Communication Complexity: Lecture 4

Nathan Harms

January 30, 2026

1 Randomized Communication: Definitions

There are four reasonable ways that one could defined randomized communication. Actually, there are many more, but let's stick with four which come from two basic choices:

1. **What is an acceptable probability of error?** If Bob simply outputs a random coin flip, the answer will be correct with probability 1/2. So, a non-trivial protocol should have error probability $< 1/2$. Now there are two options:
   (a) If we are happy with *any* error probability less than 1/2, we call this requirement *unbounded error*.
   (b) If we require that the error probability is bounded away from 1/2, i.e. we want error probability is at most some constant $\varepsilon < 1/2$, then we call this requirement *bounded error*.
2. **Where does the randomness come from?**
   (a) *Public-coin randomness:* Alice and Bob share a source of randomness, i.e. there is a uniformly random string $\boldsymbol{r} \in \{0,1\}^*$ that they both have access to.
   (b) *Private-coin randomness:* Alice and Bob each have their own random string.

We will use the following notations for the optimal cost of a protocol computing function f in with each of these combinations of requirements:

	Bounded Error	Unbounded Error
Public-Coin	$\mathsf{R}_\varepsilon(f)$	–
Private-Coin	$\mathsf{R}^{\text{priv}}_\varepsilon(f)$	$\mathsf{U}(f)$

We've left the *unbounded error public-coin* spot blank because it turns out this combination of requirements is not interesting, due to the following observation:

Observation 1.1. *Let $f\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be any function. Then there is a public-coin unbounded-error protocol with cost 2.*

Proof. Alice sends Bob the value 1 if the first n bits of the shared random string are equal to x; otherwise she sends 0. If Alice sends 1, Bob outputs $f(x,y)$, otherwise he outputs a uniformly random bit. This protocol has error probability $\frac{1}{2}(1-2^{-n}) < 1/2$. ■

In this lecture we will focus entirely on public-coin protocols, which have the simplest definition.

Definition 1.2 (Public-Coin Protocols)**.** Let $f\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ be any communication problem. A *public-coin protocol* with error ε is a probability distribution $\mathcal{P}$ over deterministic protocols, such that

$$\forall (x,y) \in \mathcal{X} \times \mathcal{Y}\colon \qquad \mathop{\mathbb{P}}_{\boldsymbol{\Pi} \sim \mathcal{P}} [\boldsymbol{\Pi}(x,y) \neq f(x,y)] \leq \varepsilon.$$

The *cost* of the protocol $\mathcal{P}$ is the maximum cost of any protocol Π in the support of the distribution $\mathcal{P}$, i.e. it is

$$\mathrm{cost}(\mathcal{P}) := \max\{\mathrm{cost}(\Pi) \mid \mathop{\mathbb{P}}_{\boldsymbol{\Pi} \sim \mathcal{P}} [\boldsymbol{\Pi} = \Pi] > 0\}.$$

For a given problem f, $\mathsf{R}_\varepsilon(f)$ is the minimum cost of a public-coin protocol computing f.

2 First Example: Equality

Our first (and best) example of randomized communication is the Equality problem, which we studied in the first lecture. Recall that on inputs $x, y \in [N]$, $\mathrm{EQ}_N(x,y) = 1$ if and only if $x = y$.

We saw earlier that $\mathsf{D}(\mathrm{EQ}_N) = \lceil \log N \rceil + 1$, so for deterministic communication this problem is as hard as possible. In stark contrast, Equality is *as easy as possible* for public-coin randomized communication!

Theorem 2.1. $\mathsf{R}_{1/4}(\mathrm{EQ}_N) \leq 3$.

Proof. Using their shared randomness, Alice and Bob choose a uniformly random function $\boldsymbol{h}\colon \{0,1\}^n \to \{0,1\}^2$, i.e. they assign two uniformly random bits $\boldsymbol{h}(z)$ to every string $z \in \{0,1\}^n$. On inputs $x, y \in \{0,1\}^n$, Alice sends $\boldsymbol{h}(x)$ and Bob outputs 1 iff $\boldsymbol{h}(x) = \boldsymbol{h}(y)$.

If $x = y$ then $\boldsymbol{h}(x) = \boldsymbol{h}(y)$ with probability 1. If $x \neq y$ then

$$\mathbb{P}[\boldsymbol{h}(x) = \boldsymbol{h}(y)] = 1/4,$$

so the protocol is always correct with probability at least $3/4$. ∎

3 Error Boosting

Let's revisit the upper bound on Equality. Our protocol had error $1/4$ because the players assigned 2 random bits to each $z \in \{0,1\}^n$. How would we bring the error probability down to a small $\varepsilon > 0$?

One answer is to simply assign more random bits to each $z \in \{0,1\}^n$; with b bits, the probability of error becomes 2^{-b}. Another answer is: repeat the protocol k times. In this case, both answers are actually the same thing. But the second answer generalizes to other problems.

Theorem 3.1. *Let $f\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ be any boolean-valued communication problem, and let $\varepsilon > 0$. Then*

$$\mathsf{R}_\varepsilon(f) = O\left(\mathsf{R}_{1/4}(f) \log(1/\varepsilon)\right).$$

Proof. Let $\mathcal{P}$ be a public-coin protocol computing f with cost c and error $1/4$. We will repeat this protocol k times and then take the majority vote. Let $\mathbf{\Pi}_1, \ldots, \mathbf{\Pi}_k \sim \mathcal{P}$ be k independent deterministic protocols drawn from the distribution $\mathcal{P}$. Alice and Bob will output

$$\text{majority}(\mathbf{\Pi}_1(x,y), \ldots, \mathbf{\Pi}_k(x,y)).$$

We will show that $k = O(\log(1/\varepsilon))$ will suffice to guarantee that this protocol has error at most ε.

For each $i \in [k]$, we write

$$\boldsymbol{Z}_i \coloneqq \mathbb{1}\left[\mathbf{\Pi}_i(x,y) \neq f(x,y)\right],$$

i.e. $\boldsymbol{Z}_i$ is the Bernoulli random variable that takes value 1 iff the i^{th} repetition of the protocol got the incorrect value. Observe that the protocol makes an error iff

$$\sum_{i=1}^{k} \boldsymbol{Z}_i \geq k/2,$$

so we need to show that this occurs with probability at most ε. To do so, we will use a basic concentration inequality:

Theorem 3.2 (Chernoff/Hoeffding Bound)**.** *Let $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$ be independent random variables taking values in $[a,b]$. Let $\mu \coloneqq \mathbb{E}\left[\sum_i \boldsymbol{X}_i\right]$. Then for all $t > 0$,*

$$\mathbb{P}\left[\sum_{i=1}^{k} \boldsymbol{X}_i \leq \mu - t\right] < e^{-\frac{2t^2}{k(b-a)^2}},$$
$$\mathbb{P}\left[\sum_{i=1}^{k} \boldsymbol{X}_i \geq \mu + t\right] < e^{-\frac{2t^2}{k(b-a)^2}}.$$

To apply this theorem to our situation, observe that $\mu \coloneqq \mathbb{E}\left[\sum_i \boldsymbol{Z}_i\right] \leq k/4$. Therefore

$$\mathbb{P}\left[\sum_i \boldsymbol{Z}_i \geq k/2\right] \leq \mathbb{P}\left[\sum_i \boldsymbol{Z}_i \geq \mu + k/4\right] < e^{-\frac{k}{8}}$$

which is less than ε whenever $k > 8\ln(1/\varepsilon)$, so it suffices to take $k = O(\log(1/\varepsilon))$, as desired. ∎

4 More Examples

4.1 Point-Line Incidence

Suppose Alice has a point $(x,y) \in \{0,\ldots,N\} \times \{0,\ldots,N\}$, and Bob has a line ℓ, which can be defined by two numbers $a, b \in \{0,\ldots,N\}$ by setting

$$\ell \coloneqq \{(u,v) \mid v = au + b\}.$$

They want to decide if $(x,y) \in \ell$, i.e. if Alice's point lies on the line ℓ. We write

$$\text{PL}_N((x,y),\ell) = \mathbb{1}\left[(x,y) \in \ell\right].$$

Theorem 4.1 ([CLV19, CHHS25])**.** $\mathsf{R}_{1/4}(\text{PL}_N) = O(\log\log N)$.

Proof. On input (x, y) and (a, b), Alice and Bob will choose a uniformly random prime $\boldsymbol{p} \leq T$, where T is to be chosen later. Alice will send $x \mod \boldsymbol{p}$ and $y \mod \boldsymbol{p}$, and then Bob will output 1 if and only if

$$y \mod \boldsymbol{p} = (a \mod \boldsymbol{p})(x \mod \boldsymbol{p}) + (b \mod \boldsymbol{p}) \equiv y = ax + b \mod \boldsymbol{p}.$$

If (x, y) is on the line, i.e. $y = ax + b$, the protocol will output 1 with probability 1. Now suppose that (x, y) is not on the line, so $y \neq ax + b$. We must determine the probability that

$$y = ax + b \mod \boldsymbol{p}.$$

Consider any two numbers $u, v \in \mathbb{Z}$ and suppose that $u = v \mod p$. Then p must divide the difference $|u - v|$. The number of primes p that divide $|u - v|$ is at most $\log_2 |u - v|$, since each prime is at least 2. Applying this observation with $u = y$ and $v = ax + b$, where $|u - v| \leq N^2 + 2N$, we see that there are at most $\log(N^2 + 2N)$ primes p for which

$$y = ax + b \mod \boldsymbol{p}.$$

Finally, let $\pi(T)$ be the number of primes below T. By the Prime Number Theorem,

$$\pi(T) = \Theta\left(\frac{T}{\log T}\right).$$

Therefore

$$\mathbb{P}\left[y = ax + b \mod \boldsymbol{p}\right] \leq O\left(\frac{\log(N)\log(T)}{T}\right).$$

By choosing sufficiently large $T = O(\log(N)\log\log(N))$, this probability is at most $1/4$. This protocol has cost $O(\log T) = O(\log\log N)$. ∎

Open Problem 1 ([CHHS25])**.** *Show that* $\mathsf{R}_\varepsilon(\mathrm{PL}_N) = \omega(1)$ *(or* $\mathsf{R}_\varepsilon(\mathrm{PL}_N) = O(1)$*).*

4.2 Gap Hamming Distance

For strings $x, y \in \{0,1\}^n$, we write $\mathsf{dist}(x, y)$ for the Hamming distance, i.e.

$$\mathsf{dist}(x, y) = |\{i \in [n] \mid x_i \neq y_i\}|.$$

We define the Gap Hamming Distance problem, $\mathrm{GHD}_{n,k}$, as follows:

$$\mathrm{GHD}_{n,k}(x, y) := \begin{cases} 0 \text{ if } \mathsf{dist}(x, y) \leq k \\ 1 \text{ if } \mathsf{dist}(x, y) \geq n - k \\ * \text{ otherwise.} \end{cases}$$

Here, $*$ means that the protocol is allowed to output either 0 or 1, and both will be considered correct.

Partial vs. Total Problems. A problem defined in this way is called *partial*, because we only fix the output of the protocol on some of the inputs, not all of them. In contrast, problems with a single correct output for every given input are called *total*.

For now, we are interested in version of the problem where $k = n/3$, i.e. $\mathrm{GHD}_{n,n/3}$.

Theorem 4.2. $\mathsf{R}_\varepsilon(\mathrm{GHD}_{n,n/3}) = O(\log(1/\varepsilon))$.

Proof. An exercise left to the reader. ■

This problem has an extremely efficient communication protocol (the cost does not even depend on the size of the inputs, n). But what happens if you replace the $*$ values with other values, i.e. what is the best way to fill in each $*$ with a 0 or a 1, to minimize the communication cost of the resulting *total* problem? We don't know the answer.

Open Problem 2 ([FGHH25, BHH+25, GHR+26]). *What is the minimum value of* $\mathsf{R}_{1/4}(M)$*, over all boolean matrices* M *obtained by filling in the* $*$ *values of* $\mathrm{GHD}_{n,n/3}$*?*

We know that every such M has $\mathsf{R}_{1/4}(M) = \Omega(\log \log n)$, a consequence of some techniques in learning theory [BHH+25].

5 Oracle Protocols

A recent focus of the literature on communication complexity is the idea of an *oracle protocol*, although oracles are generally well studied in complexity theory.

An *oracle protocol* is essentially a communication protocol where Alice and Bob have access to a "subroutine" that (instantly) computes another communication problem. Or in other words, they have access to a magic oracle that provides the solution to a problem, given inputs to that problem.

Using oracles has a few advantages:

1. It lets us design protocols more easily.
2. It lets us compare the power of different communication problems. For example, if communication cost is less when given access to oracle A than oracle B, then A is somehow "more powerful".

In this lecture we will focus entirely on the Equality oracle, i.e. an oracle which, given inputs a from Alice and b from Bob, announces to the players whether $a = b$. Let's give more detail.

Definition 5.1 (EQUALITY oracle protocol)**.** Let $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ be any communication problem. Given inputs $(x, y) \in \mathcal{X} \times \mathcal{Y}$, an EQUALITY oracle protocol is a deterministic protocol where, in every round, Alice and Bob select queries a, b respectively, and the oracle replies with 1 if $a = b$ and 0 otherwise. The queries may be of arbitrary length.

More formally, it is defined similarly to a deterministic protocol: it consists of a binary tree where each leaf is labelled with an output value, and each inner node v is assigned functions $a_v\colon \mathcal{X} \to \{0,1\}^*$ and $b_v\colon \mathcal{Y} \to \{0,1\}^*$. On input x, y, at node v, the protocol proceeds to the left child if $a_v(x) \neq b_v(y)$, and to the right child otherwise. The cost of the protocol is the depth of the tree.

We write $\mathsf{D}^{\mathrm{EQ}}(f)$ for the minimum cost of an EQUALITY oracle protocol computing f.

Two things may seem strange about this definition:

1. The inputs to the oracle can be of arbitrary length. This is justified because $\mathsf{R}_{1/4}(\mathrm{EQ}_n) = O(1)$, i.e. it does not depend on the size of the inputs, so for the purpose of understanding randomized communication, the size of inputs to the oracle don't matter.

2. The players are not allowed to send messages to each other; every single round of communication is a query to the oracle. This makes life easier for us, and it is justified because standard communication can be simulated with EQUALITY oracle queries!

   Exercise 5.2. *Show that each bit of communication in a standard protocol can be simulated by a query to an* EQUALITY *oracle.*

We want to use EQUALITY oracle protocols to design randomized protocols. For this, we require the following theorem:

Theorem 5.3. *For every communication problem f,*

$$\mathsf{R}_{1/4}(f) = O\left(\mathsf{D}^{\mathrm{EQ}}(f) \log\left(\frac{\mathsf{D}^{\mathrm{EQ}}(f)}{\varepsilon}\right)\right).$$

Proof sketch. We design a randomized protocol as follows. On input (x, y), Alice and Bob run the oracle protocol; at each node v, to determine the answer of the query "$a_v(x) = b_v(y)$?", they run the randomized protocol for EQUALITY instead.

To ensure that the answer is correct with probability $1 - \varepsilon$, we must ensure that the probability of getting any incorrect answers throughout the whole protocol is at most ε. If the depth of the oracle protocol is $d = \mathsf{D}^{\mathrm{EQ}}(f)$, there will be at most d queries. Therefore, if we use the randomized protocol for EQUALITY with error $\delta = \varepsilon/d$, the probability that we make any errors is at most $d \cdot \delta = \varepsilon$. Then each query is simulated with

$$\mathsf{R}_\delta(\mathrm{EQ}) = O\left(\log(1/\delta)\right) = O(\log(d/\varepsilon))$$

bits of communication. ■

In fact, we can do better:

Theorem 5.4 ([Nis93, HR24]). *For every communication problem f,*

$$\mathsf{R}_\varepsilon(f) = O\left(\mathsf{D}^{\mathrm{EQ}}(f) + \log(1/\varepsilon)\right).$$

Let's see some examples how to use EQUALITY oracle protocols to design randomized protocols.

5.1 Planar Adjacency

Let $G = (V, E)$ be a planar graph, known to both Alice and Bob. Suppose Alice has a vertex x and Bob has a vertex y. They would like to determine whether x and y are adjacent in G. Let's call this problem $\textsc{Adj}_G$.

Theorem 5.5. *Let G be any planar graph. Then $\mathsf{D}^{\mathrm{EQ}}(\textsc{Adj}_G) \leq 6$.*

Proof sketch. Let's first assume that G is a forest (i.e. a disjoint union of trees). In a forest, vertices x, y are adjacent if and only if one is the parent of the other, in one of the trees. Write $p(x)$ for the parent of x and $p(y)$ for the parent of y. Therefore

$$x, y \text{ adjacent} \iff (x = p(y)) \vee (y = p(x)).$$

To decide whether x, y are adjacent in the forest, Alice and Bob query the oracle on $x, p(y)$ and $p(y), x$.

To get a protocol for planar graphs, we use an interesting fact about planar graphs: any planar graph G can be written as the edge union of 3 forests. Therefore, to determine adjacency in the planar graph, we can determine adjacency in each of the 3 component forests, and then take the OR of the answers. ■

5.2 Greater-Than

Here is another one of my favorite communication problems: GREATER-THAN. Alice and Bob are given integers $i, j \in [N]$ and they wish to decide whether $i > j$. We'll denote this problem by GT_N. Note that numbers i, j have binary representations of length $\lceil \log N \rceil$.

Theorem 5.6. $\mathsf{D}^{\mathrm{EQ}}(\mathrm{GT}_N) = O(\log \log N)$.

Proof Sketch. On inputs i, j, Alice and Bob perform binary search on the binary representations of i and j, to find the highest-order bit where they differ; this will determine which one is larger.

To perform binary search, they can query the EQUALITY oracle on each half of the binary representations, and recurse on the highest-order half where a difference appears. ■

References

[BHH+25] Ari Blondal, Hamed Hatami, Pooya Hatami, Chavdar Lalov, and Sivan Tretiak. Borsuk-ulam and replicable learning of large-margin halfspaces. *arXiv preprint arXiv:2503.15294*, 2025.

[CHHS25] Tsun-Ming Cheung, Hamed Hatami, Kaave Hosseini, and Morgan Shirley. Separation of the factorization norm and randomized communication complexity. *computational complexity*, 34(2):1–27, 2025.

[CLV19] Arkadev Chattopadhyay, Shachar Lovett, and Marc Vinyals. Equality alone does not simulate randomness. In *34th Computational Complexity Conference (CCC 2019)*, pages 14–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019.

[FGHH25] Yuting Fang, Mika Göös, Nathaniel Harms, and Pooya Hatami. Constant-cost communication is not reducible to k-hamming distance. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 565–571, 2025.

[GHR+26] Mika Göös, Nathaniel Harms, Artur Riazanov, Anastasia Sofronova, Dmitry Sokolov, and Weiqiang Yuan. Pseudodeterministic communication complexity. In *Proceedings of the Symposium on the Theory of Computing (STOC)*, 2026.

[HR24] Nathaniel Harms and Artur Riazanov. Better boosting of communication oracles, or not. In *Conference on Foundations of Software Technology and Theoretical Computer Science*, 2024.

[Nis93] Noam Nisan. The communication complexity of threshold gates. *Combinatorics, Paul Erdos is Eighty*, 1(301-315):6, 1993.