18.405J / 6.841J Survey: Large Communication Complexity Classes

Brian Chen and Ray Hua Wu

Abstract

Since the introduction of communication complexity four decades ago, complexity theorists have generally achieved better results and separations in communication complexity than in normal complexity classes. For instance, it is known that no pair of P^{CC}, $\mathsf{BPP}^{\mathsf{CC}}$, $\mathsf{NP}^{\mathsf{CC}}$, and $\mathsf{PP}^{\mathsf{CC}}$ are equal, while not a single pair of all the corresponding normal classes P, BPP, NP, and PP have been proven different. Fewer results have been shown about larger complexity classes, however, and many of those are not well known. In particular, material on unbounded-error communication complexity classes and on $\mathsf{PSPACE}^{\mathsf{CC}}$ is only beginning to materialize.

We survey the structure and relations among the large communication classes $\mathsf{PSPACE}^{\mathsf{CC}}$ and $\mathsf{UPP}^{\mathsf{CC}}$ and related topics, namely space bounds and unbounded error as applied to communication complexity. Along the way, we define a simple flavor of space-bounded communication with public memory, which seems not to have been considered carefully before. We also give a more concise presentation of some commonly-used characterizations of unbounded-error protocols from linear algebra, and examine the separation of $\mathsf{UPP}^{\mathsf{CC}}$ and $\mathsf{PP}^{\mathsf{CC}}$.

1 Introduction

Communication complexity, a modern branch of complexity theory, regards the number of bits required to be transmitted between two parties (usually named Alice and Bob) that have different parts of an input to output a solution to a problem involving both parts. Generally, the problem is taken to be a function from Alice and Bob's inputs to a boolean, $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. The two parties are assumed to have arbitrary computational power when working with their inputs; what matters is minimizing the number of bits necessary to be communicated.

An example of a complexity measure we are concerned with in communication complexity is the deterministic communication complexity $D(f)$, the minimum number of bits Alice and Bob must communicate between them in the worst case to solve some problem f by determining the answer with certainty. When $D(f) = O(\text{polylog}\, n)$, the problem f is said to be in class P^{CC}. An example of a problem in P^{CC} is Odd-Max-Bit-In-Union, where the problem is to output the parity of the highest index in either of the bitstrings Alice

and Bob received that contains a 1. All that is required is for Alice or Bob to send over their highest index with a 1 ($\log n$ bits for the index), and for the other to compare this index with theirs. This is an easy problem to solve, but most are not. (We will see the analogous problem about the intersection of the inputs — Odd-Max-Bit, the highest index where Alice and Bob's bitstrings both contain 1 — later, and see that it is quite hard.)

In general, $O(\text{polylog}\, n)$ communication is considered "efficient" in communication complexity, and classes $\mathsf{NP}^{\mathsf{CC}}$, $\mathsf{RP}^{\mathsf{CC}}$, $\mathsf{PP}^{\mathsf{CC}}$, $\mathsf{BPP}^{\mathsf{CC}}$, and $\mathsf{ZPP}^{\mathsf{CC}}$ have definitions analogous to conventional complexity classes. Surprisingly, many of these have been separated. It has been shown that $\mathsf{P}^{\mathsf{CC}} \neq \mathsf{RP}^{\mathsf{CC}} \neq \mathsf{NP}^{\mathsf{CC}} \neq \mathsf{PP}^{\mathsf{CC}}$. However, while communication is easily understood to be the analogue for time in traditional complexity theory, other concepts such as space and randomness carry over to communication complexity less readily. These concepts are noted for their ability to quickly describe very large complexity classes, however; for example, PSPACE contains the entire polynomial hierarchy. This paper examines the analogues of these concepts and the resulting communication complexity classes in communication complexity, and is structured as follows:

- Section 2 discusses $\mathsf{PSPACE}^{\mathsf{CC}}$ and the more general concept of space-bounded communication. Compared to time and even randomness, it is unintuitive to think about what an analogous definition for $\mathsf{PSPACE}^{\mathsf{CC}}$ is. In this section, we discuss three alternatives for a definition of $\mathsf{PSPACE}^{\mathsf{CC}}$:

  1. a definition based on quantifiers in analogy to the polynomial hierarchy, considered by Babai et al. [1], who were the first to define $\mathsf{PSPACE}^{\mathsf{CC}}$;
  2. a definition based on restricting the additional space Alice and Bob have to remember things between steps, given by Song [13]; and
  3. a definition that restricts the size of the communication channel between Alice and Bob, given by Pálvölgyi [8].

  We show that these definitions are equivalent. Then, we study some counting arguments and one-way protocols, as discussed by Brody et al. [3] and Song [13]. [3] define *oblivious* and *non-oblivious* memory based on how they are used in a type of communication complexity protocols, and prove that Inner-Product requires at least one bit of non-oblivious memory.

- Section 3 discusses $\mathsf{PP}^{\mathsf{CC}}$ and $\mathsf{UPP}^{\mathsf{CC}}$, two classes analogous to PP from traditional complexity theory. The latter was first considered implicitly by Paturi and Simon [9]; Babai et al. [1] are the first to define the former and place the two side by side. We consider these two classes and explores the reasons for their difference: both classes allow arbitrary randomness, but $\mathsf{PP}^{\mathsf{CC}}$ requires an accuracy of $\geq 1/2 + 2^{-\text{poly}(n)}$, whereas $\mathsf{UPP}^{\mathsf{CC}}$ allows arbitrary accuracy $> 1/2$. The reason for considering both classes is that in normal complexity of decision problems, polynomial run time dictates a polynomial bound on randomness and the "granularity" of this

randomness means that an algorithm with accuracy $> 1/2$ actually must have accuracy $\geq 1/2 + 2^{-\operatorname{poly}(n)}$.)

Afterwards, we give some examples of the power of the unbounded-error communication model where any accuracy $> 1/2$ is acceptable, noting that Greater-Than takes only 1 bit and Equality only two bits. Along the way, we examine [9]'s geometric criterion for UPP complexity in depth, giving several examples and carefully analyzing the bounds. We then discuss lower bounds on unbounded-error communication complexity by Forster [5] and Sherstov [12] that build on the above results using techniques from linear algebra. [5] proves a lower bound on the unbounded-error communication complexity of functions based on the corresponding matrix's operator norm, which explicitly places functions defined by Hadamard matrices outside $\mathsf{UPP}^{\mathsf{CC}}$. Then, [12] bounds the unbounded-error communication complexity of all *symmetric functions*, or functions that are completely determined by the number of bits in the AND of the two inputs, to within a polylogarithmic factor, which implies that most of them are outside $\mathsf{UPP}^{\mathsf{CC}}$ as well.

Finally, we briefly the result of Buhrman et al. [4] that separates $\mathsf{PP}^{\mathsf{CC}}$ from $\mathsf{UPP}^{\mathsf{CC}}$ with the explicit problem Odd-Max-Bit, and also look at some related results from Göös et al. [6], which also has an excellent broader overview of communication complexity classes.

- Section 4 discusses open problems suggested in these areas.

2 $\mathsf{PSPACE}^{\mathsf{CC}}$

The communication complexity class $\mathsf{PSPACE}^{\mathsf{CC}}$ was first introduced by Babai et al. [1] as the natural extension of the analogue of the polynomial hierarchy in normal complexity theory, using quantifiers. They wrote, " 'Unlimited alteration' will define $\mathsf{PSPACE}^{\mathsf{CC}}$ (although we do not have a notion corresponding to space)."

Since, as in the case of $\mathsf{NP}^{\mathsf{CC}}$, we can assume Alice and Bob do not communicate after receiving the certificate from the prover since the prover can include their communication in the certificate, we can define $\mathsf{PSPACE}^{\mathsf{CC}}$ as follows:

Definition 1. $\mathsf{PSPACE}^{\mathsf{CC}}$ *is the class of families of functions* $\{f_n\}_{n=1}^{\infty}$*, where* $f_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$*, such that for each* n*, there exists a positive integer* $k = O(\operatorname{polylog} n)$ *and functions* ϕ, ψ *such that* $f_n(x,y) = 1$ *iff*

$$\exists u_1 \forall u_2 \exists u_3 \ldots \forall u_{2k} (\phi(x, u_1, \ldots, u_{2k}) \vee \psi(y, u_1, \ldots, u_{2k})),$$

where each u_i *ranges over* $O(\operatorname{polylog} n)$*-length strings.*

(Since we can choose k depending on n, the exact nature and number of the quantifiers doesn't matter.)

However, there are natural ways to produce a notion of space in communication protocols. Pálvölgyi [8] defines $\mathsf{PSPACE}^{\mathsf{CC}}$ apparently independently of [1]:

In their model, Alice and Bob are still capable of computing anything, but have limited shared memory they can use to record and remember messages passed between them. This can be formalized as follows:

Definition 2. *A* public $s(n)$-space-bounded communication protocol *is defined by two transition functions* $\phi_{0,1} : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^s \cup \{\textsc{Halt}_0, \textsc{Halt}_1\}$.

Alice and Bob have an s-bit shared memory, which is initially all 0s. Taking turns, Alice or Bob can either put a new message into the shared memory, depending on only their input and the previous message in the shared memory, or halt and declare the answer, according to the transition function.

Such a protocol computes a function f if for all inputs x, y, one of Alice and Bob halts and declares the answer $f(x, y)$.

The more-studied definition of a space bound is given by Song [13], who gives the same definition of $\mathsf{PSPACE}^{\mathsf{CC}}$ as [1], using alternating quantifiers, but later defines (essentially) the following natural notion of space and proves in Corollary 3.2 that it is an equivalent definition.

Definition 3. *A* (private) $s(n)$-space-bounded communication protocol *is defined by two transition functions* $\phi_{0,1} : \{0,1\} \times \{0,1\}^n \times \{0,1\}^s \to (\{0,1\}^s \times \{0,1\}) \cup \{\textsc{Halt}_0, \textsc{Halt}_1\}$.

Alice and Bob each have an s-bit private memory, which is initially all 0s. Taking turns, Alice or Bob can either:

i. set the contents of their memory and send a single bit to the other player, both only depending on their input, the previous contents of their memory, and the previous bit received from the other player; or

ii. halt and declare the answer,

according to the transition function.

Such a protocol computes a function f if for all inputs x, y, one of Alice and Bob halts and declares the answer $f(x, y)$.

In this model, it is important that Alice and Bob can only send each other single bits in each turn, since otherwise the bits in transit between them are like a type of memory.

Also note that there are many ways to define exactly when a protocol halts; for simplicity, we allow either player to unilaterally halt the protocol and declare the answer. Most reasonable ways are equivalent up to an additive constant, since at worst Alice and Bob can designate two messages to indicate to each other that the protocol should halt with a certain output, or spend an extra bit asking whether the other player has an answer.

In fact, the natural definitions of $\mathsf{PSPACE}^{\mathsf{CC}}$ derived from the two notions of memory above are asymptotically equivalent, and both coincide with the quantifier-based definition:

Theorem 1. *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a function.*

i. *If there exists a public s-space-bounded communication protocol that computes f, then there exists a private $(s + \log s + 1)$-space-bounded communication protocol that computes f.*

ii. *If there exists a private s-space-bounded communication protocol that computes f, then there exists a public $(2s + 2)$-space-bounded communication protocol that computes f.*

Proof. i. Alice and Bob can simulate a public s-space-bounded communication protocol with a private $(s + \log s + 1)$-space-bounded communication protocol as follows. Both players will have s bits of their private memory reflecting the contents of the shared memory. Each turn in the original protocol where Alice writes a message in the shared memory translates into $2s$ turns in the private protocol (which both players count using $\log s$ bits in their own private memories). At the start of these turns, Alice computes the new message and records it in her private memory; then, over the $2s$ turns, Alice sends Bob her new message bit by bit, and Bob sends 0s. A similar translation occurs when Bob wants to send Alice a message, with roles swapped. (Both players use the last bit to record whose turn it is to send bits.)

ii. Alice and Bob can simulate a private s-space-bounded communication protocol with a public $(2s + 2)$-space-bounded communication protocol as follows. The shared memory has s bits storing Alice's private memory, s bits storing Bob's private memory, 1 bit recording whose turn it is to send a message, and 1 bit where players write a message to send/receive.

□

Theorem 2. $\mathsf{PSPACE}^{\mathsf{CC}}$ *is precisely the class of families of functions computable by public $O(\text{polylog}\, n)$-space-bounded communication protocols.*

Proof. The proof is similar to the proof in traditional complexity theory that $TQBF$ is PSPACE-complete, so we will just briefly sketch it.

- Any $\mathsf{PSPACE}^{\mathsf{CC}}$ function can be simulated by a public $O(\text{polylog}\, n)$-space-bounded communication protocol because the players can recursively evaluate the truth of each quantifier, using shared memory to store the settings for each variable in each quantifier so far.

- Conversely, to prove that a function computed by a public $O(\text{polylog}\, n)$-space-bounded communication protocol is in $\mathsf{PSPACE}^{\mathsf{CC}}$, note that because there are only $2^{O(\text{polylog}\, n)}$ possible messages in the shared memory, protocols can only last for $2^{O(\text{polylog}\, n)}$ steps. Also, players can individually verify that some message in the shared memory would have led to the next message. So we can inductively use quantifiers to bisect the history of messages in the shared memory. This lets us compute, using $O(\text{polylog}\, n)$ quantifiers, whether a history of messages of length $2^{O(\text{polylog}\, n)}$ exists that goes from the initial state to some state where either player halts and outputs 1.

□

Corollary 1. $\mathsf{PSPACE}^{\mathsf{CC}}$ *is precisely the class of families of functions computable by private* $O(\text{polylog}\, n)$*-space-bounded communication protocols.*

This is also Corollary 3.2 of [13], so the reader may consult that for an alternate proof.

2.1 Counting Arguments

A nice feature of the communication protocols considered above is that they are easy to count. Counting allows us to prove that even $(1-\varepsilon)n$ bits of memory is not enough to calculate all functions, much less the $O(\text{polylog}\, n)$ bits allowed by $\mathsf{PSPACE}^{\mathsf{CC}}$; it also allows us to prove a space hierarchy theorem.

Concretely, fix n. Note that there are $2^{2^{2n}}$ functions $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, but, according to the above definitions, only $\left((2^s+2)^{2^{n+s}}\right)^2 \approx 2^{s\cdot 2^{n+s+1}}$ public s-space-bounded communication protocols and $\left((2\cdot 2^s+2)^{2^{n+s+1}}\right)^2 \approx 2^{s\cdot 2^{n+s+2}}$ private s-space-bounded communication protocols. So for any $\varepsilon < 1$, there exist functions that cannot be computed by $\varepsilon\cdot n$-space-bounded communication protocols of either flavor; in fact, considering the limit of the ratio quickly yields that most functions cannot be so computed. The reader may consult [13] for the (simple) computational details.

Theorem 3. *(Theorem 3.7 of [13]) For any* $\varepsilon < 1$*, as* n *approaches infinity, the fraction of functions computable by* $\varepsilon \cdot n$*-space-bounded communication approaches 0.*

Furthermore, there are $2^{2^{n+s}}$ functions that only depend on the first s bits of Alice's input, which can clearly be computed by either flavor of s-space-bounded communication protocol. For public protocols, only one step is needed, in which Alice writes the first s bits in the shared memory and sends it over to Bob; for private protocols, Alice can send it over bit by bit, with Bob writing down the bits one by one in his private memory.

As a result, we have the following space hierarchy theorem:

Theorem 4. *(approximately Corollary 12 of [3]) For every* $s(n) < n - \log n$ *and large enough* n*, there exist functions computable by* $(s+\log s+O(1))$*-space-bounded communication protocols (of either flavor) not computable by* s*-space-bounded communication protocols.*

2.2 One-Way Protocols and Oblivious versus Non-Oblivious Memory

Brody et al. [3] further examine space-bounded *one-way* communication complexity, which involves communication protocols in which all communication is from Alice to Bob, along with the concepts of oblivious and non-oblivious memory:

Definition 4.
- *Oblivious memory is memory that does not depend on input (only on messages received from the other player).*
- *Non-oblivious (or general) memory is memory that may depend on input.*

In general, Bob has both oblivious and non-oblivious memory, and oblivious and non-oblivious memory may have different bounds. In particular, one typically examines cases where non-oblivious memory has a tighter bound than oblivious memory.

EQUALITY and INNER-PRODUCT have different complexities in this model:

Theorem 5. EQUALITY *can be computed with no non-oblivious bits and* $O(\log n)$ *oblivious bits.*

Proof. In a protocol for EQUALITY, Alice simply sends bits over and Bob checks each one to his corresponding bit, using oblivious bits to keep track of the index in their bitstrings. No non-oblivious bits are used. □

Theorem 6. *(Theorem 22 of [3]) To compute* INNER-PRODUCT *with no non-oblivious bits requires at least* $n/8$ *oblivious bits.*

The proof is quite technically detailed, so we can only offer some basic intuition: Every bit in Alice's communication can potentially unilaterally flip the answer to INNER-PRODUCT, so if Bob can't store Alice's entire input, he needs a bit that will vary depending on his input, so at least one bit is necessary.

Even more precisely: Without non-oblivious bits, note that in each step, Bob's oblivious memory and Alice's message can be computed based on only Alice's input. If Bob's oblivious memory is bounded, there are too few memory states to allow Bob to correctly answer for all possible pairs of inputs, due to the way any bit in Alice's input can potentially flip the answer.

On the other hand, INNER-PRODUCT is computable with one non-oblivious bit and $O(\log n)$ oblivious bits, as Bob could simply store the cumulative sum so far as a bit at any stage in the process. So assuming the availability of $O(\text{polylog}\, n)$ oblivious bits, exactly one non-oblivious bit is required for INNER-PRODUCT.

2.3 Further One-Way Memory-Limited Communication

Song [13] continues to develop the theory of one-way space-bounded communication complexity by limiting Bob's memory further.

The first model considered is the *(one-way) memoryless model*, in which Alice can remember how many steps have occurred in the protocol, and sends Bob a message of length s in each step. Bob has no memory and does not know how many steps have occurred. After receiving each message, he decides whether to halt and accept, halt and reject, or continue; this decision can depend only on the message he received and his input. (In this model, we may equivalently assume Alice has s bits of memory, since the protocol can only usefully go on for 2^s steps — there is no point in sending the same message twice.)

Although Bob's lack of memory makes this seems like a very weak model, one should note that it can certainly compute the simple functions considered in Section 2.1, and thus have similar guarantees from the space hierarchy theorem. Perhaps more surprising is this link into the polynomial hierarchy:

Theorem 7. *(Theorem 4.6 of [13]) The set of functions computable in the (one-way) memoryless model with $s = O(\text{polylog}\, n)$ is precisely $\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$.*

To understand the class $\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$, we should define the concept of an oracle in communication complexity:

Definition 5. *(Definition 6.4 of [1], loosely translated) In a communication protocol with some class $\mathcal{C}$ of communication complexity problems as an oracle, either Alice or Bob can pick $f \in \mathcal{C}$, $f : \{0,1\}^m \times \{0,1\}^m \to \{0,1\}$, and $g, h : \{0,1\}^n \to \{0,1\}^m$, and obtain the values $f(g(x), h(y))$, where x, y are Alice and Bob's inputs, respectively.*

The proof of Theorem 7 invokes a concept of *rectangle overlays* developed by [13], which are a more powerful version of the rectangle partitions often considered in communication complexity. We will only briefly sketch the proof.

Proof. (Sketch)

- To convert a one-way memoryless protocol to a $\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$ protocol, Alice and Bob use the $\mathsf{NP}^{\mathsf{CC}}$ oracle to binary-search the timeline of messages Alice would send in the memoryless protocol for the first one that Bob declares an answer on.
- To convert a $\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$ protocol to a one-way memoryless protocol, Alice brute-forces all possible computational histories of the $\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$ protocol, including all possible sequences of oracle answers and every possible certificate that she would accept for every YES answer. For every oracle query, she always sends all possible histories where the answer is YES before trying a history where the answer is NO. Bob waits for a computation history where every certificate for an oracle query is also one he would accept, and outputs the answer that the $\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$ protocol would give.

□

The second model considered is the *(one-way) limited memory model*. As above, Alice can remember how many steps have occurred in the protocol, and sends Bob a message of length s in each step. Bob still does not know how many steps have occurred, but can set himself to one of k memory states, which persist between steps. After receiving each message, he decides whether to halt and accept, halt and reject, or set his memory state and continue; this decision can depend only on the message he received, his input, and his previous memory state. Amazingly, even for $k = 5$ this model does everything the general model does with little overhead:

Theorem 8. *(Theorem 5.14 of [13]) The set of languages computed by a one-way limited memory model with $s = O(\text{polylog}\, n)$ and $k = 5$ is precisely* $\mathsf{PSPACE}^{\mathsf{CC}}$.

The proof of this appeals to a model of computation known as *branching programs*: see [2]. While the proof is too long to give here, we can give some sense of the flavor, including motivation for the constant 5:

Lemma 1. *(Lemma 3 of [2]) There are 5-cycles σ and τ in S_5 such that the commutator $\sigma\tau\sigma^{-1}\tau^{-1}$ is also a 5-cycle.*

Proof.

$$(12345)(13542)(54321)(24531) = (14352).$$

□

(Curiously, [2] writes this sequence of 5-cycles but composes it left-to-right. We follow the right-to-left convention of composing functions above. Fortunately, the lemma is robust to this issue!)

The relationship between the permutations in this lemma and our one-way limited-memory model of communication complexity is as follows: if we fix Bob's input and the message he receives from Alice in some timestep, we can think of what Bob does to his memory state in this step as a permutation over memory states. (It could be any function from memory states to memory states, but permutations are the most powerful, since they don't lose memory information.) So for example, if Bob calculates the 5-cycle $\sigma = (14352)$ from Alice's message and his input and is currently in memory state 2, he sets his memory state to $\sigma(2) = 1$.

This lemma allows Alice and Bob to compose permutations together to build a sort of AND gate. For example, if Bob runs these four instructions:

1. If y_1 is true, follow permutation τ^{-1}; else do nothing (i.e. follow the identity permutation)
2. If y_2 is false, follow permutation σ^{-1}; else do nothing
3. If y_1 is true, follow permutation τ; else do nothing
4. If y_2 is false, follow permutation σ; else do nothing

then Bob will have performed the commutator $\sigma\tau\sigma^{-1}\tau^{-1}$ if y_1 is true and y_2 is false, and otherwise will have done nothing, since the permutations will cancel out. Since the commutator is itself a 5-cycle, this AND gate can be fed into a larger AND gate.

The overall proof strategy of Theorem 8 is to convert the general communication protocol to a type of low-depth circuit with "local preprocessing", a feature defined by [13] that allows it to compute arbitrary functions of booleans in only one half. The low-depth circuit can then be converted into a branching program with the lemma and strategy above.

3 $\mathsf{PP}^{\mathsf{CC}}$ and $\mathsf{UPP}^{\mathsf{CC}}$

The other type of large communication complexity classes is obtained by allowing randomness and very small *bias*, so it is enough for a protocol to accept with probability $1/2+\varepsilon$ for small ε, which may be $o(1)$. By default, we will consider *private* randomness.

Definition 6. *A protocol computes a function with* bias β *if, for every pair of inputs, it computes the right answer with probability at least* $1/2+\beta$.

We may contrast the classes we are about to consider with $\mathsf{BPP}^{\mathsf{CC}}$, which is the class of functions computable with constant positive bias, often taken to be $1/6$ or $1/4$. The exact value does not matter due to amplification. Two large communication complexity classes are defined with very small bias in this way, $\mathsf{PP}^{\mathsf{CC}}$ and $\mathsf{UPP}^{\mathsf{CC}}$.

Definition 7.

- $\mathsf{PP}^{\mathsf{CC}}$ *is the communication complexity class of problems that can be solved with a protocol with access to private randomness that uses* $O(\operatorname{polylog} n)$ *communication, and computes the answer with bias* $\Omega(1/2^{\operatorname{polylog} n})$.
- $\mathsf{UPP}^{\mathsf{CC}}$ *is the communication complexity class of problems that can be solved with a protocol with access to private randomness using* $O(\operatorname{polylog} n)$ *communication, and computes the answer with arbitrary positive bias.*

Observe that the two classes $\mathsf{PP}^{\mathsf{CC}}$ and $\mathsf{UPP}^{\mathsf{CC}}$ have only one analogous class, PP, in the traditional complexity of deterministic algorithms. The reason is that in traditional complexity, randomness is essentially already counted as part of the complexity, since an algorithm must devote computational time to using that randomness, and the fact that limited randomness is available actually means the bias cannot be too small while being positive. More specifically, if a protocol only uses k bits of randomness, its bias must be an integral multiple of $1/2^k$. So, any deterministic algorithm running in $\operatorname{poly}(n)$ time with positive bias automatically has $\Omega(1/2^{\operatorname{poly}(n)})$ bias, and the analogues of $\mathsf{PP}^{\mathsf{CC}}$ and $\mathsf{UPP}^{\mathsf{CC}}$ become identical in traditional complexity theory.

In addition, note that [1] defines $\mathsf{PP}^{\mathsf{CC}}$ by limiting the protocol to only use $O(\operatorname{polylog} n)$ randomness. This is stricter than our definition, since for the same reasons as above, $O(\operatorname{polylog} n)$ randomness implies a bias of $\Omega(1/2^{\operatorname{polylog} n})$. The fact that these definitions are still equivalent can be seen from the conversion in the standard proof of Newman's theorem, which we restate as a standalone theorem below:

Theorem 9. *(based on Newman's theorem) A randomized protocol* P *that computes a function* $f:\{0,1\}^n\times\{0,1\}^n\to\{0,1\}$ *with bias* β *can be converted to a randomized protocol* P' *with bias* $\beta-\delta$ *that uses the same amount of communication but only* $O(\log n+\log\delta^{-1})$ *bits of randomness.*

For a proof, see [10].

Corollary 2. $\mathsf{PP}^{\mathsf{CC}}$ *is the communication complexity class of problems that can be solved with a randomized protocol using* $O(\text{polylog}\, n)$ *communication and* $O(\text{polylog}\, n)$ *bits of randomness.*

Newman's theorem also allows us to see that $\mathsf{PP}^{\mathsf{CC}}$ is unchanged if we consider public instead of private randomness.

Corollary 3. $\mathsf{PP}^{\mathsf{CC}}$ *is the communication complexity class of problems that can be solved with a randomized protocol using* $O(\text{polylog}\, n)$ *communication and* $O(\text{polylog}\, n)$ *bits of* public *randomness.*

However, note that Theorem 9 does not imply that $\mathsf{UPP}^{\mathsf{CC}}$ is unchanged if we consider public randomness, and in fact it is different. Fortunately, $\mathsf{UPP}^{\mathsf{CC}}$ with public randomness is not a very interesting class (a result noted at least in passing in [12], Section 2.1):

Theorem 10. *All function* $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ *can be computed with a protocol using* $O(1)$ *communication and* $O(n)$ *bits of public randomness with positive bias.*

Proof. Alice checks if the first n bits of public randomness exactly match her input. If so, she tells Bob this, so Bob knows Alice's input and can compute and output the correct value of f. Otherwise, they output a random bit. This computes f with bias $1/2^{n+1}$. □

3.1 Power of Unbounded-Error Communication

Intuitively, the model of unbounded-error communication is very powerful. Perhaps the simplest example is Greater-Than, which can be computed with 1 bit of communication.

In general:

Definition 8. *For a function* $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$*, let* $\mathsf{UPP}(f)$ *denote the unbounded-error communication complexity of* f*, that is, the minimum number of bits Alice and Bob must exchange in a communication complexity protocol with unlimited private randomness before one of them can declare the answer with* $> 1/2$ *accuracy.*

Theorem 11.

$$\mathsf{UPP}(\textsc{Greater-Than}) = 1.$$

Proof. Alice and Bob both divide their numbers by 2^n, to get p_a and p_b respectively. Alice chooses to send the bit '1' over to Bob with probability $p_a + \varepsilon$; otherwise, she sends '0'. Bob chooses the bit '1' with probability p_a and '0' otherwise, and compares this to the number Alice sends. If one of them produces bit '1' and the other produces bit '0', the one who produces bit '1' is declared to have the larger number. Otherwise, either '0' or '1' is outputted with probability 1/2.

Regardless what numbers Alice and Bob have, if Alice's number is greater than or equal to Bob's, Alice generates a 1 while Bob generates a 0 with higher probability than Alice generating a 0 while Bob generates a 1. In the case of equals, Alice has a higher probability of being evaluated as greater than or equals due to adding ε to p_a. If Alice's number is less than Bob's, then Alice generates a 1 while Bob generates a 0 with lower probability than Alice generating a 0 while Bob generates a 1. In either of these cases, the two generating the same bit gives either result with $1/2$ probability, so overall in any case the correct answer is generated with probability $> 1/2$. □

A more interesting example is given by EQUALITY, which can be computed with positive bias after Alice sends Bob just 2 bits! (In fact, she does not need the full second bit — she can restrict herself to 3 possible messages, say 00, 01, and 10!) In other words, $\mathsf{UPP}(\text{EQUALITY}) = 2$. The proof is considerably trickier, however; we prove this result below as Corollary 5.

To equip ourselves to prove this, we give a condensed version of some key results of [9], including what we believe to be a more geometrically intuitive version of their Protocol 1. Their first result is the observation that UPP protocols can be assumed to be one-way with only 1 bit of overhead, an interesting type of restricted protocol that reappears from our analysis of space-bounded communication protocols. We give our own less formal proof of this result and invite the reader to read the original paper for a more formal and symbolic treatment.

Definition 9. *For a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, let $\mathsf{UPP}_{\to}(f)$ denote the unbounded-error* one-way *communication complexity of f: the minimum number of bits Alice must send Bob, in a communication complexity protocol with unlimited private randomness where Bob cannot send bits to Alice, before Bob can declare the answer with $> 1/2$ accuracy.*

Theorem 12. *(Theorem 1 of [9]) Fix a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. Suppose there exists a protocol that computes f with positive bias using c bits of communication and private randomness. Then there exists a* one-way *protocol that computes f with at most $c+1$ bits of communication: that is, in this protocol, Alice sends Bob $c+1$ bits and Bob declares the answer. In other words,*

$$\mathsf{UPP}_{\to}(f) \leq \mathsf{UPP}(f) + 1.$$

Proof. Alice randomly chooses a bitstring and pretends it is Bob's input, then simulates the protocol with an imaginary copy of Bob under this assumption. Then she sends the entire history and the output to Bob, who checks if imaginary Bob's answers are all actually the answers Bob would have actually given under the original protocol (which happens at least if the random bitstring Alice chose exactly matches Bob's input, hence with positive probability). If so, he outputs the declared output; otherwise, he outputs a random bit. □

Equality can hold: the most trivial example is a function such as $f(x, y) = x_0$ that only depends on Alice's input. Then clearly $\mathsf{UPP}(f) = 0$, since Alice can

just declare the answer; but $\mathsf{UPP}_{\to}(f) = 1$, since Alice can just tell Bob the correct output and have him declare it, while Bob is hapless to guess the answer without at least one bit from Alice. A more sophisticated example is given after Corollary 4.

Their second result is a geometric characterization of functions that can be computed with positive-bias randomized protocols. The characterization deals with *half-spaces* in $\mathbb{R}^d$, which is a choice of half of $\mathbb{R}^d$ delimited by a hyperplane. We state this more precisely below.

Theorem 13. *(Theorem 2 of [9]) Fix a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. Let d be the smallest positive integer such that there exists an arrangement of 2^n half-spaces $h_1, \ldots, h_{2^n}$ in $\mathbb{R}^d$ satisfying this condition: for every input x, there exists a point P_x such that $P_x \in h_y$ iff $f(x,y) = 1$. (Also note we can assume no P_x lies on any half-space-delimiting-hyperplane by slightly adjusting it if this does not hold.) Let m be the smallest positive integer such that there exists a one-way protocol with private randomness that computes f with positive bias, in which Alice sends one of m messages to Bob and Bob declares the answer. Then*

$$d + 1 = m.$$

As a consequence, the one-way positive-bias communication complexity of f is exactly

$$\mathsf{UPP}_{\to}(f) = \lceil \log m \rceil = \lceil \log(d+1) \rceil .$$

Proof. Let T be the d-dimensional hypertriangle in $\mathbb{R}^{d+1}$ given by points whose coordinates are all nonnegative and sum to 1.

- To prove the forward direction, project the arrangement onto T and scale it such that all points P_x lie in T.

  If Alice's input is x, Alice finds the point P_x and sends a random message from the d possible messages to Bob, choosing message i with probability equal to the ith coordinate of P_x.

  If Bob's input is y, Bob finds the half-space h_y and constructs a linear function ℓ_y of $\mathbb{R}^{d+1}$ such that $0 \le \ell_y(P) \le 1$ on T and $\ell_y(P) \ge 1/2$ precisely when $P \in h_y$. (This can be done by fixing a point P_0 on the border of h_y, taking the dot product of $P - P_0$ with a normal to h_y, scaling suitably, and then adding $1/2$.) Then when Bob receives bitstring s, Bob constructs the point P' that has the sth coordinate 1 and all other coordinates 0 and accepts with probability $\ell_y(P')$.

  This protocol works because Bob's function is linear, so for fixed x and y, since the average value of P' is precisely P along all coordinates, the expected value of $\ell_y(P')$ is precisely $\ell_y(P_x)$, which is $> 1/2$ precisely when $f(x,y) = 1$.

- To prove the reverse direction, the process can be simply reversed after setting $d = m - 1$: for each x, Alice's probabilities of transmitting each of the m possible messages can be combined into the coordinates of a single

point P_x on T, and for each y, Bob's probability of accepting if he receives the bitstring s can be interpolated into a linear function ℓ_y on T. Then the half-spaces h_y can be recovered from ℓ_y. Since the protocol accepts on inputs (x, y) if and only if $P_x \in h_y$, we know that the half-spaces h_y satisfy the theorem conditions.

□

Combining this with Theorem 12 gives:

Corollary 4. $\lceil \log(d+1) \rceil - 1 \leq \mathsf{UPP}(f) \leq \lceil \log(d+1) \rceil$.

Example. Consider a function f described by the communication matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

For this function, we have $d = 2$, since you can take two copies each of the two half-spaces with positive x-coordinate and positive y-coordinate, respectively, in $\mathbb{R}^2$. This is minimal, since two points ("hyperplanes" in $\mathbb{R}^1$) cannot partition $\mathbb{R}^1$ into 4 regions; but since all 4 rows are distinct if you restrict to the first two columns, all 4 P_x must be distinct and must be in different regions as partitioned by the points corresponding to the first two columns. Then $\mathsf{UPP}_{\rightarrow}(f) = 2$.

However, if we transpose the matrix to get

$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

and consider (by abuse of matrix notation) the resulting function f^T, we now have $d = 1$: we can take the rays

$$(-\infty, 0], (-\infty, 3], [4, \infty), [1, \infty)$$

for the four columns respectively, and the points

$$5, 2, 5, 2$$

for the four rows respectively. This implies

$$\mathsf{UPP}(f) = \mathsf{UPP}(f^T) = \mathsf{UPP}_{\rightarrow}(f^T) = 1.$$

As a result,

$$\lceil \log(d_f + 1) \rceil - 1 = \mathsf{UPP}(f) = 1 = \mathsf{UPP}(f^T) = \left\lceil \log(d_{f^T} + 1) \right\rceil,$$

so both equalities in Corollary 4 can hold.

With the geometric criterion of Theorem 13 (which is in fact constructive, since we can convert an arrangements of half-spaces to a protocol and back), it is not too hard to see that Equality has a 3-message protocol, because it is equivalent to the following statement:

Lemma 2. *For any k, there exists a configuration of k half-planes (defined by lying on one side of lines in $\mathbb{R}^2$) such that every half-plane contains some point not in any other half-plane.*

Proof. Fix a circle and draw k distinct tangents to it. For each tangent, take the half-plane defined by it that does not contain the circle. Then a point just outside the circle and near a point of tangency of a line ℓ will be separated from the circle by only that line. □

Corollary 5.

$$\mathsf{UPP}(\textsc{Equality}) = 2.$$

Proof. For any n, the configuration given by Lemma 2 satisfies Theorem 13 in $\mathbb{R}^2$. Thus, $d = 2$ in Theorem 13, so $m = 3$ and $\mathsf{UPP}(\textsc{Equality}) = \lceil \log m \rceil = 2$. □

3.2 Linear-Algebraic Characterization of UPP

Note also that if we increment the dimension by 1, we can consider half-spaces in $\mathbb{R}^{d+1}$ whose delimiting hyperplanes pass through the origin, which are called *homogeneous*. [5] uses this phrasing, which is easier to work with using linear algebraic tools, because then each half-space can be described with a single inner product as $\{u \mid \langle u, v_y \rangle > 0\}$ for some vector v_y. Theorem 3 in [9] also uses a related rephrasing in terms of rank of matrices that *sign-represent* the communication matrix, which becomes the basis for many further proofs in this area.

Definition 10. *A matrix $\hat{M}$* sign-represents *a matrix M with entries in $\{0, 1\}$ if the following conditions are true:*

- *If $M_{xy} = 0$, then $\hat{M}_{xy} < 0$;*
- *If $M_{xy} = 1$, then $\hat{M}_{xy} > 0$.*

The sign-rank *of M is the minimum rank of any matrix that sign-represents M.*

We collect these equivalent criteria in a lemma:

Lemma 3.

- *Let d be defined as in Theorem 13.*
- *Let d' be the smallest positive integer such that there exists an arrangement of 2^n* homogeneous *half-spaces $h_1, \ldots, h_{2^n}$ in $\mathbb{R}^{d'}$ satisfying this condition: for every input x, there exists a point P_x such that $P_x \in h_y$ iff $f(x, y) = 1$.*
- *Let d'' be the smallest positive integer such that there exist 2^n unit vectors u_x and 2^n unit vectors v_y such that for all x, y, the sign of $\langle u_x, v_y \rangle$ gives the value of $f(x, y)$, that is:*
  - *If $f(x, y) = 0$, then $\langle u_x, v_y \rangle < 0$;*
  - *If $f(x, y) = 1$, then $\langle u_x, v_y \rangle > 0$.*

- *(Theorem 3 of [9]) Let d''' be the sign-rank of the communication matrix of f.*

Then $d' = d'' = d'''$ and $d \leq d' \leq d + 1$.

Proof.
- It is clear that $d \leq d'$, since we're just adding the restriction that the half-spaces be homogeneous. To see that $d' \leq d + 1$, just note that an arrangement of half-spaces in $\mathbb{R}^d$ can be converted to an arrangement of homogeneous half-spaces in $\mathbb{R}^{d+1}$ by projecting onto the hyperplane of T as above and turning each $(d-1)$-dimensional hyperplane h into a d-dimensional hyperplane h' that passes through h and the origin.

- We have already essentially proven $d' = d''$: just note that each homogeneous half-space can be written $\{u \mid \langle u, v_y \rangle > 0\}$ for some vector v_y, and normalize each P_x to get v_x. Then note that it does no harm to normalize these vectors.

- We use the classical fact from linear algebra: the rank d''' of a matrix $\hat{M}$ is the minimum positive integer such that $\hat{M}$ can be written as a product of two matrices AB, where A has dimensions $2^n \times d'''$ and B has dimensions $d''' \times 2^n$. Then if we take u_x to be the rows of A and v_y to be the columns of B, we see that this is equivalent to the previous definition so $d'' = d'''$. □

Note, by the way, that these inequalities can both be tight:

Examples.

- Consider a function f described by the communication matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

  For this function, we have $d = d' = 3$, since you can take the three half-spaces with positive x-coordinate, positive y-coordinate, and positive z-coordinate, respectively, in $\mathbb{R}^3$ (in addition to five other trivial half-spaces with hyperplanes far away from the origin that do not contain it). This is tight since three lines cannot partition $\mathbb{R}^2$ into 8 regions, but since all 8 rows are distinct, all 8 P_x must be distinct and must be in different regions. Then $\mathsf{UPP}(f) = 2$.

- However, if we delete the last row (and copy some other row, say the first one, to keep the matrix square) and consider the function described by

the communication matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

we have $d = 2$ while $d' = 3$, since three lines *can* partition $\mathbb{R}^2$ into 7 regions and we can arrange three half-planes to have 7 points satisfying Theorem 13, but not if all three lines are required to go through the origin. Then $\mathsf{UPP}(f) = 1$.

However, the important result arises from combining Lemma 3 with Corollary 4:

Corollary 6.

$$\lceil \log d' \rceil - 1 \leq \mathsf{UPP}(f) \leq \lceil \log(d'+1) \rceil .$$

The inequality on the left side is weaker by 1 than what [5] cites to be the main theorem of [9]. Despite having given examples where equality holds for all four inequalities in Lemma 3 and Corollary 4, however, we were unable to construct an example showing that the left inequality above is tight or prove the tightened version without the -1. An example of f where equality on the left side holds would have to simultaneously satisfy $\mathsf{UPP}(f) = \mathsf{UPP}_{\rightarrow}(f) - 1$ and $\lceil \log d' \rceil = \lceil \log(d+1) \rceil$. A further consequence is that, since $\mathsf{UPP}(\cdot)$ and d' are both symmetric with respect to the roles of Alice and Bob, if (abusing matrix notation again) we consider the function f^T defined by $f^T(x,y) = f(y,x)$, then we must have $\mathsf{UPP}(f) = \mathsf{UPP}(f^T) = \mathsf{UPP}_{\rightarrow}(f^T) - 1$ as well. So such an f must require some back-and-forth communication between Alice and Bob in all optimal protocols. This rules out many simple functions.

In any case, using Corollary 6 and tools from linear algebra, [5] produces a bound based on the operator norm of the communication matrix that proves that several functions, including Inner-Product, requires at least $n/2 - 1 = \Theta(n)$ bits.

Theorem 14. *(Theorem 2.2 of [5], specialized) Consider the communication matrix $M_1 \in \{-1,1\}^{2^n \times 2^n}$ with $\{\pm 1\}$ as entries instead of $\{0,1\}$. With d' as in Lemma 3,*

$$d' \geq \frac{2^n}{\|M_1\|}.$$

Proof. (Sketch) By choosing a suitable linear transformation (Theorem 4.1 of [5]), we may rebalance the u_x so that

$$\sum_{x \in X} u_x u_x^T = 2^n \cdot I_{d'}/d'.$$

Then

$$\sum_{x \in X} |\langle u_x, v_y \rangle| \geq \frac{2^n}{d'}$$

and we can bound the operator norm $\|M_1\|$ with the scalar products $\langle u_x, v_y \rangle$ (Lemma 2.1 of [5]) to get

$$2^n \cdot \left(\frac{2^n}{d'}\right)^2 \leq \sum_{y \in Y} \left(\sum_{x \in X} |\langle u_x, v_y \rangle| \right)^2 \leq 2^n \|M_1\|^2 \Longrightarrow \frac{2^n}{d'} \leq \|M_1\| .$$

□

Corollary 7. *(Corollary 2.1 of [5])*

$$\mathsf{UPP}(f) \geq n - \log_2 \|M_1\| - 1.$$

Then, because M_1 of the matrix for Inner-Product has orthogonal columns, we have $\|M_1\| = \sqrt{2^n}$, so:

Corollary 8. *(Corollary 2.2 of [5])*

$$\mathsf{UPP}(\textsc{Inner-Product}) \geq n/2 - 1.$$

As discussed, [5] states these two results without the -1, and we are not sure whether it can be removed. However, asymptotics are unaffected and the result as stated still suffices to place Inner-Product well outside UPP.

[12] builds on this result to obtain bounds on the unbounded-error communication complexity of all *symmetric functions*, which are functions that only depend on the number of bits in $x \wedge y$ (the most well-known of which are probably Inner-Product and Disjoint). Another result is by [11], which proves that $\Pi_2 P$ is not included in UPP.

3.3 Separation from $\mathsf{PP}^{\mathsf{CC}}$

[4] separates $\mathsf{UPP}^{\mathsf{CC}}$ from $\mathsf{PP}^{\mathsf{CC}}$ with the problem Odd-Max-Bit: given bitstrings $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$ what is the parity of the highest index i such that $x_i = y_i = 1$?

Theorem 15. *(Section 3.1 of [4])*

$$\mathsf{UPP}(\textsc{Odd-Max-Bit}) = O(\log n).$$

Proof. Alice picks a number randomly from $\{1, \ldots, n\}$ such that she picks i with probability

$$\frac{2^i}{2^{n+1} - 1}$$

(the denominator is just a normalizing constant). She sends i and x_i to Bob (taking $\lceil \log n \rceil + 1$ bits). If Bob finds that $x_i = y_i = 1$, he outputs the parity of i; otherwise, he outputs randomly. This computes Odd-Max-Bit with positive bias. □

Theorem 16. *(Section 3.2 of [4]) At least* $\Omega(n^{1/3})$ *communication is required to compute* Odd-Max-Bit *with* $\Omega(1/2^{\operatorname{polylog} n})$ *bias.*

A proof of this theorem is rather difficult, and relies on a result of Razborov involving quantum communication.

Incidentally, [6] observes that Odd-Max-Bit is in fact in $\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$, very low on the communication complexity polynomial hierarchy, and so this provides a lower bound for $\mathsf{PP}^{\mathsf{CC}}$ with that class.

The idea is to note that Nondisjoint is in $\mathsf{NP}^{\mathsf{CC}}$, so to compute Odd-Max-Bit Alice and Bob can use $O(\log n)$ oracle calls to Nondisjoint to binary-search for the longest suffixes of their inputs that are nondisjoint, whereupon they know the answer. Thus, Odd-Max-Bit $\in \mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$.

A related contribution of [6] is to place $\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}}$ inside UPP (in fact, in a somewhat smaller class they designate $\mathsf{UPostBPP}_{\square}$), which provides an interesting link between the unbounded-error classes considered in this section and space-bounded classes in light of Theorem 7:

Theorem 17. *(Observeation 25 of [6])*

$$\mathsf{P}^{\mathsf{NP}^{\mathsf{CC}}} \subseteq \mathsf{UPP}.$$

Proof. (Sketch) For each $\mathsf{NP}^{\mathsf{CC}}$ oracle query, Alice and Bob guess the oracle answer is YES and pick a random certificate with very high probability, and the oracle answer is NO for very low probability. For each YES answer, they check the randomly chosen certificate verifies the YES answer. If any certificate fails to verify, they output a random bit; otherwise, they output the answer based on the guessed oracle answers. For suitably chosen probabilities, they will be more likely to guess the sequence of correct oracle answers and certificates for all the YES answers than any strict subset thereof, and will thus be more likely to output the correct answer in that case, and thus in general. □

4 Open Problems

The biggest open problem involving the two types of classes surveyed is probably whether $\mathsf{UPP}^{\mathsf{CC}} \subseteq \mathsf{PSPACE}^{\mathsf{CC}}$. [1] conjectures that this is false and the two classes are incomparable. (As we have seen, Inner-Product and some other problems are known to be outside $\mathsf{UPP}^{\mathsf{CC}}$, but easily shown to be in $\mathsf{PSPACE}^{\mathsf{CC}}$.)

Somewhat more generally, it would be interesting to have an explicit description a function outside $\mathsf{PSPACE}^{\mathsf{CC}}$. Section 3.5.1 of [13] discusses of the difficulty of this. Proving a function outside $\mathsf{PSPACE}^{\mathsf{CC}}$ would also prove it outside $\mathsf{SPACE}(O(\operatorname{polylog} n))$ in normal deterministic complexity theory; few problems have such bounds except for PSPACE-complete problems such as TQBF, and due to concerns over uniformity, proving even those to be outside $\mathsf{PSPACE}^{\mathsf{CC}}$ would imply $\mathsf{PSPACE} \not\subseteq \mathsf{NC}^1$, which is probably stronger than the celebrated 2010 result $\mathsf{NEXP} \not\subseteq \mathsf{ACC}^0$ of [14].

Taking a broader view, it would be interesting to continue some of the work of [6] to more precisely relate the probabilistic classes like $\mathsf{PP}^{\mathsf{CC}}$ and $\mathsf{UPP}^{\mathsf{CC}}$ to classes in the polynomial hierarchy. They also mention the open problem of whether $\mathsf{UPP}^{\mathsf{CC}}$ is closed under intersection, which reflects the issues the very low bias makes it hard to compose communication protocols.

Looking at the techniques invoked in this survey instead, it might be interesting to examine suitably-defined one-way protocols in other communication complexity classes and see whether they are less powerful and by how much.

Finally, although constant terms are probably generally uninteresting in a complexity theory setting, we have the issue of whether the -1 term in Corollary 6 can be removed.

References

[1] Babai, László, Péter Frankl, János Simon, *Complexity classes in communication complexity theory*. 1986. http://www.tcs.tifr.res.in/~prahladh/teaching/2011-12/comm/papers/BabaiFS1986.pdf

[2] Barrington, David A., *Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1*. Journal of Computer and System Sciences, Vol. 38, No. 1, p. 150–164. 1989. https://people.cs.umass.edu/~barring/publications/bwbp.pdf

[3] Brody, Joshua, Shiteng Chen, Periklis A. Papakonstantinou, Hao Song, and Xiaoming Sun. *Space-Bounded Communication Complexity*. August 14, 2012. http://cs.au.dk/~jbrody/papers/sbcc.pdf

[4] Buhrman, Harry, Nikolay Vereshchagin, Ronald de Wolf. *On Computation and Communication with Small Bias*. 22nd IEEE Annual Conference on Computational Complexity (CCC 07), pp.24-32. http://homepages.cwi.nl/~rdewolf/publ/other/ccsmallbias.pdf

[5] Forster, Jürgen, *A Linear Lower Bound on the Unbounded Error Probabilistic Communication Complexity*. Journal of Computer and System Sciences, Volume 65, Issue 4, December 2002, Pages 612625, Special Issue on Complexity 2001. http://www.cise.ufl.edu/~sitharam/forster

[6] Göös, Mika, Toniann Pitassi, Thomas Watson. *The Landscape of Communication Complexity Classes*. Preprint. 2015. http://www.cs.toronto.edu/~thomasw/papers/lnd.pdf

[7] Lam, T. W., P. Tiwari, and M. Tompa. Tradeoffs between communication and space. In Proceedings of the 21st Annual ACM Symposium on Theory of Computing. https://www.academia.edu/15825043/Trade-offs_between_communication_and_space

[8] Pálvölgyi, Dömötör. Communication Complexity. Eötvös Loránd University, Faculty of Sciences, 2005. http://www.cs.elte.hu/szakdolg/dom.pdf

[9] Paturi, Ramamohan and Janos Simon. Probabilistic Communication Complexity. Journal of Computer and System Sciences 33(1):106-123, July 1986. https://cseweb.ucsd.edu/~paturi/myPapers/pubs/PaturiSimon_1984_focs.pdf

[10] Pitassi, Toniann. *Foundations of Communication Complexity.* https://www.cs.toronto.edu/~toni/Courses/CommComplexity2/Lectures/lecture1.pdf

[11] Razborov, Alexander A., and Alexander A. Sherstov. *The Sign-Rank of AC^0*. SIAM Journal on Computing , 39(5):1833–1855, 2010. http://people.cs.uchicago.edu/~razborov/files/sign.pdf

[12] Sherstov, Alexander A. *The Unbounded-Error Communication Complexity of Symmetric Functions*. http://web.cs.ucla.edu/~sherstov/pdf/unbounded-symm.pdf

[13] Song, Hao. *Space-Bounded Communication Complexity.* Tsinghua University, 2014. http://iiis.tsinghua.edu.cn/~RC3/pdfs/song_phd_thesis.pdf

[14] Williams, Ryan. *Non-Uniform ACC Circuit Lower Bounds*. 2010. http://www.cs.cmu.edu/~ryanw/acc-lbs.pdf

18.405J / 6.841J Advanced Complexity Theory

Spring 2016