Overlays and Limited Memory Communication

Periklis Papakonstantinou*, Dominik Scheder†, Hao Song*
*Tsinghua University, Beijing, China
†Simons Institute for the Theory of Computing, Berkeley, USA, and Tsinghua University, Beijing, China

***Abstract*—We give new characterizations and lower bounds relating classes in the communication complexity polynomial hierarchy and circuit complexity to limited memory communication models.**

We introduce the notion of *rectangle overlay complexity* of a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. This is a natural combinatorial complexity measure in terms of combinatorial rectangles in the communication matrix of f. Furthermore, we consider memoryless and limited-memory communication models, originally introduced in [1] with slightly different terminology. In these communication models there are two parameters of interest: The maximum message length s (which we think of as *space*) and the number of memory states w. Specifically, these are one-way protocols which proceed in rounds. In each round, Alice sends a message of at most s bits to Bob; receiving a message from Alice, Bob has to decide on the spot whether to output 0 or 1 and halt, or to continue the protocol. If he decides to continue, he immediately forgets Alice's message. In memoryless protocols, no memory is transferred between different rounds (but Bob still has "space" to hold Alice's messages within each round). We can make Bob more powerful by giving him w memory states. He can change into a new state at the end of each round.

We show that rectangle overlays completely characterize memoryless protocols. Then, we go on to show several connections to the communication complexity polynomial hierarchy defined by Babai, Frankl and Simon in 1986 [2]. This hierarchy has recently regained attention because its connection to the algebrization barrier in complexity theory [3]. We show that $\mathsf{P}^{\mathsf{NP}^{cc}}$ is completely characterized by memoryless protocols with $\mathsf{polylog}(n)$ space (maximum message length), and thus it admits a purely combinatorial characterization in terms of rectangle overlays. If Bob has 3 memory states and Alice sends messages of length $\mathsf{polylog}(n)$, they can compute every level of Σ_k^{cc} in the communication complexity hierarchy (for constant k), and also every function in AC^0. Furthermore, we show that with 5 memory states and messages of length $\mathsf{polylog}(n)$ they can compute exactly the functions in the communication class PSPACE^{cc}. This gives the first meaningful characterization of PSPACE^{cc} in terms of space, originally defined in [2] without any notion of space. We also study equivalences and separations between our limited memory communication model and branching programs, and relations to circuit classes.

***Keywords*-Communication Complexity, Space, Polynomial Hierarchy, Combinatorial Characterization.**

I. Introduction

Communication complexity is one of the jewels of theory of computation. There is a number of profound questions of interest to communication complexity itself, though its power mostly shows when proving lower bounds for other models of computation. For instance, lower bounds for streaming e.g. [4], [5]), circuit complexity e.g. [6], [7], and property testing [8].

In classical communication complexity, two players Alice and Bob are each given only part of the input to a computational problem. The goal is to cooperatively solve the problem by exchanging as little information as possible. The computational power of Alice and Bob is unlimited. In particular, they have unlimited time and unlimited memory space. In this two-player setting, we assume Alice and Bob receive inputs $x, y \in \{0,1\}^n$ respectively and wish to compute a boolean function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ on (x, y), that is to determine the value of $f(x, y)$.

In this work we revisit some novel one-way communication models, first introduced in [1] with slightly different terminology,[1] in which the memory of the receiving end is severely limited (in some cases, even without any explicit memory). Our models show surprising connections to the communication classes corresponding to the polynomial time hierarchy [2]. These classes regained interest recently through their relation to algebrization [3], [9]. Our models give rise to characterizations of different levels of this hierarchy, either through very simple and intuitive combinatorial characterizations or through our limited memory communication models. First, these characterizations make transparent known separation results in the hierarchy (in particular those achieved recently by Impagliazzo and Williams [10]). In addition, we explore relationships between our models and bounded-width branching programs and bounded-depth circuit classes.

We hope that this rich set of connections leads to further advancements in the aforementioned fields of complexity. We discuss future research directions in Section VIII.

A. Memoryless Communication Model and its Combinatorial Characterization

In the *one-way memoryless* model, protocols proceed in rounds. In round i, Alice sends a message $A(i, x)$ to Bob. Upon receiving this message, Bob has three choices: He can terminate and output 0; terminate and output 1; or decide

[1]We introduced new terminology since [1] reflecting our understanding on what matters. In [1] Bob had two types of memory. A large oblivious one that was only a function of the communication, and a small non-oblivious one that was a function of everything. Now, Bob has a small number w of memory states—this corresponds to the old notion of non-oblivious memory. Each of Alice's messages consist of up to s bits—the s corresponds to the old notion of oblivious memory.

to continue. If he continues, he forgets everything that has happened so far, and enters the next round. The *cost* of the protocol is $\max_{x,i} |A(i,x)|$, i.e. the maximum message length. The *memoryless complexity* of a function f, denoted by $S(f)$, is the complexity of the most efficient protocol that correctly computes f, i.e. $\min_{\mathcal{P}} \max_{x,i} |A(i,x)|$ where $\mathcal{P}$ ranges over all protocols that correctly compute f.

1) Rectangle Overlay: This one-way memoryless model has a simple and intuitive combinatorial characterization: *rectangle overlays*. The rectangle overlay concept generalizes the rectangle partition and rectangle cover concepts that have been previously used in the study of classical communication complexity. Similar to rectangle partitions and rectangle covers, rectangle overlays are defined using combinatorial rectangles of the form $R = X \times Y$, where $X, Y \subseteq \{0,1\}^n$. A *rectangle overlay of length* l is a sequence of tuples $(R_1, b_1), (R_2, b_2), \ldots, (R_l, b_l)$ where each R_i is a combinatorial rectangle that has "color" $b_i \in \{0,1\}$. For completeness, the union of these rectangles must cover the whole domain $\{0,1\}^n \times \{0,1\}^n$. This overlay defines a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ in the following way: For each input pair $(x,y) \in \{0,1\}^n \times \{0,1\}^n$, find the first rectangle R_i in the sequence that contains (x,y), then define the output of $f(x,y)$ to be b_i. We denote the minimum length of any rectangle overlay that computes f as $RO(f)$.

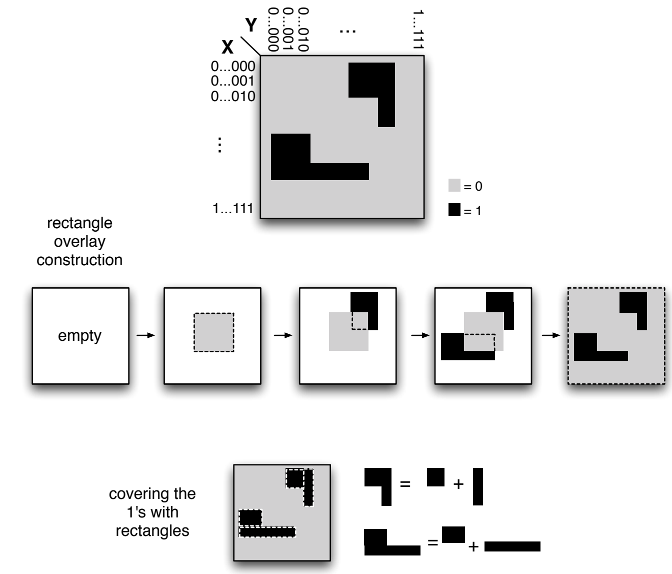
Y
X
0...000
0...001
0...010
...
1...111
= 0
= 1
rectangle overlay construction
empty
covering the 1's with rectangles
=
+

Figure 1. The top square is the communication matrix of some function. Below it is a construction of this matrix by a rectangle overlay (we draw this using the "overlay" feature of image processing tools, hence the name). The third row shows a covering of the 1s by rectangles. We will see that rectangle overlays can be exponentially shorter than rectangle covers (in terms of number of rectangles).

As one reviewer pointed out, there is a simple equivalent definition of overlays: Consider a decision list (i.e., width-1 branching program) of length l that can query not only single bits of the input but can make *rectangle queries*, i.e., check whether $(x,y) \in R_i$. Our first result shows that the overlay complexity $RO(f)$ of a function characterizes its memoryless complexity $S(f)$.

Theorem 1 (Rectangle overlay characterizes memoryless complexity)**.** *For every boolean function* $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$*, it holds that* $S(f) \leq \lceil \log(RO(f)) \rceil \leq 2S(f) + 1$.

As an example, we give a rectangle overlay of "Greater-Than" function GT: For $x, y \in \{0,1\}^n$, $\text{GT}(x,y) = 1$ if and only if $x > y$, where $>$ is the lexicographic order. A standard fooling set argument (cf. [11] Section 1.3) shows that one needs at least $2^n - 1$ monochromatic rectangles to cover all the $x > y$ inputs; and at least 2^n monochromatic rectangles for all the $x \leq y$ inputs. However, there exists a rectangle overlay of length $2n+1$ that computes GT. Say $x = x_1 x_2 \ldots x_n$ and $y = y_1 y_2 \ldots y_n$. We define $R_i^{>} \overset{\text{def}}{=} \{(x,y) \mid x_i = 1, y_i = 0\}$ and give it color 1; similarly, $R_i^{<} \overset{\text{def}}{=} \{(x,y) \mid x_i = 0, y_i = 1\}$ gets color 0. Finally, we set $R_{n+1} = \{0,1\}^n \times \{0,1\}^n$ and give it color 0. The rectangle overlay

$$(R_1^{>}, 1),\ (R_1^{<}, 0),\ \ldots,\ (R_n^{>}, 1),\ (R_n^{<}, 0),\ (R_{n+1}, 0)$$

has length $2n+1$ and computes GT. If one can cover the 1-entries of a function f with m rectangles, then surely $RO(f) \leq m+1$ (one needs an additional "background rectangle" for the 0's), thus by Theorem 1, $S(f) \leq N^1(f) + 1$, and similarly $S(f) \leq N^0(f) + 1$. Furthermore, GT shows that $S(f)$ can be exponentially smaller than $\min(N^1(f), N^0(f))$.

Another nice fact, pointed out by one of the reviewers, is that the deterministic communication complexity $D(f)$ is at most $RO(f)$: Indeed, in a deterministic protocol, Alice can send a bit-vector telling Bob with which rectangles her input x is consistent. Bob can then compute the smallest index i such that $(x,y) \in R_i$, and outputs b_i.

The rectangle overlay characterization allows a very intuitive lower bound technique for one-way memoryless protocols.

Theorem 2 (Rectangle Lower Bound)**.** *Let* $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ *be a boolean function and* μ *a product probability distribution on* $\{0,1\}^n \times \{0,1\}^n$*. Let* $\epsilon_\mu \overset{\text{def}}{=} \max_R \mu(R)$*, where* R *ranges over all monochromatic rectangles in the communication matrix of* f*. Then* $S(f) \geq \frac{1}{4} \log\left(\frac{1}{\epsilon_\mu}\right) - \frac{1}{2}$.

Using this lower bound technique we prove nearly tight lower bounds in the memoryless complexity of several functions, including the inner product function over GF(2)

(Corollary 12), the "List-Non-Equality" function (Corollary 13) and the "Gap-Hamming-Distance" function (Corollary 14.) [2]

Open Problem 1. *In the setup of Theorem 2, let $\epsilon \stackrel{def}{=} \min_\mu \epsilon_\mu$, where μ ranges over all product distributions. Is $S(f) \leq \mathsf{polylog}\left(\frac{1}{\epsilon}\right)$?*

2) The Class $\mathsf{P}^{\mathsf{NP}^{cc}}$: Babai, Frankl, and Simon [2] defined a communication analogue of the polynomial hierarchy of the Turing Machine world. This hierarchy is defined using non-determinism and alternating quantifiers, similarly to its Turing Machine counterpart. For completeness we recall the formal definitions of these communication classes in Section II. For now, let us just say that the first level of this hierarchy $\Sigma_1^{cc} = \mathsf{NP}^{cc}$ is the class of all functions $f = \{f_n\}_{n=1}^{\infty}$ [3] whose nondeterministic communication complexity $N^1(f)$ is at most $\mathsf{polylog}(n)$. Similarly, $\Pi_1^{cc} = \mathsf{coNP}^{cc}$ contains those f's with $N^0(f) \in \mathsf{polylog}(n)$.

The class $\mathsf{P}^{\mathsf{NP}^{cc}}$ is defined using protocols with access to a certain *oracle*. Suppose Alice and Bob can exchange messages as usual, but in addition they can query *Oracles* in the class NP^{cc} (by each sending her part for the query). The cost of such a protocol is the total amount of communication, where an oracle query on strings of length s counts as $\lceil \log(s) \rceil$ bits of communication. Now, $\mathsf{P}^{\mathsf{NP}^{cc}}$ is the class of all functions computable by such an oracle protocol of cost at most $\mathsf{polylog}(n)$.

Theorem 3. *A function $f = \{f_n\}_{n=1}^{\infty}$ is in $\mathsf{P}^{\mathsf{NP}^{cc}} \iff S(f) \in \mathsf{polylog}(n)$.*

$\mathsf{P}^{\mathsf{NP}^{cc}}$ is strictly between the first and second level of the polynomial hierarchy:

Fact 4 ([10]). $\Sigma_1^{cc} \cup \Pi_1^{cc} \subsetneq \mathsf{P}^{\mathsf{NP}^{cc}} \subsetneq \Sigma_2^{cc} \cap \Pi_2^{cc}$

Impagliazzo and Williams [10] proved the second separation using the "List-Non-Equality" function LNE: $S(\text{LNE}) = \Omega(\sqrt{n})$. Note that LNE can be computed by a depth-3 circuit of polynomial size. We improve this separation showing that there are two functions of small depth-3 circuits but *linear* memoryless complexity: One of them in Σ_2^{cc} and the other in Π_2^{cc}. Both of these functions are total function extensions of the partial function "Gap-Hamming-Distance" (Corollary 14.)

The complexity classes Σ_k^{cc} and Π_k^{cc} have a nice combinatorial characterization as iterated intersections and unions of monochromatic rectangles [2]. Before our work no such characterization was known for $\mathsf{P}^{\mathsf{NP}^{cc}}$. We give a very elegant (and somewhat unexpected) combinatorial characterization for $\mathsf{P}^{\mathsf{NP}^{cc}}$: Rectangle overlays characterize one-way memoryless protocols (Theorem 1), which in turn characterize $\mathsf{P}^{\mathsf{NP}^{cc}}$ (Theorem 3). In addition, we are now able to give very intuitive proofs for several results about $\mathsf{P}^{\mathsf{NP}^{cc}}$ in [10], including the above separation.

[2]Technically, the "Gap-Hamming-Distance" function is a partial function, we prove lower bound for every total-function extension of this function

[3]When we discuss asymptotics, we use the notation $f = \{f_n\}_{n=1}^{\infty}$ to denote a family of boolean functions, one for each input length n, $f_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. The communication protocol (or branching program, or circuit) involved should also be understood as a non-uniform family of protocols (or branching programs, or circuits), one for each input length.

B. Limited Memory Communication Model

In the *one-way limited memory* communication model, we give Bob w memory states. Note that this memory is "permanent": Bob can make his decisions depending on Alice's message and his current memory state, and can also update the memory state for the next round. Contrary to this, he will not remember Alice's old message in the next round. We let $\mathsf{SPACE}_{\mathsf{LTD}}[s, w]$ be the set of all functions computable by a one-way limited memory protocol with w memory states and maximum message length s. Surprisingly, protocols in which Bob has 5 memory states and Alice sends messages of length $\mathsf{polylog}(n)$ can compute exactly those functions PSPACE^{cc}, which is the "infinite-level" limit of the hierarchy in [2].)

Theorem 5. $\mathsf{PSPACE}^{cc} = \mathsf{SPACE}_{\mathsf{LTD}}[\mathsf{polylog}(n), 5]$. *Moreover, the same holds true when replacing 5 by a larger constant.*

Despite what its name suggests, when the class PSPACE^{cc} was first defined by Babai, Frankl and Simon in 1986, the authors noted that "we do not have a notion corresponding to space". Now through this newly found equivalence, we can actually put a "polynomial" space notion behind this long known class PSPACE^{cc}. We emphasize that we really need this subtle notion of "space" in order to characterize PSPACE^{cc}, i.e. $\mathsf{polylog}(n)$ message length and these all-powerful 5 memory states (lacking those 5 states by Theorem 3 we go down to $\mathsf{P}^{\mathsf{NP}^{cc}}$).

Theorem 5 is the communication complexity analogue of Barrington's Theorem [12]. Barrington's Theorem states that every function in NC^1 can be computed by a width-5 branching program of polynomial length. In fact, the proof of Theorem 5 also shows the following.

Theorem 6. *Suppose $f \in \mathsf{NC}^1$. Then $f \in \mathsf{SPACE}_{\mathsf{LTD}}[O(\log n), 5]$ (under every input partition).*

With 3 memory states for Bob and maximum message length polylog, Alice and Bob can compute any constant level of the communication polynomial hierarchy.

Theorem 7. *For every constant $k \in \mathbb{N}^+$ (independent of the input length n), $\Sigma_k^{cc} \cup \Pi_k^{cc} \subseteq \mathsf{SPACE}_{\mathsf{LTD}}[\mathsf{polylog}(n), 3]$.*

As a corollary, we have:

Corollary 8. *Suppose $f \in \mathsf{AC}^0$, then $f \in \mathsf{SPACE}_{\mathsf{LTD}}[\mathsf{polylog}(n), 3]$ (under every input partition).*

We prove Theorem 7 by derandomizing Razborov and Smolensky's low-degree polynomial approximation of bounded-depth circuits [13], [14]. For this we borrow ideas from Newman's randomness reduction scheme [15] and Braverman's error indicator [16]. Allender and Hertrampf did something similar in their circuit depth reduction paper [17].

The following fact states that limited-memory protocols can simulate bounded-width branching programs:

Fact 9. *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. If there is a width-$w$ branching program of length l that computes f, then $f \in \mathsf{SPACE}_{\mathsf{LTD}}[\lceil \log l \rceil + 1, w]$.*

The other direction does not hold. Indeed, there is a natural function for which limited-memory protocols are more efficient than branching programs. Consider the inner product modulo 3: $\mathsf{IP}_3(x,y) \overset{\text{def}}{=} \sum_{i=1}^{n} x_i y_i \mod 3$. An unpublished result by Shearer [18] shows that every width-2 branching program computing IP_3 has length $\Omega\left(\left(\frac{3}{2\sqrt{2}}\right)^n\right)$. On the other hand, there is a protocol in which Bob has two memory states that computes IP_3 with $O(\sqrt{n})$ maximum message length. More generally:

Theorem 10. *Suppose $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ can be decomposed as $f(x,y) = g(h_1(x,y), \ldots, h_k(x,y))$ where each $h_i : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ depends on at most ℓ bits from each of its two n-bit inputs. Then $f \in \mathsf{SPACE}_{\mathsf{LTD}}[\ell + k + \lceil \log k \rceil, 2]$.*

The details of the proof of this theorem are deferred to Appendix. Please see the full version of this paper [19] for all appendices. We can decompose IP_3 into $\sqrt{n}$ blocks of $\sqrt{n}$ bits each. The value of IP_3 on one block is a number in $\{0,1,2\}$ and can be represented using two bits. Thus, $k = 2\lceil \sqrt{n} \rceil$ and $\ell = \lceil \sqrt{n} \rceil$, and $\mathsf{IP}_3 \in \mathsf{SPACE}_{\mathsf{LTD}}[O(\sqrt{n}), 2]$.

C. Roadmap to the Rest of the Paper

We recall basic definitions in Section II. Formal definitions of our communication model are given in Section III. Section III also contains our results concerning rectangle overlays and some preliminary discussions and relations to circuit classes (in the Appendix we give an example of a function separating our memoryless model from AC^0). Sections IV, V, VI contain characterizations and separations of $\mathsf{P}^{\mathsf{NP}^{cc}}$, Σ_k^{cc}, and PSPACE^{cc}. In Section VIII we discuss some future research directions.

II. Standard models: Definitions and notation

This paper studies relations of communication memory models with boolean circuits and branching programs. For a detailed treatment cf. [20], [21]. For completeness we briefly recall the definitions here. We also recall the definitions of oracle classes in Communication Complexity [2]. For a comprehensive treatment and definitions of deterministic and non-deterministic communication complexity cf. [11] (Section 1.1 and 2.1).

Definition 1 (Boolean Circuits)**.** *A* boolean circuit *is a directed acyclic graph (DAG) with a unique sink. Each source of this graph corresponds to a boolean input variable; each internal node has a label in $\{\wedge, \vee, \neg\}$ denoting the boolean operation (gate) to apply (the in-degree of $\neg$ nodes must be 1). The* fan-in *of a node is its in-degree. We evaluate the output of the circuit inductively by evaluating the corresponding gates. The* depth *of a circuit is the length of the longest path from one of the sources to the unique sink.*

Definition 2 (Bounded-Depth Circuit Classes: AC^0 and NC^1)**.** *A family $\{f_n\}_{n=1}^{\infty}$ of boolean functions is in AC^0 if every f_n can be computed by a boolean circuit on the basis $\{\vee, \wedge, \neg\}$ with constant depth, polynomial size and unbounded fan-in. A family $\{f_n\}_{n=1}^{\infty}$ is in NC^1 if every f_n can be computed by a boolean circuit on the basis $\{\vee, \wedge, \neg\}$ with $O(\log n)$ depth, polynomial size and fan-in 2.*

Definition 3 (Branching Programs)**.** *Let $w, l, n \in \mathbb{N}^+$. A width w length l branching program defined on an n-bit input, is defined as a sequence of l instructions and a set $S_{\text{YES}} \subseteq \{1, 2, \ldots, w\}$. Each instruction is of the form $< i, f, g >$, where $i \in \{1, 2, \ldots, n\}$ and f, g are functions from $\{1, 2, \ldots, w\}$ to $\{1, 2, \ldots, w\}$. Such a branching program computes a function $B : \{0,1\}^n \to \{0,1\}$ with the computation defined as follows: A* state *M is initially set to 1. We execute the instructions in the branching program in order: For an instruction $< i, f, g >$, we read the i^{th} bit of the input. If this is 0, update $M \overset{def}{=} f(M)$; if it is 1, update $M \overset{def}{=} g(M)$. If the state M is in S_{YES} after the final instruction, output 1. Otherwise output 0.*

We now define related communication complexity classes: The polynomial hierarchy PH^{cc}, the class PSPACE^{cc}, and oracle communication classes. The polynomial hierarchy PH^{cc} can be defined in terms of a game: Given $x, y \in \{0,1\}^n$, a prover and a disprover take k turns in trying to prove / disprove that $f(x,y) = 1$. In the end, the "referees" Alice and Bob communicate and output 0 or 1. The value of this protocol on (x,y) is defined to be 1 if the prover has a winning strategy, i.e., can make sure that Alice and Bob output 1 in the end. Otherwise, the value is 0. The whole communication of prover, disprover, Alice, and Bob must be at most $\mathsf{polylog}(n)$ bits long. If the prover speaks first, this is a Σ_k-protocol. If the disprover speaks first, a Π_k-protocol. Now Σ_k (Π_k) is the set of functions $\{f_n\}_{n=1}^{\infty}$ that can be computed by a Σ_k-protocol (Π_k-protocol). More formally:

Definition 4 (Communication Complexity Polynomial Hierarchy)**.** *Let $k(n) \leq \mathsf{polylog}(n)$. We say that a family of boolean functions $\{f_n\}_{n=1}^{\infty}$ can be computed by a $\Sigma_{k(n)}^{cc}$ protocol if there exist $\phi, \psi : \{0,1\}^n \times \{0,1\}^* \to \{0,1\}$,* *such that for every two n-bit strings x, y, $f_n(x, y) = 1$ if and only if*

$$\exists u_1 \forall u_2 \exists u_3 \ldots Q_{k(n)} u_{k(n)} (\phi(x, u) \diamond \psi(y, u))$$

Here for each $i \in [k(n)]$, u_i is a bit string of length at most $\mathsf{polylog}(n)$. *When k is even, $\diamond$ stands for $\vee$ and $Q_{k(n)}$ is $\forall$; when k is odd, $\diamond$ stands for $\wedge$ and $Q_{k(n)}$ is $\exists$.*

We define $\Pi^{cc}_{k(n)}$ analogously, by switching the roles of the two quantifiers ($\exists$ and $\forall$) and the two boolean operators ($\wedge$ and $\vee$).

Finally, we define $\mathsf{PH}^{cc} \overset{def}{=} \bigcup_{k=1}^{\infty} \Sigma_k$ and $\mathsf{PSPACE}^{cc} \overset{def}{=} \Sigma_{\mathsf{polylog}(n)}$.

That is, PH^{cc} allows any *constant* number of alternations between prover and disprover, whereas PSPACE^{cc} allows $\mathsf{polylog}(n)$ many alternations.

Example: List-Non-Equality. Consider the function LNE, defined as follows: Partition $x \in \{0,1\}^n$ into $\sqrt{n}$ many blocks of equal size: $x = x^{(1)} \ldots x^{(\sqrt{n})}$. Similarly $y = y^{(1)} \ldots y^{(\sqrt{n})}$. Now $\mathsf{LNE}(x, y) = 1$ if and only if $x^{(i)} \neq y^{(i)}$ for all $1 \leq i \leq \sqrt{n}$. This function is in Π_2: If $f(x, y) = 0$, the disprover can send an index i such that $x^{(i)} = y^{(i)}$. If this is wrong, and in fact $x^{(i)} \neq y^{(i)}$, then the prover can send some j such that $x^{(i)}_j \neq y^{(i)}_j$. After this, Alice and Bob exchange two bits to determine whether prover or disprover was right. The total amount of communication is $2 \lceil \log n \rceil + 2$. In fact, $\mathsf{LNE} \in \Sigma_2$, too, although this is a little bit more difficult to prove [22].

Example: Inner Product. Let $\mathsf{IP}(x, y) \overset{def}{=} \sum_{i=1}^{n} x_i y_i \bmod 2$. This function is in PSPACE^{cc}. Partition x into two $n/2$-bit strings: $x = x^{(1)} x^{(2)}$; similarly $y = y^{(1)} y^{(2)}$. To prove that $f(x, y) = 1$, the prover announces two bits: $\mathsf{IP}(x^{(1)}, y^{(1)})$ and $\mathsf{IP}(x^{(2)}, y^{(2)})$. If the prover lies, the disprover can challenge him, and the protocol recurses to determine IP on the part about which the prover allegedly lied. If this sub-protocol determines that the prover indeed lied, the protocol outputs 0. Otherwise, it outputs the parity of the two announced bits. Proving that $f(x, y) = 0$ is analogous, with the roles of the prover and disprover exchanged. This protocol has a total of $O(\log n)$ alternations and $O(\log n)$ communicated bits. Thus, $\mathsf{IP} \in \mathsf{PSPACE}^{cc}$.

We should remark that it is not known whether $\mathsf{IP} \in \Sigma_2$ or even whether $\mathsf{PSPACE}^{cc} = \Sigma_2$.

Observation 11 (Relations between Communication Classes and Circuit Classes)**.** *Let $\{f_n\}_{n=1}^{\infty}$ be a family of functions $f_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. It holds that $\{f_n\}_{n=1}^{\infty} \in \Sigma^{cc}_{k(n)}$ if and only if there exist $m \leq 2^{\mathsf{polylog}(n)}$, functions $A, B : \{0,1\}^n \to \{0,1\}^m$, and a circuit C on $2m$ input variables such that:*

- *The depth of C is at most $k(n) + 1$.*
- *The output gate of C is $\vee$.*
- *C has size at most $2^{\mathsf{polylog}(n)}$.*
- *$f(x, y) = C(A(x), B(y))$.*

Similarly, $\{f_n\}_{n=1}^{\infty} \in \Pi^{cc}_{k(n)}$ if and only if the circuit is as described, but with an $\wedge$-gate at the top.

Definition 5 ($\mathsf{P}^{\mathsf{NP}^{cc}}$)**.** *An* oracle protocol *is a classical communication protocols, where Alice and Bob can also make queries of the form $Q(f(x), g(y))$. Here $f, g : \{0,1\}^n \to \{0,1\}^m$ are arbitrary preprocessing functions and Q is the oracle. The cost of an oracle query is defined to be $\log m$. The total cost of the protocol is the number of communicated bits plus the total cost of all queries. The class $\mathsf{P}^{\mathsf{NP}^{cc}}$ contains exactly those functions computed by protocols with an oracle in NP^{cc} and total cost $\mathsf{polylog}(n)$.*

III. Model Definition and a Combinatorial Characterization

Here we provide a formal definition of our one-way limited memory communication model, related complexity measures, complexity classes and the rectangle overlay characterization. Also, at the end of this section we list some important consequences of these definitions and characterizations.

Definition 6. *A* one-way limited-memory *protocol has two parameters of interest: s, the maximum length of Alice's messages, and w, the number of Bob's memory states. Alice is specified by a function $A : \mathbb{N}^+ \times \{0,1\}^n \to \{0,1\}^s$, Bob by $B : \{0,1\}^n \times \{0,1\}^s \times [w] \to \{0, 1, \perp\} \times [w]$. The protocol proceeds in rounds. In round i, Alice computes a message $\alpha := A(i, x)$ and sends it to Bob. Bob, who is in memory state $q \in [w]$ when he receives α, computes $(\beta, q') := B(y, \alpha, q)$. If $\beta \in \{0, 1\}$, he outputs β and the protocol ends. If $\beta = \perp$, he changes his memory state to q' and the players enter round $i + 1$. The number of rounds is limited: If the protocol has not halted within $w \cdot 2^s$ rounds, it outputs 0 by default.*

$\mathsf{SPACE}_{\mathsf{LTD}}[s, w]$ *is the set of functions computable by a one-way limited-memory protocol with maximum message length s and w memory states. If $w = 1$ we call the protocol* memoryless *and define* $\mathsf{SPACE}_{\mathsf{LESS}}[s] \overset{def}{=} \mathsf{SPACE}_{\mathsf{LTD}}[s, 1]$.

It is essential that the number of rounds be limited. Otherwise, every problem becomes solvable with message length $\lceil \log(n + 1) \rceil + 1$ and two memory states; this is an analog to the (easy) fact every function can be computed by a width-2 branching program (of exponential length). In a *memoryless* protocol though, Alice will never need to send the same message twice. That is, an optimal memoryless protocol with messages of length s will take at most 2^s rounds, and the limit we imposed becomes redundant.

A. The Rectangle Overlay Characterization

As already discussed in the introduction, one of the main contributions of this paper is a combinatorial characterization of one-way memoryless protocols through *rectangle* *overlays*. Below is the formal definition and the proofs of the characterization theorems.

Definition 7. *For a positive integer n, a* rectangle overlay *on $\{0,1\}^n \times \{0,1\}^n$ of length l is a sequence of tuples $(R_1, b_1), (R_2, b_2), \ldots, (R_l, b_l)$ such that*

- *for each $i \in \{1, 2, \ldots, l\}$, R_i is a combinatorial rectangle on the domain $\{0,1\}^n \times \{0,1\}^n$ (in other words, $R_i = X_i \times Y_i$ where $X_i \subseteq \{0,1\}^n$ and $Y_i \subseteq \{0,1\}^n$) and $b_i \in \{0,1\}$*
- $\bigcup_{i=1}^{l} R_i = \{0,1\}^n \times \{0,1\}^n$

Such a rectangle overlay computes/defines a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ in a natural way. For every input pair $(x, y) \in \{0,1\}^n \times \{0,1\}^n$, find the smallest i such that $(x, y) \in R_i$, then define the output of $f(x, y)$ to be b_i.

The rectangle overlay complexity *of a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, denoted as $RO(f)$, is defined to be $RO(f) \overset{\text{def}}{=} \min\{l :$ there is a rectangle overlay of length l that defines $f\}$.*

Theorem 1 (restated). *For every boolean function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, it holds that $S(f) \leq \lceil \log (RO(f)) \rceil \leq 2S(f) + 1$.*

Proof: begin with the first (and easier) inequality: $S(f) \leq \lceil \log (RO(f)) \rceil$. Let $l = RO(f)$. There is a rectangle overlay $(R_1, b_1), (R_2, b_2), \ldots, (R_l, b_l)$ that computes f. We construct a one-way memoryless protocol with messages of length at most $s \overset{\text{def}}{=} \lceil \log l \rceil$. For each $1 \leq i \leq l$ write $R_i = X_i \times Y_i$, where $X_i, Y_i \subseteq \{0,1\}^n$. Given input $(x, y) \in \{0,1\}^n \times \{0,1\}^n$, let $\{X_{i_1}, X_{i_2}, \ldots, X_{i_{l'}}\}$ be the set of all X_i's such that $x \in X_i$, and $i_1 < i_2 < \ldots < i_{l'}$. Our protocol works as follows: In round j, Alice sends i_j to Bob. Bob outputs b_i if $y \in Y_i$ and $\perp$ if otherwise. Every message of Alice can be encoded with s bits.

Second, we show that $\lceil \log (RO(f)) \rceil \leq 2S(f) + 1$. Let $s \overset{\text{def}}{=} S(f)$. There is a one-way memoryless protocol with maximum message length s that computes f. We construct a rectangle overlay of length at most 2^{2s+1} that computes f. According to Definition 6, there are functions A and B such that in the i-th round, Bob computes $B(y, A(i, x)) \in \{0, 1, \perp\}$ to determine whether to give the final answer from the set $\{0,1\}$ or continue with the computation. For each tuple $(i, \alpha, b) \in \{1, 2, \ldots, 2^s\} \times \{0,1\}^s \times \{0,1\}$, we define $X_{i,\alpha,b} \overset{\text{def}}{=} \{x \mid A(i, x) = \alpha\}$, $Y_{i,\alpha,b} \overset{\text{def}}{=} \{y \mid B(y, \alpha) = b\}$ and $R_{i,\alpha,b} \overset{\text{def}}{=} X_{i,\alpha,b} \times Y_{i,\alpha,b}$.

Note that for fixed i, the rectangles $R_{i,\alpha,b}$ are disjoint. We order the rectangles $\{R_{i,\alpha,b}\}_{(i,\alpha,b)}$ in the increasing order of i and color each rectangle $R_{i,\alpha,b}$ with color b. This gives a rectangle overlay that computes f and has length 2^{2s+1}. ■

Theorem 2 (restated). *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a boolean function and μ a product probability distribution on $\{0,1\}^n \times \{0,1\}^n$. Let $\epsilon_\mu \overset{\text{def}}{=} \max_R \mu(R)$, where R ranges over all monochromatic rectangles in the communication matrix of f. Then $S(f) \geq \frac{1}{4} \log \left(\frac{1}{\epsilon_\mu} \right) - \frac{1}{2}$.*

Proof: Let $\ell = RO(f)$. There is a rectangle overlay (R_1, b_1), (R_2, b_2), …, (R_ℓ, b_ℓ) that computes f. Then according to Theorem 1, $S(f) \geq \frac{\log(\ell)}{2} - \frac{1}{2}$.

Next we will construct a sequence of rectangles $T_0, T_1, T_2, \ldots, T_\ell$ such that: (i) $T_0 = \{0,1\}^n \times \{0,1\}^n$; (ii) $T_\ell = \emptyset$; (iii) $R_j \cap T_i = \emptyset$ for all $1 \leq j \leq i \leq \ell$; (iv) for every $i \in \{1, 2, \ldots, \ell\}$, we have $T_i \subseteq T_{i-1}$, and (v) $\mu(T_i) \geq \mu(T_{i-1}) - \sqrt{\epsilon}$. The existence of such a sequence implies that $\ell \geq \frac{1}{\sqrt{\epsilon}}$ and $S(f) \geq \frac{\log(\ell)}{2} - \frac{1}{2} \geq \frac{1}{4} \log \left(\frac{1}{\epsilon} \right) - \frac{1}{2}$.

We construct the sequence $\{T_i\}$ as follows: Set $T_0 = \{0,1\}^n \times \{0,1\}^n$. We define $T_1, T_2, \ldots, T_\ell$ inductively. For every $i \in \{1, 2, \ldots, \ell\}$, define $\bar{R}_i \overset{\text{def}}{=} R_i \cap T_{i-1}$. $\bar{R}_i$ is clearly a combinatorial rectangle, and by property (iii) it is disjoint from all previous $R_1, \ldots, R_{i-1}$. Thus, the function defined by the rectangle overlay outputs color b_i for all $(x, y) \in \bar{R}_i$. In other words, $\bar{R}_i$ is monochromatic. Write $T_{i-1} = X_{i-1} \times Y_{i-1}$ and $\bar{R}_i = \bar{X}_i \times \bar{Y}_i$. Since μ is a product distribution, we can write $\mu = \mu_A \times \mu_B$, where μ_A and μ_B are both probability distributions on $\{0,1\}^n$. We have $\mu(\bar{R}_i) = \mu_A(\bar{X}_i) \cdot \mu_B(\bar{Y}_i)$ and $\min\{\mu_A(\bar{X}_i), \mu_B(\bar{Y}_i)\} \leq \sqrt{\mu(\bar{R}_i)} \leq \sqrt{\epsilon}$. Then we define

$$T_i \overset{\text{def}}{=} \begin{cases} (X_{i-1} \setminus \bar{X}_i) \times Y_{i-1} & \text{if } \mu_A(\bar{X}_i) \leq \mu_B(\bar{Y}_i) \\ X_{i-i} \times (Y_{i-1} \setminus \bar{Y}_i) & \text{if } \mu_A(\bar{X}_i) > \mu_B(\bar{Y}_i) \end{cases}$$

In words, we obtain T_i by cutting a piece away from T_{i-1} to ensure that T_i and R_i are disjoint. Thus, T_i satisfies (iii) and (iv). Finally, the piece we cut away has weight at most $\sqrt{\epsilon}$, thus (v) is satisfied, too. Since $\cup_{i=1}^{2^m} R_i = \{0,1\}^n \times \{0,1\}^n$ it holds that $T_\ell = \emptyset$. This completes the construction of the sequence and concludes the proof.

■

B. *Consequences of the Rectangle Overlay Characterization*

The above characterization significantly simplifies and strengthens lower bounds proved previously (as in e.g. [1]), and significantly simplifies and appropriately conceptualizes [10]. Also, involving the isoperimetric properties of the Hamming Cube we obtain Lemma 14 whose proof is given in the Appendix.

Corollary 12 (Theorem 22 in [1]). $S(\mathrm{IP}) \geq \frac{n}{4} - \frac{1}{2}$. *Here* IP *is the Inner-Product function for two n-dimensional vectors over* $\mathsf{GF}(2)$.

Proof: It is well-known that every monochromatic rectangle of Inner Product has size at most 2^n. See for example Example 1.25 in the book by Kushilevitz and Nisan [11]. Let μ be the uniform distribution on $\{0,1\}^n \times \{0,1\}^n$. Thus $\mu(R) \leq 2^{-n}$ for every monochromatic rectangle R in communication matrix. ■

Corollary 13. $S(\mathrm{LNE}_{\sqrt{n},\sqrt{n}}) \geq \frac{1}{2}\sqrt{n} - \frac{1}{4}\log n - \frac{1}{2}$. *Here* $\mathrm{LNE}_{b,k}$ *is a boolean function over two* (bk)*-bit inputs* $x = x_{1,1}x_{1,2}\dots x_{1,k}x_{2,1}\dots x_{b,k}$ *and* $y = y_{1,1}y_{1,2}\cdots y_{1,k}y_{2,1}\cdots y_{b,k}$ *(that is, each input is divided into* b *blocks, each block is* k*-bit long, e.g.* $x_{3,5}$ *is the fifth bit in the third block). And* $\mathrm{LNE}_{b,k}(x,y)$ *outputs* 1 *if and only if for every block* i*, there is at least one bit position* j *within this block such that* $x_{i,j} \neq y_{i,j}$*.*

Proof: Choose μ to be the uniform distribution on $\{0,1\}^n \times \{0,1\}^n$. In the communication matrix of $\mathrm{LNE}_{b,k}$, every 0-monochromatic rectangle R_0 satisfies $\mu(R_0) \leq b^2 \cdot 2^{-2k}$, and every 1-monochromatic rectangle R_1 satisfies $\mu(R_1) \leq 2^{-2b}$. See Impagliazzo and Williams [10] for a proof. We simply apply Theorem 2, substitute in $b = k = \sqrt{n}$, and the proof is done. ■

Note that this lower bound, together with the fact that $\mathrm{LNE}_{\sqrt{n},\sqrt{n}} \in \Sigma_2^{cc} \cap \Pi_2^{cc}$ [22], gives a separation $\mathsf{P}^{\mathsf{NP}^{cc}} \subsetneq \Sigma_2^{cc} \cap \Pi_2^{cc}$.

Lemma 14. *There exists* $f = \{f_n\}_{n=1}^{\infty} \in \Sigma_2^{cc}$*, and* $g = \{g_n\}_{n=1}^{\infty} \in \Pi_2^{cc}$*, both extensions of the partial function "Gap Hamming Distance"* GHD*, such that* $S(f) = \Omega(n)$ *and* $S(g) = \Omega(n)$*.*

$\mathrm{GHD} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ *is defined as follows:* $\mathrm{GHD}(x,y) = 0$ *if the Hamming distance* $d_H(x,y) \leq n/3$*; if* $d_H(x,y) \geq 2n/3$*, then* $\mathrm{GHD}(x,y) = 1$*; in between,* GHD *is not defined.*

The proof of this lemma is given in the Appendix. Actually f and g are computable by depth-3 AC^0 circuits. Thus, there are functions in AC^0 requiring rectangle overlays of length $2^{\Omega(n)}$.

IV. $\mathsf{P}^{\mathsf{NP}^{cc}}$ and Memoryless Protocols

Theorem 3 (restated). *A function* $f = \{f_n\}_{n=1}^{\infty}$ *is in* $\mathsf{P}^{\mathsf{NP}^{cc}}$ $\iff$ $S(f) \in \mathsf{polylog}(n)$*.*

Proof: Let us start with the easier direction. Assume that $S(f) \in \mathsf{polylog}(n)$. We will show that $f \in \mathsf{P}^{\mathsf{NP}^{cc}}$. The oracle we will use here is the "NP^{cc}-complete" function[4] "intersect", denoted by INT. For two n-bit strings $x = x_1x_2\dots x_n$ and $y = y_1y_2\dots y_n$, $\mathrm{INT}(x,y) = 1$ if and only if there exists $i \in \{1,2,\dots,n\}$ such that $x_i = y_i = 1$.

By Theorem 1 each f_n has a rectangle overlay $(R_1,b_1),(R_2,b_2),\dots,(R_{\ell(n)},b_{\ell(n)})$ where $\ell(n) \leq 2^{\mathsf{polylog}(n)}$. Write $R_i = X_i \times Y_i$ for each $i \in \{1,2,\dots,\ell(n)\}$. Alice and Bob preprocess x and y into two $\ell(n)$-bit strings $\hat{x},\hat{y} \in \{0,1\}^{\ell(n)}$: For each $1 \leq i \leq \ell(n)$, set $\hat{x}_i$ to be 1 if $x \in X_i$ and 0 otherwise; define $\hat{y}$ analogously. Given these two $\ell(n)$-bit strings, Alice and Bob plan to find the smallest i such that $\hat{x}_i = \hat{y}_i = 1$ and output b_i. They can find this i using binary search, by quering INT at most $\mathsf{polylog}(n)$ many times, each query of length at most $2^{\mathsf{polylog}(n)}$. This gives us an oracle protocol in $\mathsf{P}^{\mathsf{NP}^{cc}}$ and shows that $f \in \mathsf{P}^{\mathsf{NP}^{cc}}$.

For the other direction, we assume that $f = \{f_n\}_{n=1}^{\infty} \in \mathsf{P}^{\mathsf{NP}^{cc}}$. We will give a one-way memoryless protocol with maximum message length $\mathsf{polylog}(n)$. Consider f_n, let $\mathcal{P}$ be the corresponding oracle protocol and $\mathcal{T}$ its protocol tree. According to Definition 5, $\mathcal{T}$ has maximum depth $\mathsf{polylog}(n)$, and every query node makes a query to an NP^{cc}-function of input length $2^{\mathsf{polylog}(n)}$.

For every NP^{cc}-function Q, there exists a family of combinatorial rectangles $\{R_i\}$, such that for each (x,y), $Q(x,y) = 1$ if and only there is i such that $(x,y) \in R_i$. We call i a *certificate* for (x,y). Each certificate is of length at most $\mathsf{polylog}(n)$.

For each input (x,y) to f_n, we can describe the computational history the protocol will follow in $\mathcal{T}$ as follows: First we use a $\mathsf{polylog}(n)$-bit string p to denote all the communication bits and query answers along the way, from the root down to one of the leaf nodes; then we scan the path from root the leaf, for each query that answers 1, we concatenate one of the certificates for (x,y). This gives a string $(p,c_1,c_2,\dots,c_t)$, in which $p,c_1,c_2,\dots,c_t$ are all of $\mathsf{polylog}(n)$ length and $t = O(\mathsf{polylog}(n))$. Therefore the whole string is of length $\mathsf{polylog}(n)$.

Now let H be the set of all possible computational histories for all possible input pairs (x,y). We construct a one-way oblivious protocol $\mathcal{P}'$ for f in the following way: Alice enumerates all computational histories in H that are compatible with her input x, lexicographically decreasing in p. That is, if $h = (p,c_1,\dots,c_t)$ and $h' = (p',c'_1,\dots,c'_{t'})$ are two histories with $p < p'$ (in lexicographic order), then Alice enumerates h' before h.

Each time Alice sends the computational history to Bob, Bob checks to see if this computational history is also compatible to his input y. If so, he outputs the label of the corresponding leaf node, otherwise he just continues.

Clearly $\mathcal{P}' \in \mathsf{SPACE}_{\mathsf{LESS}}[\mathsf{polylog}(n)]$. Let us prove that $\mathcal{P}'$ correctly computes f. Consider an input pair (x,y). Let p be root-to-leaf path in $\mathcal{T}$ followed by the original protocol $\mathcal{P}$ on input (x,y), and let h be one of the corresponding histories containing p as defined above. Let $h^* = (p^*,c_1^*,\dots,c_t^*)$ be the computational history accepted by Alice and Bob in the protocol $\mathcal{P}'$. We will prove that $p = p^*$ by contradiction. Suppose $p \neq p^*$. Let v be the last node in $\mathcal{T}$ that is common to p and p^*.

- v cannot be a communication node associated with either Alice or Bob: Otherwise they would reject p^* as incompatible to their input.
- So v must be a query node.
  - Suppose p takes the 0-child of v whilst p^* takes the 1-child of v. This is impossible: The correct

[4]INT is complete for NP^{cc} under the so-called "rectangle reduction", please refer to [2] for more details.

answer for the query at node v is 0 (since p is the correct path), so either Alice or Bob would have rejected the purported 1-certificate in h^*.

– Suppose p takes the 1-child of v whilst p^* takes the 0-child of v. Then $p > p^*$ lexicographically, and thus Alice enumerates h before h^*. Since h is the correct history and is consistent for both Alice and Bob, they would have accepted it before they even consider h^*.

Thus $p = p^*$ and $\mathcal{P}'$ is correct. We conclude that $f \in \mathsf{SPACE}_{\mathsf{LESS}}[\mathsf{polylog}(n)]$. ∎

V. Constant levels of Σ_k^{cc} and 3-state protocols

We define $\mathsf{PH}^{cc} \overset{\text{def}}{=} \bigcup_{k=1}^{\infty} \Sigma_k^{cc}$. This means that k can be an arbitrarily large *constant*, independent of the input length n.

Theorem 7 (restated). $\mathsf{PH}^{cc} \subseteq \mathsf{SPACE}_{\mathsf{LTD}}[\mathsf{polylog}(n), 3]$.

Proof outline: First, we show that for an arbitrary function $f \in \mathsf{PH}^{cc}$, there is a *randomized* protocol that computes f, in which Bob has only two states. That is, on every $(x, y) \in \{0,1\}^n \times \{0,1\}^n$, the protocol outputs the correct value with probability at least $2/3$. Second, we show how to make this protocol zero-error. Finally, we show that every zero-error protocol can be completely derandomized by giving Bob one additional memory state.

A. Bounded-error and zero-error protocols

In a bounded-error protocol, Alice and Bob share a source of randomness: At the beginning of the protocol, Alice and Bob are given the same infinite random bit string $r_1 r_2 r_3 \ldots$, and Alice's messages can depend on r: $P_A(i, x, r)$ instead of $P_A(i, x)$. Similarly, Bob can base his decision to output 0, output 1, or continue, on this random string. A protocol is a *bounded-error protocol computing* $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ if for each $(x, y) \in \{0,1\}^n \times \{0,1\}^n$, it outputs the correct value $f(x, y)$ with probability at least $2/3$.

As in bounded-error protocols, in zero-error protocols Alice and Bob share a random string r. In a zero-error protocol, Bob has the additional possibility to abort the protocol and output "don't know". A protocol is a *zero-error protocol computing* $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ if for each $(x, y) \in \{0,1\}^n \times \{0,1\}^n$, (i) the output of the protocol is either $f(x, y)$ or "don't know", and (ii) Bob outputs "don't know" with probability smaller than $1/2$.

Theorem 15. *Let $f \in \mathsf{PH}^{cc}$. There exists a zero-error protocol computing f where Bob has two memory states and Alice sends messages of length at most* $\mathsf{polylog}(n)$.

To prove this theorem, we first give a bounded-error protocol computing f; in a second step we make this protocol zero-error. We do not know whether every bounded-error protocol can be made zero-error without increasing the maximum message length or Bob's memory. In the special case of $f \in \mathsf{PH}^{cc}$, it turns out that we can: The trick is to use not only a Razborov-Smolensky approximation for f, but also an "error-indicator", in spirit similar to Braverman [16].

Proof of Theorem 15: Let $f \in \mathsf{PH}^{cc}$. That is, $f \in \Sigma_k^{cc}$ for some k (that is independent of the input length n). By the connection between complexity classes and circuits (Observation 11), there exists $m \in 2^{\mathsf{polylog}(n)}$, functions $A, B : \{0,1\}^n \to \{0,1\}^m$, and an AC^0-circuit C over variables $u, v \in \{0,1\}^m$ such that $f(x, y) = C(A(x), B(y))$. We approximate C using the well-known method of Razborov and Smolensky [13], [14].

Lemma 16 (Razborov and Smolensky [13], [14]). *Given a circuit C of depth d and size S on the basis $\{\vee, \oplus, \neg\}$ with unbounded fan-in on an m-bit input x, there is a scheme of assigning random polynomials over* $\mathsf{GF}(2)$ *to each gate in the circuit (the additive identity 0 corresponds to boolean value false, and the multiplicative identity 1 corresponds to boolean value true), such that*

- *For each gate g in $\mathcal{C}$, denote the random polynomial chosen as p_g. The degree of p_g is at most $(\log(3S))^d$.*
- *For every m-bit input x, we have*

$$Pr[\exists \text{ gate } g \in \mathcal{C} \text{ such that } g(x) \neq p_g(x)] \leq \frac{1}{3} .$$

Here, $g(x)$ is the output of gate g.

- *If g is an input gate with input x_i, the i-th bit in x, then $p_g = x_i$.*
- *If g is a $\neg$-gate and c is its only child, then $p_g = p_c + 1$*
- *If g is an $\oplus$-gate with children $c_1, c_2, \ldots, c_t$, then $p_g = \sum_{i=1}^{t} p_{c_i}$*
- *If g is a $\vee$-gate with children $c_1, c_2, \ldots, c_t$ and $p_{c_1}(x) = \cdots = p_{c_t}(x) = 0$, then $p_g(x) = 0$, too.*

Note that by applying DeMorgan's rule, we can assume our circuit C consists only of $\vee$-, $\oplus$-, and $\neg$-gates. The above lemma is already enough to obtain a bounded-error protocol evaluating C: Use common randomness to approximate the output gate by a $\mathsf{GF}(2)$-polynomial of polylogarithmic degree. Alice and Bob can evaluate this polynomial, provided Bob has space for messages of length $\mathsf{polylog}(n)$, and Bob has at least two memory states. They also can evaluate f, by precomputing $u = A(x)$ and $v = B(y)$ and then evaluating the $\mathsf{GF}(2)$-polynomial with u and v as input.

We show how to make this protocol zero-error. The last four bullet points of Lemma 16 imply the following: If $g(x) = p_g(x)$ for all $\vee$-gates $g \in C$, then indeed $g(x) = p_g(x)$ for all gates. In other words, errors can only be introduced at $\vee$-gates. Such an error is introduced at $\vee$-gate g if and only if $p_g(x) = 0$ and $p_{c_1}(x) \vee \cdots \vee p_{c_t}(x) = 1$, where $c_1, \ldots, c_t$ are the children of g. That is, if and only if

$$\mathcal{E}_g(x) \overset{\text{def}}{=} \bigvee_{i=1}^{t} (\neg p_g(x)) \wedge p_{c_i}(x) \ .$$

evaluates to 1. Note that each disjunct in this expression is itself a polynomial of polylogarithmic degree. We conclude: If there is a gate g in C such that $g(x) \neq p_g(x)$, then

$$\mathcal{E}(x) \overset{\text{def}}{=} \bigvee_{\vee-\text{gates } g} \bigvee_{i=1}^{t} (\neg p_g(x)) \wedge p_{c_i}(x)$$

evaluates to 1.

We are ready to state our zero-error protocol for evaluating the circuit C. Alice and Bob evaluate $\mathcal{E}(x)$ by iterating over all $\vee$-gates g and all children c_i of g. They can evaluate the $\mathsf{GF}(2)$-polynomial $(\neg p_g(x)) \wedge p_{c_i}(x)$ with two memory states and $\mathsf{polylog}(S)$ maximum message length. If this polynomial evaluates to 1, Bob immediately outputs "don't know" and stops. If they iterate over all $\vee$-gates g and c_i without Bob outputting "don't know", they know that $\mathcal{E}(x) = 0$. This implies that $g(x) = p_g(x)$ for all gates g in the circuit, including the output gate. Thus, they evaluate $p_g(x)$ for the output gate g and output its value.

What is the probability that Bob answers "don't know"? This can only happen if some error is introduced at a $\vee$-gate. This happens with probability at most $1/3$. Finally, Alice and Bob evaluate f by evaluating C on $A(x)$ and $B(y)$. This establishes a zero-error protocol for $f \in \mathsf{PH}^{cc}$ and completes the proof of Theorem 15. ■

B. Derandomization of Zero-Error Protocols

Theorem 17 (Derandomization of Zero-Error Protocols)**.** *Suppose there is a zero-error protocol computing f with maximum message length s and w memory states. Then there is a deterministic protocol computing f with maximum message length $s + \lceil \log(3n) \rceil$ and $w + 1$ memory states.*

The idea behind this proof is that Alice and Bob can iterate over all possible random strings r and simulate the zero-error protocol with this random string. If the simulated zero-error protocol causes the simulated Bob to output "don't know", Bob can remember this by moving into his $(w+1)^{\text{th}}$ state. Alice and Bob can afford to iterate over all random strings only if the number of random strings is small. Here, we use a technique similar to the proof of Newman's Theorem that replaces public by private randomness ([15], see also Theorem 3.14 of [11]). We now prove Theorem 17, and Theorem 7 clearly follows as a corollary of Theorem 15 and Theorem 17.

Proof of Theorem 17: Suppose Alice and Bob compute f using a zero-error protocol with maximum message length s in which Bob has w memory states. Denote by $P(x, y, r)$ the output of that protocol on input (x, y), when the common random string is r. Clearly, $P(x, y, r) \in \{0, 1, \text{"don't know"}\}$. Call the random string r *good for input* (x, y) if $P(x, y, r) \in \{0, 1\}$. That is, P computes f correctly. Since P is zero-error, $\Pr[r \text{ is good for } (x, y)] \geq 1/2$. Sample $3n$ random strings $r^{(1)}, \ldots, r^{(3n)}$ independently. Consider an input $(x, y) \in \{0, 1\}^n \times \{0, 1\}^n$.

$$\Pr[\text{none of } r^{(1)}, \ldots, r^{(3n)} \text{ is good for } (x, y)] \leq 8^{-n} \ .$$

Thus, by a union bound over $\{0, 1\}^n \times \{0, 1\}^n$, we obtain

$$\Pr[\exists (x, y) : \text{ none of the } r^{(i)} \text{ is good for } (x, y)] \leq 2^{-n} \ .$$

Thus, there exists a fixed choice $r^{(1)}, \ldots, r^{(3n)}$ of strings such that for each $(x, y) \in \{0, 1\}^n \times \{0, 1\}^n$, at least one such string is good.

We define a new, deterministic protocol P'. In P' Alice and Bob now iterate over all these random strings $r^{(1)}, \ldots, r^{(3n)}$. For each $1 \leq i \leq 3n$, they simulate the zero-error protocol $P(x, y, r^{(i)})$. If in this simulated protocol, Bob outputs 0 or 1, then in P', Bob outputs the same. If in the simulated protocol, Bob outputs "don't know", then in P' Bob moves to its $(w + 1)^{\text{th}}$ memory state to remember that this current simulation has failed. Since there is some $r^{(i)}$ that is good for (x, y), Alice and Bob will at some point arrive at a simulation that does not fail, but outputs the correct value. In addition to the maximum message length s of P, in protocol P' Alice uses additional $\lceil \log(3n) \rceil$ bits in each round to tell Bob the index i of the current random string. ■

VI. PSPACE^{cc} AND 5-STATE PROTOCOLS

According to Theorem 5, if instead of three we have five or more states and the maximum message length is polylogarithmic, then we *fully* characterize PSPACE^{cc}. As mentioned before, this is the first characterization of PSPACE^{cc} in terms of "space".

Theorem 5 (restated). $\mathsf{PSPACE}^{cc} = \mathsf{SPACE}_{\mathsf{LTD}}[\mathsf{polylog}(n), 5]$. *Moreover, the same holds true when replacing 5 with a larger constant.*

Theorem 6 (restated). *Suppose $f \in \mathsf{NC}^1$. Then $f \in \mathsf{SPACE}_{\mathsf{LTD}}[O(\log n), 5]$ (under every input partition).*

Proof of Theorem 5 and Theorem 6: Let $f \in \mathsf{PSPACE}^{cc}$. By the connection between circuits and communication classes (Observation 11), Alice and Bob can preprocess their inputs x, y and then evaluate a circuit of $\mathsf{polylog}(n)$ depth and $2^{\mathsf{polylog}(n)}$ size. To be more precise, there is $m \leq 2^{\mathsf{polylog}(n)}$, functions $A, B : \{0, 1\}^n \to \{0, 1\}^m$ and a circuit C on $2m$ variables of depth $\mathsf{polylog}(n)$ such that $f(x, y) = C(A(x), B(y))$.

Theorem 18 (Generalized Version of Barrington's Theorem)**.** *There exists a universal constant $c \in \mathbb{N}^+$ such that if a function f can be computed by a depth d fan-in 2 circuit over basis $\{\vee, \wedge, \neg\}$, it can also be computed by a width 5 length c^d branching program.*

Thus, there is a width-5 branching program of length $2^{\mathsf{polylog}(n)}$ that computes C. Alice and Bob use A and B to preprocess their inputs into $A(x)$ and $B(y)$. Then we use Fact 9 to evaluate C on inputs $A(x), B(y)$. This uses five memory states and has maximum message length $\mathsf{polylog}(n)$.

This also proves Theorem 6, as Barrington's theorem transforms an NC^1-circuit into a width-5 branching program of polynomial length. Using Fact 9, Alice and Bob can simulate this with 5 memory states $\log(n)$ maximum message length.

For the converse of Theorem 5, let $f \in \mathsf{SPACE}_{\mathsf{LTD}}[\mathsf{polylog}(n), 5]$. That is, f is computed by a protocol $\mathcal{P}$ of maximum message length $S(n) \leq \mathsf{polylog}(n)$. We will show that $f \in \mathsf{PSPACE}^{cc}$, which amounts to constructing a quantified boolean formula as in Definition 4.

Using the same notation as in Definition 6, for a fixed input (x, y), the configuration C of a particular round of $\mathcal{P}$ is denoted by $C \overset{\text{def}}{=} (\alpha, i, \gamma)$, where i is the round number, α is the message sent by Alice, and γ is Bob's memory state. Such a configuration is x-consistent if $P_A(i, x) = \alpha$. Two configurations $C_1 \overset{\text{def}}{=} (\alpha_1, i_1, \gamma_1)$ and $C_2 \overset{\text{def}}{=} (\alpha_2, i_2, \gamma_2)$ are adjacent if $i_2 = i_1 + 1$ and there exists $\beta \in \{0, 1, \perp\}$ such that $T_B(y, \gamma_1, \alpha_2) = (\beta, \gamma_2)$.

Therefore, to characterize the condition that $f(x, y) = 1$, we only need to certify that there is a sequence of configurations $C_0, C_1, \ldots, C_l$ such that

- $l \leq 5 \cdot 2^{S(n)}$
- for each $i \in \{1, 2, \ldots, l\}$, C_i is x-consistent
- for each $i \in \{0, 1, \ldots, l-1\}$, C_i and C_{i+1} is adjacent
- $C_0 = (\emptyset, 0, \gamma_0)$ is the designated initial configuration, γ_0 is Bob's designated initial memory state, and $\emptyset$ is a placeholder
- for each $i \in \{0, 1, \ldots, l-2\}$, C_i and C_{i+1} do not lead Bob to output 0
- C_{l-1} and C_l lead Bob to output 1

We encode this computation (sequence of configurations) as a quantified boolean formula of length a logarithm of the computation size by (i) existentially guessing the middle configuration and (ii) universally asserting the correctness of both parts – i.e. exactly as in the standard theorem by Stockmeyer and Meyer [23] (see also 4.13 in [20]). The only thing to observe and conclude is that certifying correctness for Alice through ψ and for Bob through ϕ can be done completely independently which means that the unquantified formula is always in the form $\psi(u, x) \diamond \phi(u, y)$. ■

VII. Proof of Theorem 10

Theorem 10 (restated). *Suppose $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ can be decomposed as $f(x, y) = g(h_1(x, y), \ldots, h_k(x, y))$ where each $h_i : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ depends on at most ℓ bits from each of its two n-bit inputs. Then $f \in \mathsf{SPACE}_{\mathsf{LTD}}[\ell + k + \lceil \log k \rceil, 2]$.*

Proof: Suppose $f(x, y)$ can be decomposed as $g(h_1(x, y), \ldots, h_k(x, y))$ where each h_i depends on at most ℓ variables. We give a protocol with maximum message length $k + \ell + \lceil \log k \rceil$ where Bob has two memory states.

The protocol proceeds in phases. In each phase, Alice and Bob have a "working hypothesis" $z \in \{0,1\}^k$ and want to verify that $h_i(x, y) = z_i$ for all $1 \leq i \leq k$. To do so, Bob first initializes himself into his first memory state, then they use k rounds of communication. In round i, they evaluate $h_i(x, y)$. Note that h_i depends on at most ℓ variables in x. Alice sends those bits, and Bob can evaluate $h_i(x, y)$. If $h_i(x, y) \neq z_i$, he moves into his second memory state, remembering that z was a wrong hypothesis. Once i reaches k, Bob remembers whether the z was the correct hypothesis or not. If it is was not, he moves back into his first memory state and continues, and they move on to the next $z \in \{0,1\}^k$. Otherwise, he knows that $z_i = h_i(x, y)$ for all $1 \leq i \leq k$, and outputs $g(z_1, \ldots, z_k)$.

In each step, Alice sends Bob up to ℓ bits of her own input. Additionally, she has to send Bob the "guess" $z \in \{0,1\}^k$, and finally the counter $i \in \{1, \ldots, k\}$. Thus, her message consists of at most $\ell + k + \lceil \log k \rceil$ bits.

■

VIII. Future Directions

We are just beginning to understand the power of the one-way limited memory model. For models that are not completely memoryless we cannot prove any lower bounds so far. Given the various connections between this model and the communication complexity polynomial hierarchy, and the fact that separating classes higher up in the communication complexity hierarchy remains an open problem (e.g. as far as we know, it is open whether $\Sigma_2^{cc} = \Pi_2^{cc}$), any new lower bound technique would bring substantial progress.

We do not have lower bounds on *randomized* versions of memoryless protocols. Indeed, the most natural randomized version contains the class AM^{cc} (see Babai et al. [2] or Klauck [24]); proving lower bounds on AM^{cc} would be significant progress and is considered a difficult problem.

Another direction that is worth exploring is the study of space-communication tradeoffs. For example, tradeoffs between the maximum message length required by a memoryless protocol and the total amount of communication. More concrete: Is there a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ such that $OS(f) \leq \mathsf{polylog}(n)$, but every protocol with $\mathsf{polylog}(n)$ maximum message length has a superpolynomial total amount of communication?

Acknowledgements

We would like to thank Paul Beame, Kristoffer A. Hansen, Aram Harrow for helpful comments and suggestions on this topic.

We are also thankful to Paul Beame for communicating Shearer's proof of width-2 branching program length lower bound of "modulo 3" function.

We want to thank Prahladh Harsha and Andrej Bogdanov: The idea to look at Gap Hamming Distance came up during a discussion with them. Dominik Scheder is especially thankful to Justin Thaler, who pointed him to several related results.

This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61361136003, 61350110536.

Dominik Scheder acknowledges support from the Danish National Research Foundation and The National Science Foundation of China (under the grant 61061130540) for the Sino-Danish Center for the Theory of Interactive Computation, within which this work was performed. He also acknowledges support from the Simons Institute for the Theory of Computing, UC Berkeley.

References

[1] J. Brody, S. Chen, P. A. Papakonstantinou, H. Song, and X. Sun, "Space-bounded communication complexity," in *ITCS*, R. D. Kleinberg, Ed. ACM, 2013, pp. 159–172.

[2] L. Babai, P. Frankl, and J. Simon, "Complexity classes in communication complexity theory (preliminary version)," in *FOCS*. IEEE Computer Society, 1986, pp. 337–347.

[3] S. Aaronson and A. Wigderson, "Algebrization: A new barrier in complexity theory," *TOCT*, vol. 1, no. 1, 2009.

[4] N. Alon, Y. Matias, and M. Szegedy, "The space complexity of approximating the frequency moments," *Journal of Computer and System Sciences*, vol. 58, no. 1, pp. 137–147, 1999, preliminary version in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pages 20–29, 1996.

[5] P. Indyk and D. Woodruff, "Tight lower bounds for the distinct elements problem," in *Proceedings of the 45th Annual Symposium on Foundations of Computer Science*, 2003, pp. 283–289.

[6] M. Karchmer and A. Wigderson, "Monotone circuits for connectivity require super-logarithmic depth," in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, 1988, pp. 539–550.

[7] R. Raz and A. Wigderson, "Monotone circuits for matching require linear depth," *Journal of the ACM*, vol. 39, no. 3, pp. 736–744, 1992.

[8] E. Blais, J. Brody, and K. Matulef, "Property testing lower bounds via communication complexity," in *Proceedings of the 26th Annual IEEE Conference on Computational Complexity*, 2011, pp. 210–220.

[9] R. Impagliazzo, V. Kabanets, and A. Kolokolova, "An axiomatic approach to algebrization," in *STOC*, M. Mitzenmacher, Ed. ACM, 2009, pp. 695–704.

[10] R. Impagliazzo and R. Williams, "Communication complexity with synchronized clocks." in *Proceedings of the 25th Annual IEEE Conference on Computational Complexity*, 2010, pp. 259–269.

[11] E. Kushilevitz and N. Nisan, *Communication Complexity*. Cambridge University Press, 1997.

[12] D. A. Barrington, "Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1," *J. Comput. Syst. Sci.*, vol. 38, no. 1, pp. 150–164, 1989.

[13] A. A. Razborov, "Lower bounds on the size of bounded depth networks over a complete basis with logical addition (Russian)," *Matematicheskie Zametki*, vol. 41, no. 4, pp. 598–607, 1.

[14] R. Smolensky, "Algebraic methods in the theory of lower bounds for boolean circuit complexity," in *STOC*, A. V. Aho, Ed. ACM, 1987, pp. 77–82.

[15] I. Newman, "Private vs. common random bits in communication complexity," *Inf. Process. Lett.*, vol. 39, no. 2, pp. 67–71, 1991.

[16] M. Braverman, "Poly-logarithmic independence fools bounded-depth boolean circuits," *Commun. ACM*, vol. 54, no. 4, pp. 108–115, 2011.

[17] E. Allender and U. Hertrampf, "Depth reduction for circuits of unbounded fan-in," *Inf. Comput.*, vol. 112, no. 2, pp. 217–238, 1994.

[18] Shearer, "Lower bound for width-2 branching programs," private communication with Paul Beame.

[19] P. A. Papakonstantinou, D. Scheder, and H. Song, "Overlays and limited memory communication mode(l)s," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 20, p. 189, 2013.

[20] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[21] H. Vollmer, *Introduction to Circuit Complexity - A Uniform Approach*. Springer, 1999.

[22] T. W. Lam and W. L. Ruzzo, "Results on communication complexity classes," *J. Comput. Syst. Sci.*, vol. 44, no. 2, pp. 324–342, 1992.

[23] L. J. Stockmeyer and A. R. Meyer, "Word problems requiring exponential time: Preliminary report," in *STOC*, A. V. Aho, A. Borodin, R. L. Constable, R. W. Floyd, M. A. Harrison, R. M. Karp, and H. R. Strong, Eds. ACM, 1973, pp. 1–9.

[24] H. Klauck, "On Arthur Merlin games in communication complexity," in *IEEE Conference on Computational Complexity*. IEEE Computer Society, 2011, pp. 189–199.