

Communication vs. Computation

Prahladh Harsha¹, Yuval Ishai², Joe Kilian³, Kobbi Nissim⁴, and S. Venkatesh⁵

¹ Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139, USA, prahladh@mit.edu.*

² Computer Science Department, Technion, Haifa 32000, Israel, yuvali@cs.technion.ac.il.

³ NEC Laboratories America Inc, Princeton, NJ 08540, USA, joe@nec-labs.com.

⁴ Microsoft Research, SVC. 1065 La Avenida, Mountain View, CA 94043, USA, kobbi@microsoft.com.**

⁵ Computer Science Department, University of Victoria, Victoria, BC, Canada V8W 3P6, venkat@cs.uvic.ca.***

Abstract. We initiate a study of tradeoffs between communication and computation in well-known communication models and in other related models. The fundamental question we investigate is the following: Is there a computational task that exhibits a strong tradeoff behavior between the amount of communication and the amount of time needed for local computation?

Under various standard assumptions, we exhibit boolean functions that show strong tradeoffs in the following computation models: (1) two-party randomized communication complexity; (2) query complexity; (3) property testing. For the model of deterministic communication complexity, we show a similar result relative to a random oracle.

Finally, we study a time-degree tradeoff problem that arises in arithmetization of boolean functions, and relate it to time-communication tradeoff questions in multi-party communication complexity and in cryptography.

1 Introduction

A Motivating Riddle. Consider the following multi-party communication game. Fix a finite field F and let M be a $n \times k$ matrix over F . The columns of F are assigned to k players so that each player j knows all columns of M *except* the j th. (This is known as the “input on the forehead” model [CFL83].) The players’ goal is to compute the product of the n row sums, namely the function

$$\text{PS}(M) = \prod_{i=1}^n \sum_{j=1}^k M_{i,j},$$

by means of simultaneously sending messages to an external referee. This can be easily done by having the entire matrix M sent to the referee (e.g., letting P_1 send

* Research done while the author was at NEC Laboratories America.

** Research done while the author was at NEC Laboratories America.

*** Research done while the author was at MPI for Informatik, Germany.

the second column and P_2 the remaining columns). The goal is to minimize the *communication complexity*, measured as the length of the longest message sent. A closely related problem was studied in [BGKL03]. When $k > n$ (say, $k = n + 1$) our problem admits the following simple solution, implicit in [BGKL03]. Write $\text{PS}(M)$ as the sum of k^n terms, where each term is a product involving a single entry from each row of M . Since there are more players than rows, for each such term there is a player holding all of its values. Hence, one can assign each term to some player who knows its value, and have each player send the sum of all terms assigned to it. The referee can then recover $\text{PS}(M)$ by simply adding up the k field elements it received. While this protocol is very efficient in communication, the combined *computation* of the players is exponential in n . Note that if one uses the natural greedy strategy of assigning each term to the *first* player to which it can be assigned, then player $n + 1$ will need to compute the *permanent* of an $n \times n$ sub-matrix of M , a $\#P$ -hard problem.⁶ Thus, a natural question is the following:

Does the function $\text{PS}(M)$ admit a protocol in which (1) each player only sends a single element of F ; and (2) the local computation of each player is polynomial in n ?

A negative answer seems likely in light of the failure of the natural term assignment strategy. It also seems reasonable that for *any* valid way of assigning the k^n terms to the players, some player will be forced to compute a hard function. Thus, this problem looks like a good candidate for a *time-communication tradeoff*: it requires little time to compute when there is no limit on the communication complexity, requires little communication when there is no limit on the time complexity, but seems to defy solutions that are simultaneously efficient with respect to both complexity measures.

Quite surprisingly, it turns out that the answer to the above question is “yes”. (The impatient reader can skip to Section 5 for a solution to the riddle.) Thus, this particular problem does not exhibit the time-communication tradeoff that was initially suspected. However, this question served as the original motivation for this work, which explores the existence of similar kinds of tradeoffs in related contexts.

1.1 Problem Description

Let $f : X \times Y \rightarrow Z$ be an arbitrary function of two inputs. In the two-party communication model of Yao [Yao79], there are two players A and B . A is given $x \in X$, B is given $y \in Y$ and they need to compute $z = f(x, y)$ by communicating with each other. In any communication protocol designed for f , there are three useful measures of complexity:

⁶ Even if F has characteristic 2, in which case the permanent can be efficiently computed, it is not clear that the computation of (say) the middle player can be made efficient.

- **Communication complexity:** The total number of bits exchanged between A and B ;
- **Time complexity:** The amount of time needed by A and B for local computation;
- **Round complexity:** The number of messages exchanged by A and B .

Given any two of these three complexity measures, it is natural to ask if there are tasks which exhibit a tradeoff between them. The question of rounds vs. computation does not arise in the two-party model, as the simple protocol in which A send his entire input over to B is optimal with respect to both measures.⁷ Tradeoffs between round complexity and communication complexity have been well studied (see below). In this paper, we initiate the study of the remaining question: proving tradeoffs between communication and local computation. Specifically, our main goal is to find functions f such that: (1) f can be efficiently computed given both its inputs, i.e., given no restriction on the communication; (2) f has a protocol with low communication complexity given no restriction on the computation; and (3) there is no protocol for f which simultaneously has low communication and efficient computation.

1.2 Related work

Papadimitriou and Sipser [PS84] first discussed the problem of showing tradeoffs between rounds of communication and communication complexity. For any fixed k , they proposed a boolean function p_k called the *pointer chasing problem* that has a k -round protocol with $O(\log n)$ bits of communication. They conjectured that its communication complexity is at least linear if only $k - 1$ rounds are allowed. In other words, p_k shows a strong tradeoff behavior between rounds and communication complexity. This conjecture was proved in a series of papers [PS84,DGS87,NW93].

Additional complexity measures which are not considered in this work are *space* complexity and *randomness* complexity. Tradeoffs between space and communication were considered by Beame et al. [BTY94]. Tradeoffs between randomness and communication were studied by Canetti and Goldreich [CG93].

1.3 Our Results

Our first result is a strong time-communication tradeoff for a boolean function in the two-party randomized communication model.

Randomized communication model. Suppose that there is a UP relation R such that the search problem corresponding to R is not in $\text{BPTIME}[2^{O(T(n))}]$. (This would follow from the existence of a one-way permutation secure against

⁷ However, this question does make sense in a cryptographic setting when players need to compute a function of their inputs without revealing their inputs to each other. Such a tradeoff question is addressed in Section 5.3.

a $2^{O(T(n))}$ bounded adversary.) Then, there is an efficiently computable boolean function f_R with the following properties. If Alice and Bob are computationally unbounded, then there is an $O(\log n)$ -bit 1-round randomized protocol that computes f_R . But if Alice and Bob are computationally bounded, then any randomized protocol for f_R , even with multiple rounds, will require $\Omega(T(n))$ bits of communication (see Section 3).

As a corollary we get the following strong separation result. Let F_c denote the class of functions $f(x, y) \in \text{PTIME}$ such that the randomized communication complexity of f is bounded by c . Similarly, let F_c^{poly} be the functions $f(x, y) \in \text{PTIME}$ such that $f(x, y)$ is computable by polynomial-time parties with communication c . Then there is an explicit boolean function f in $F_{\log n} \setminus F_{T(n)}^{\text{poly}}$ for $T(n)$ as above.

Deterministic communication model. Obtaining similar tradeoff results for the deterministic two-party model appears to be much harder. We show a strong tradeoff result relative to a random oracle. Specifically, let L be a random sparse language. Then, with probability 1 over choice of L , there is a boolean function f_L (efficiently computable relative to L) with the following properties. There is a *deterministic* communication protocol for f_L with, say, $O(\log^2 n)$ bits of communication if both Alice and Bob are computationally unbounded with oracle access to L . However, any protocol in which Alice and Bob are computationally bounded will require $\Omega(n)$ bits of communication, even with oracle access to L . We defer the proof of this tradeoff to the full version of the paper [HIKNV].

Query complexity and property testing. Our next results prove tradeoffs in related models like the query complexity model and the property testing model. In these models, information is stored in the form of a table and the queries are answered by bit-probes to this table. We view the probes as communication between the stored table and the query scheme (or the tester), and the computation of the query scheme (or the tester) as the local computation. We show that: (a) Under a cryptographic assumption, there exists a language L such that, on inputs of length n , a query scheme with unlimited computation makes $O(\log n)$ queries while a query scheme with efficient local computation requires $\Omega(n^\varepsilon)$ queries for some fixed $\varepsilon < 1$; (b) assuming $\text{NP} \not\subseteq \text{BPP}$, given any $\varepsilon > 0$, there exists a property P such that, on inputs of length n , a computationally unbounded tester will require only n^ε bits to check if the input satisfies the property or is far from satisfying it. On the other hand, a computationally bounded tester will require $n^{1-\varepsilon}$ bits. We only provide the proof of the tradeoff in the query complexity model (see Section 4) and defer the proof of the tradeoff in the property testing model to the full version of the paper [HIKNV].

Natural tradeoff questions. In addition to proving the *existence* of tradeoffs in various contexts, we also put forward several concrete *natural* tradeoff questions and relate them to each other. We propose three different tradeoff questions arising in different contexts: arithmetization of boolean functions, multi-party

communication, and cryptography. We relate them by showing that a “positive” resolution of the first would imply a solution to the second, which in turn would imply a solution to the third. Hence, the cryptographic application may serve as an additional motivation for studying the other two. For want of space, we defer the entire discussion on these natural tradeoff questions to the full version of the paper [HIKNV].

2 Preliminaries

In this section, we describe the communication complexity model, a formal definition of the problem we consider and the notion of UP relations.

2.1 The Communication Complexity Model [Yao86]

Let X , Y and Z be arbitrary finite sets and $f : X \times Y \rightarrow Z$ be an arbitrary function. There are two players, Alice and Bob who wish to evaluate $f(x, y)$ for $x \in X$ and $y \in Y$. However, Alice only knows x and Bob only knows y . To evaluate the function, they communicate with each other according to some fixed protocol P in which they send messages to each other.

The cost of a protocol P on an input (x, y) is the number of bits exchanged by Alice and Bob when Alice is given x and Bob is given y . The cost of a protocol P is the worst case cost of P over all inputs (x, y) . The (deterministic) communication complexity of f is the minimum cost of a protocol that computes f .

If Alice and Bob are allowed access to random coin tosses and their messages depend also on the result of the coin tosses besides their input and the communication so far, we say that the protocol P is randomized. The randomized communication complexity of a function f is the minimum cost of a randomized protocol that computes f with error at most $\frac{1}{4}$ on any input (x, y) . The error is over the internal coin tosses of the protocol.

2.2 Tradeoffs

We now describe formally our tradeoff problem in the two-party communication complexity model. Similar definitions can be given for other models we consider. Our goal is to find a boolean function $f : X \times Y \rightarrow \{0, 1\}$ with the following properties:

- $f(x, y)$ can be computed efficiently, that is in polynomial time, if both the inputs $x \in X$ and $y \in Y$ are given.
- f has very efficient communication protocols, that is, protocols with communication complexity $(\log n)^c$ for some c .
- There is no protocol for f which is simultaneously communication and computation efficient. In other words, any protocol in which Alice and Bob use only polynomial time for local computation requires almost linear number of bits of communication in the worst case.

2.3 UP Relations

Definition 2.1. A relation $R \subseteq \Sigma^* \times \Sigma^*$ is said to be a UP relation (with witness size n^k) if

1. there exists a deterministic Turing machine that decides the language $\{(x, w) \mid (x, w) \in R\}$ in polynomial time.
2. for every x , there exists at most one w such that $(x, w) \in R$ and furthermore, this w satisfies $|w| = |x|^k$. We denote this w , if it exists, by $w(x)$.

The search problem corresponding to R is the problem of finding w such that $R(x, w)$ holds, given x .

We will assume the existence of UP relations for which the corresponding search problem is very hard. Such an assumption is standard in cryptography since the existence of strong one-way permutations implies the existence of such hard UP relations. More formally,

Definition 2.2. We say that a UP relation R is $T(n)$ -hard if no probabilistic algorithm running in time $2^{O(T(n))}$ solves the search problem corresponding to R .

3 Tradeoffs in the Two-Party Communication Complexity Model

We start with the definition of the boolean function we consider.

Definition 3.1. Let $R \subseteq \{0, 1\}^* \times \{0, 1\}^*$ be a UP relation with witness size n^k . Consider the 2-player (Alice and Bob) boolean function $f_R : \{0, 1\}^{n+n^k} \times \{0, 1\}^{n^k} \rightarrow \{0, 1\}$.

Alice's input: $(x, z) \in \{0, 1\}^n \times \{0, 1\}^{n^k}$; Bob's input: $w \in \{0, 1\}^{n^k}$

$$f_R((x, z), w) = \begin{cases} \langle z, w \rangle & \text{if } R(x, w) \text{ holds} \\ 0 & \text{otherwise} \end{cases}$$

where $\langle a, b \rangle$ denotes the inner product of a, b modulo 2.

Theorem 3.2. Let R be a $T(n)$ -hard UP relation. Then, the predicate f_R has the following properties.

1. f_R is computable in polynomial time.
2. There exists a randomized protocol that computes f_R with $O(\log n)$ -bit communication.
3. If Alice and Bob are computationally bounded, then any randomized protocol for f_R , even with multiple rounds, will require $\Omega(T(n))$ bits of communication.

Proof. Observe that f_R can be computed efficiently given both its inputs. We just need to check that $R(x, w)$ holds and if so, output $\langle z, w \rangle$.

Lemma 3.3. *If Alice is computationally unbounded, then there exists a randomized protocol that computes f_R with $O(\log n)$ -bit communication.*

Proof. Alice computes the unique w such that $R(x, w)$ holds. Alice and Bob then engage in an “equality” protocol⁸ to check that Bob’s input equals w . If so, she computes and sends Bob the answer $\langle z, w \rangle$. \square

The following lemma demonstrates that such a communication-efficient protocol is unlikely when Alice and Bob are computationally bounded. In fact, it is sufficient for the proof that only Alice is computationally bounded. Bob is allowed to be computationally unbounded.

Lemma 3.4. *Suppose there exists a $b(n)$ -bit communication randomized multi-round protocol Π that computes f_R involving Alice whose running time is at most $T_A(n)$, then there exists a randomized algorithm that solves the search problem corresponding to R in time $\text{poly}(n, 2^{b(n)}) \cdot T_A(n)$.*

Proof. For the rest of the argument, we assume that for any x , w is the unique w such that $R(x, w)$ holds, denoted by $w(x)$. Hence, for our purposes, $f_R((x, z), w) = \langle z, w \rangle$.

Our goal is to relate the search problem of computing w given x to the problem of computing $\langle z, w \rangle$ with a low communication protocol. Our approach is to convert a low communication protocol into an efficient oracle that computes $\langle z, w \rangle$ with some advantage over random guessing. Given such an oracle, we can then use the Goldreich-Levin reconstruction algorithm to compute a small number of candidates for w . More precisely, we create a “small” set of oracles, one of the oracles computes $\langle z, w \rangle$ with some nontrivial advantage. We try each oracle by exhaustive search, and use the fact that we can recognize the correct w .

Converting protocols into oracles

Let \mathcal{T} be a transcript. For simplicity, we assume Alice outputs $f_R((x, z), w)$ as its final bit; this convention increases the size of the transcript by at most 2 bits. Thus, \mathcal{T} includes a “guess” as to $\langle z, w \rangle$. We define the probabilistic oracle $A_{\mathcal{T}}(x, z)$ for computing $\langle z, w \rangle$, as follows.

Algorithm $A_{\mathcal{T}}$ (Input: $(x, z) \in \{0, 1\}^n \times \{0, 1\}^{n^k}$).

Simulate the protocol Π from Alice’s end. Whenever a message from Bob is required, use the transcript \mathcal{T} to obtain the corresponding message. If at any point the message generated by Alice according to the protocol Π disagrees with the contents of the transcript \mathcal{T} , abandon the protocol and output a random bit b . Otherwise, follow the protocol to the end and output the bit b generated by the protocol Π .

⁸ Recall that the randomized communication complexity of equality is $O(\log n)$.

First we define our notation for the advantage of Π and $A_{\mathcal{T}}$ in guessing $\langle z, w \rangle$.

Definition 3.5. Let $x \in \{0, 1\}^n$, $w = w(x)$ and z be distributed uniformly. We define $\text{ADV}(\Pi, x)$ by

$$\text{ADV}(\Pi, x) = \Pr[\text{Alice outputs } \langle z, w \rangle] - \Pr[\text{Alice doesn't output } \langle z, w \rangle],$$

where Alice and Bob run Π with respective inputs (x, z) and w , and the probability is taken over the choice of z and over the coin tosses of Alice and Bob. We define $\text{ADV}(A_{\mathcal{T}}, x)$ analogously. Fixing x and a transcript \mathcal{T} , we define $\text{ADV}(\Pi, x, \mathcal{T})$ by

$$\text{ADV}(\Pi, x, \mathcal{T}) = \Pr[\mathcal{T} \text{ occurs and Alice outputs } \langle z, w \rangle] - \Pr[\mathcal{T} \text{ occurs and Alice doesn't output } \langle z, w \rangle].$$

Note that the only contribution to $A_{\mathcal{T}}$'s advantage is by events in which \mathcal{T} occurs, hence we do not bother to define $\text{ADV}(A_{\mathcal{T}}, x, \mathcal{T})$. It follows from the definitions that,

$$\text{ADV}(\Pi, x) = \sum_{\mathcal{T}} \text{ADV}(\Pi, x, \mathcal{T}). \quad (1)$$

Since the protocol Π computes f_R correctly, it holds that $\text{ADV}(\Pi, x) \geq \frac{1}{2}$ for every x . Since there are at most $2^{2b(n)}$ possible transcripts \mathcal{T} , it follows from Equation (1) that for every $x \in \{0, 1\}^n$, there exists a transcript \mathcal{T}^* ,

$$\text{ADV}(\Pi, x, \mathcal{T}^*) \geq \frac{1}{2^{2b(n)+1}} \quad (2)$$

Let $\rho_w(\mathcal{T})$ be the probability that Bob's coins are consistent with \mathcal{T} . Note that $\rho_w(\mathcal{T})$ is independent of z . It can easily be verified from the definitions that

$$\text{ADV}(\Pi, x, \mathcal{T}) = \text{ADV}(A_{\mathcal{T}}, x) \rho_w(\mathcal{T}). \quad (3)$$

Since $0 \leq \rho_w(\mathcal{T}) \leq 1$, it follows from Equation (2) that

$$\text{ADV}(A_{\mathcal{T}^*}, x) \geq \frac{1}{2^{2b(n)+1}}. \quad (4)$$

Set $\varepsilon = \frac{1}{2^{2b(n)+1}}$. Now we run the Goldreich-Levin algorithm GL (See Theorem 3.6) with parameters n, ε , oracle access to $A_{\mathcal{T}^*}(x, \cdot)$ and predicate $R(x, \cdot)$.

Theorem 3.6 (Goldreich-Levin [GL89]). *There exists a randomized algorithm GL with oracle access to a function and a predicate satisfying the following: Fix $u \in \{0, 1\}^n$. Let $h : \{0, 1\}^n \rightarrow \{0, 1\}$ be a randomized algorithm such that $h(v) = \langle u, v \rangle$ with probability at least $\frac{1}{2} + \varepsilon$ where the probability is over choice of v , picked uniformly at random, and the internal coin tosses of h . Let $P : \{0, 1\}^n \rightarrow \{0, 1\}$ be a polynomial time computable predicate such that $P(v) = 1$ iff $u = v$. Then, the randomized algorithm GL with oracle access to h and P satisfies*

$$\Pr[\text{GL}^{h,P}(n, \varepsilon) = u] \geq \frac{3}{4}$$

Moreover, the running time of GL is at most $\text{poly}(n, \frac{1}{\varepsilon})$.

Theorem 3.6 guarantees that the algorithm GL computes w in time $\text{poly}(n, 1/\varepsilon)$ with constant probability. However, we do not have the transcript \mathcal{T}^* . (Recall that we only know that there exists a transcript \mathcal{T}^* that satisfies Equation (4), we do not know how to obtain one.) For this purpose, we run the Goldreich-Levin algorithm GL for every possible transcript \mathcal{T} with parameters n and ε . One of these must succeed. Moreover, we can check which one succeeds by verifying that $R(x, w)$ holds. The total time taken by this algorithm is at most $2^{2b} \cdot \text{poly}(n, 2^{2b+1}) \cdot T_A(n) = \text{poly}(n, 2^b) \cdot T_A(n)$. This proves Lemma 3.4. \square

To conclude the proof of the tradeoff result, we now use the assumption that the search problem corresponding to UP relation R does not have randomized algorithm that run in time $2^{o(T(n))}$ on inputs of length n . Therefore, $\text{poly}(n, 2^b) \cdot T_A(n) \geq 2^{\Omega(T(n))}$ and hence $b(n) = \Omega(T(n))$ since $T_A(n)$ is polynomially bounded in n . \square

Remarks:

1. If we make the assumption that there is a search problem in UP that does not have sub-exponential time randomized algorithms, we get a very strong tradeoff. Such an assumption is used in cryptography.
2. We can prove the same result under a weaker assumption that the class FewP has a relation whose search problem is hard. In this case, we could use the set membership function instead of equality.
3. If the search problem corresponding to the relation R had average-case complexity at least $2^{\Omega(T(n))}$ when x is chosen from the distribution \mathcal{D} (instead of worst case complexity), then the same proof as above demonstrates that f_R has average-case communication complexity at least $\Omega(T(n))$ for polynomially bounded Alice and Bob when x is chosen from the distribution \mathcal{D} , z uniformly and $w = w(x)$.

4 Communication vs. Computation in the Query Complexity Model

We consider the query complexity model in which a decision procedure D probes its input x choosing to look at some bits, but not others. The query complexity of a predicate P on n -bit inputs is given by $\min_D \max_x (\# \text{ probes } D \text{ makes on } x)$. Here, D ranges over all decision procedures for P and x ranges over all inputs of length n .

We can consider the computationally bounded analog of this measure, where D is restricted to run in probabilistic polynomial time. Some subtleties arise in such a definition. For example, D must be quantified before n , since polynomial time is an asymptotic notion, but under this quantification there may be no “best” D for all inputs. Also, we may wish to augment our definitions to allow for an error probability.

Fortunately, Theorem 4.2 establishes a tradeoff that is clearly resilient to these technical issues.

Definition 4.1. We say that a one-way permutation p is $\ell(n)$ -lsb hard if no probabilistic polynomial-time procedure, on input x , can compute (simultaneously) the $\ell(n)$ least significant bits of $p^{-1}(x)$ with probability non-negligibly greater than $2^{-\ell(n)}$, where x is chosen uniformly from $\{0,1\}^n$.

We note that such permutations exist based on the hardness of computing discrete logarithms over composite integers [SS90,HSS93].

Theorem 4.2. Let p be $\ell(n)$ -lsb hard. Then there exists a predicate $C_p : (\{0,1\}^n)^{2^{\ell(n)+1}} \rightarrow \{0,1\}$ with the following properties:

1. C_p is computable in polynomial time.
2. The query complexity of C_p is at most $2n$.
3. No polynomial-time bounded decision procedure Q can compute C_p querying only $2^{\alpha\ell(n)}$ bits, where $\alpha < 1$ is any constant. In particular, there is a distribution on the inputs so that if Q computes C_p with advantage ε , then one can compute $\text{lsb}_{\ell(n)}(x)$ from $p(x)$ with probability $\Omega(\varepsilon 2^{-\alpha\ell(n)})$.

Proof. (Sketch) For notational simplicity, we write ℓ instead of $\ell(n)$. We define $C_p(y, x_1, \dots, x_{2^\ell})$ to be 1 iff there exists some i , $1 \leq i \leq 2^\ell$, such that $p(x_i) = y$ and $\text{lsb}_\ell(x_i) = i$ (treating i as an ℓ -bit string).

The predicate C_p is computable in polynomial time, since we can run over all the (polynomially-many) possible values of i . To see that C_p has query complexity at most $2n$, consider the following (computationally unbounded decision procedure):

1. Query y (which is n bits long)
2. Compute $x = p^{-1}(y)$ and $i = \text{lsb}_\ell(x)$.
3. Query x_i (which is n bits long), and accept iff $x_i = x$.

Our proof that no polynomial-time bounded decision procedure exists is by contradiction. Given Q , as above, we construct a polynomial-time algorithm G for guessing $\text{lsb}_\ell(x)$ from $p(x)$, as follows:

1. Given $p(x)$, compute $y = p(x)$ and choose x_1, \dots, x_{2^ℓ} uniformly at random from $\{0,1\}^n$.
2. Run Q on input $(y, x_1, \dots, x_{2^\ell})$. Define I by

$$I = \{i : Q \text{ queries at least one bit of } x_i\}.$$

3. Choose a random index i from I and output i (as an ℓ -bit quantity).

We relate the success probability of G to Q 's advantage, ε at computing $C_p(y, x_1, \dots, x_{2^\ell})$ under the distribution of inputs obtained as follows:

1. Choose x uniformly from $\{0,1\}^n$, and let $y = p(x)$ and $i = \text{lsb}_\ell(x)$.
2. For $j \neq i$, choose x_j uniformly from $\{0,1\}^n$.
3. With probability $1/2$, choose $x_i = x$ (the predicate is true). Else, choose x_i uniformly from $\{0,1\}^n - x$ (the predicate is false).

Clearly, if on a particular run, Q never queries any bit in x_i , it has no advantage in guessing the value of the predicate. It follows that with probability $\Omega(\varepsilon)$, $i \in I$, where I is defined as above. In this case, choosing from I uniformly will yield i with probability $1/|I|$. Since $I \leq 2^{\alpha \ell(n)}$, the theorem follows. \square

Our construction only assumes that $p()$ is strong against polynomial adversaries, resulting in any polynomial tradeoff. With stronger assumptions on the simultaneous hardness of bits in $p()$, we can prove any sub-exponential tradeoff.

5 Solution to the riddle

We now present the solution to the riddle introduced in the introduction. Let s_i denote the sum of the entries in the i th row of M . We show how $k = n + 1$ players can communicate $\text{PS}(M) = \prod_{i=1}^n s_i$ to the referee by each sending a single, *efficiently computable* element of F . (The same solution will work for any larger number of players.) The high-level idea is to first convert the “additive” representation of s_i to a degree-1 polynomial representation over a sufficiently large extension field, then make each player locally multiply its values of the n polynomials (one for each s_i), and finally project down to the original field. The protocol’s outline is described below.

1. Each entry of M is lifted to an extension field F' of F such that $|F'| \geq k + 1$. (This is only a conceptual step and requires no action, since F is a subfield of F' .) Let $\alpha_1, \dots, \alpha_k$ be distinct nonzero elements of F' .
2. The players locally process their entries of M , and each outputs a single element of F' for each row. Let $P_{i,j}$ denote the output of player j corresponding to the i th row. The values $P_{i,j}$ should satisfy the following requirement: for each i , the k points $(\alpha_j, P_{i,j})$ lie on a degree-1 polynomial over F' whose free coefficient is s_i . The implementation of this stage will be described below.
3. Each player j multiplies its n local outputs $P_{i,j}$ from the previous state, resulting in a single element $q_j \in F'$. Note that the k points (α_j, q_j) now lie on a degree- n polynomial whose free coefficient is precisely $\prod_{i=1}^n s_i = \text{PS}(M)$. Since $k > n$, this polynomial can be uniquely determined by interpolation and its free coefficient can be written as $\sum_{j=1}^k \lambda_j q_j$ for some fixed coefficients $\lambda_j \in F'$. Each player j projects $\lambda_j q_j$ down to the original field using a field homomorphism $h : F' \rightarrow F$, and sends the result to the referee.
4. The referee outputs the sum of the k field elements it received.

It remains to describe the implementation of Step 2. Define a $k \times k$ matrix L over F' such that $L_{\ell,m} = 1 - \frac{\alpha_\ell}{\alpha_m}$. For each i , we let $P_{i,j} = \sum_{m=1}^k L_{j,m} M_{i,m}$. Note that since $L_{j,j} = 0$, player j can compute this sum based on his local input. It remains to argue that the above local computations indeed produce the required degree-1 representation of s_i . This follows by noting that for any column m of L , the values $(\alpha_\ell, L_{\ell,m})$ lie on a degree-1 polynomial whose free coefficient is 1. By linearity, the values $(\alpha_j, P_{i,j})$ lie on a degree-1 polynomial whose free coefficient is $\sum_{j=1}^k 1 \cdot M_{i,j} = s_i$. Thus, we have shown:

Theorem 5.1. *The function $\text{PS}(M) = \prod_{i=1}^n \sum_{j=1}^k M_{ij}$, where $k > n$, admits a computationally efficient simultaneous messages protocol in which each player holds all but one column of M and sends a single field element to the referee.*

References

- [BGKL03] BABAI, L., GÁL, A., KIMMEL, P. G., AND LOKAM, S. V. Communication complexity of simultaneous messages. *SIAM Journal of Computing* 33, 1 (2003), 137–166. (Preliminary Version in *12th STACS*, 1995).
- [BTY94] BEAME, P., TOMPA, M., AND YAN, P. Communication-space tradeoffs for unrestricted protocols. *SIAM Journal of Computing* 23, 3 (June 1994), 652–661. (Preliminary Version in *31st FOCS*, 1990).
- [CG93] CANETTI, R., AND GOLDREICH, O. Bounds on tradeoffs between randomness and communication complexity. *Computational Complexity* 3 (1993), 141–167. (Preliminary Version in *31st FOCS*, 1990).
- [CFL83] CHANDRA, A. K., FURST, M. L., AND LIPTON, R. J. Multi-party protocols. In *Proc. 15th ACM Symp. on Theory of Computing* (Boston, Massachusetts, 25–27 Apr. 1983), pp. 94–99.
- [DGS87] DURIS, P., GALIL, Z., AND SCHNITGER, G. Lower bounds on communication complexity. *Information and Computation* 73, 1 (Apr. 1987), 1–22.
- [GL89] GOLDREICH, O., AND LEVIN, L. A. A hard-core predicate for all one-way functions. In *Proc. 21st ACM Symp. on Theory of Computing* (Seattle, Washington, 15–17 May 1989), pp. 25–32.
- [HIKNV] HARSHA, P., ISHAI, Y., KILIAN, J., NISSIM, K., AND VENKATESH, S. Communication vs. computation. Technical Report (to be posted in ECCC). Available at <http://theory.csail.mit.edu/~prahladh/papers/>
- [HSS93] HÅSTAD, J., SCHRIFT, A. W., AND SHAMIR, A. The discrete logarithm modulo a composite hides $O(n)$ bits. *Journal of Computer and System Sciences* 47, 3 (Dec. 1993), 376–404. (Preliminary Version in *22nd STOC*, 1990).
- [NW93] NISAN, N., AND WIGDERSON, A. Rounds in communication complexity revisited. *SIAM Journal of Computing* 22, 1 (Feb. 1993), 211–219. (Preliminary Version in *23rd STOC*, 1991).
- [PS84] PAPADIMITRIOU, C. H., AND SIPSER, M. Communication complexity. *Journal of Computer and System Sciences* 28, 2 (Apr. 1984), 260–269. (Preliminary Version in *14th STOC*, 1982).
- [SS90] SCHRIFT, A. W., AND SHAMIR, A. The discrete log is very discreet. In *Proc. 22nd ACM Symp. on Theory of Computing* (Baltimore, Maryland, 14–16 May 1990), pp. 405–415.
- [Yao79] YAO, A. C.-C. Some complexity questions related to distributive computing (preliminary report). In *Proc. 11th ACM Symp. on Theory of Computing* (Atlanta, Georgia, 30 Apr.–2 May 1979), pp. 209–213.
- [Yao86] YAO, A. C.-C. How to generate and exchange secrets? (extended abstract). In *Proc. 27th IEEE Symp. on Foundations of Comp. Science* (Toronto, Ontario, Canada, 27–29 Oct. 1986), pp. 162–167.