213-223.

[13] Newman, I., private communication.

[14] Orlitsky, A., "Communication Issues in Distributed Communication", PhD thesis, Stanford University, 1986.

[15] Orlitsky, A., "Two Messages are Almost Optimal for Conveying Information", *Proc. of 9th PODC*, 1990, pp. 219-232.

[16] W. Paul, "Realizing Boolean Function on Disjoint Sets of Variables", *TCS* 2, 1976, pp. 383-396.

[17] Raz R., and A. Wigderson, "Monotone Circuits for Matching Require Linear Depth", *Proc. of 22nd STOC*, 1990, pp. 287-292.

[18] Q. F. Stout, "Meshes with multiple buses", *Proc. of 27th FOCS*, 1986, pp. 264-273.

[19] Witsenhausen, H. S., "The Zero-Error Side Information Problem and Chromatic Numbers", *IEEE Transactions on Information Theory*, 1976, pp. 592-593.

[20] Witsenhausen, H. S. and A. D. Wyner, "Interframe Coder for Video Signals", *United States Patent number 4,191,970*, 1980.

[21] Yao, A. C., "Some Complexity Questions Related to Distributed Computing", *Proc. of 11th STOC*, 1979, pp. 209-213.