COMMUNICATION COMPLEXITY TOWARDS LOWER BOUNDS ON CIRCUIT DEPTH

Jeff Edmonds, Russell Impagliazzo, Steven Rudich, and Jiří Sgall

Abstract. Karchmer, Raz, and Wigderson, 1991, discuss the circuit depth complexity of n bit Boolean functions constructed by composing up to $d = \log n / \log\log n$ levels of $k = \log n$ bit boolean functions. Any such function is in AC^1. They conjecture that circuit depth is additive under composition, which would imply that any (bounded fan-in) circuit for this problem requires $dk \in \Omega(\log^2 n / \log\log n)$ depth. This would separate AC^1 from NC^1. They recommend using the communication game characterization of circuit depth. In order to develop techniques for using communication complexity to prove circuit lower bounds, they suggest an intermediate communication complexity problem which they call the Universal Composition Relation. We give an almost optimal lower bound of $dk - O(d^2(k \log k)^{1/2})$ for this problem. In addition, we present a proof, directly in terms of communication complexity, that there is a function on k bits requiring $\Omega(k)$ circuit depth. Although this fact can be easily established using a counting argument, we hope that the ideas in our proof will be incorporated more easily into subsequent arguments which use communication complexity to prove circuit depth bounds.

Keywords. Communication complexity, Boolean complexity, lower bounds.

Subject classification. 68Q15, 68Q17.

1. Introduction

Karchmer & Wigderson (1990) observed that the circuit depth (over the basis $\wedge, \vee, \neg$) of a Boolean function f, $Depth(f)$, can be characterized as the communication complexity of a certain game R_f. For communication game G, we use the notation $CC(G)$ to represent the communication complexity of G, i.e., the number of bits communicated during the course of the best protocol solving G on the worst-case instance of G. In this notation, $CC(R_f) = Depth(f)$.

Based on this equivalence, Karchmer, Raz & Wigderson (1995) sketched a plan of attack for proving a super-logarithmic lower bound for the circuit depth required to compute a specific function in P. (In fact, the function is in AC^1.) This function, which we will refer to as the **Iterated Multiplexor**, can be described as follows. The function has two parameters k and d, which, for optimal results, are set at $k = \log n$ and $d = \log n / \log\log n$. (All logarithms in this paper are base 2.) The input to the function can be thought of as a complete ordered k-ary tree with depth $d+1$, counting the root as depth 1. Each of the k^d leaves is labeled with an input bit. Every interior (i.e., non-leaf) node of the tree is labeled by a 2^k input bit string, which is thought of as explicitly describing a function from $\{0,1\}^k$ to $\{0,1\}$. This initial labeling induces a labeling of every node by a bit as follows. The induced bit of a leaf is just its original label. If an interior node u is labeled by a function f_u, and its children, in order from left to right, have induced bits $b_1, \ldots, b_k$, then the induced bit for u is $b_u = f_u(b_1, \ldots, b_k)$. The output of the Iterated Multiplexor is the induced bit of the root. The input to the function is of length $k^d + 2^k \cdot \sum_{i=0}^{d-1} k^i \in O(n^2)$. The Iterated Multiplexor function can easily be seen to be in AC^1.

Suppose that for each node of the tree, the 2^k bits describing the function are fixed, and moreover on each level all the functions are identical. Then the Iterated Multiplexor becomes a composition of these functions. For example, consider the case $d = 3$, where the nodes at level $i \leq 3$ are labeled by f_i. Then the restricted function has k^3 boolean inputs, which we can partition into k^2 vectors of length k, $\vec{x}_{ij}$, $i, j \in \{1, \ldots, k\}$. We can write the restricted function as $f_1(f_2(f_3(\vec{x}_{11}), \ldots, f_3(\vec{x}_{1k})), \ldots, f_2(f_3(\vec{x}_{k1}), \ldots, f_3(\vec{x}_{kk})))$. Intuitively, the circuit depth complexity of this function should be the sum of that for f_1, f_2, and f_3, since any circuit should have to compute at least some of the values for f_3 before going on to f_2, etc.

To prove an $\Omega(\log^2 n / \log\log n)$ depth lower bound for the Iterated Multiplexor function, it suffices to prove that the circuit depth of the composition of functions is roughly the sum of the depths required for each of the functions involved: if the functions being composed are randomly chosen k bit functions, then they require depth $k - o(k)$. Hence, if there are $d = \log n / \log\log n$ levels of these, then the necessary circuit depth should be approximately $dk \in \Omega(\log^2 n / \log\log n)$.

There are no known examples of functions where the depth of the composition is smaller than the sum of the depths. For example, if all of the functions are parity on n bits, the composition of d such functions is parity on n^d bits. The depth of the composition is then roughly $2\log n^d = d(2\log n)$, d times the depth of the function being composed. This also shows that to gain via composition, we need to start with a function not in NC. The iterated Multiplexor construction gets around this, by composing arbitrary functions on $O(\log n)$ bits, rather than fixed functions.

Proving that circuit depth is additive under composition seems extremely difficult, even for random functions.

Therefore, Karchmer *et al.* (1995) suggested an abstraction of the communication game for the Composition of Functions. They call this game the **Universal Composition Relation**, since any protocol solving this problem will also solve the associated communication game for any composition of functions. (Thus, a lower bound for any composition implies the same lower bound for the Universal Composition Relation.) In addition, the known protocols for the communication game for a composition of randomly chosen functions also apply to the Universal Composition Relation. Therefore, the hope is that a lower bound for the Universal Composition Relation problem will yield insight into the circuit depth of compositions; it is at least a necessary first step.

We now give an informal description of the Karchmer-Wigderson communication game (Karchmer & Wigderson 1990) and the Universal Composition Relation. Let f be a boolean function on k bit inputs. The **Karchmer-Wigderson** game for f, R_f is as follows. One player, called the A-Player, is given a k bit vector x for which $f(x) = 0$, and the other, the B-Player, a y for which $f(y) = 1$. The object of the game is to find a bit position i where the two differ, i.e., $x_i \neq y_i$.

Consider the harder game, which we call the Universal Game, where as before, the two players are each given a k bit vector and must find a differing bit position. However, they are only promised that their vectors are different. It is known that the complexity of this game is between $k-1$ and $k+2$. The lower bound follows similarly as the lower bound for testing equality, and we will present an adversary argument for it later. For the upper bound, it is easy to achieve $k + \log k$: one player sends his input and the other replies by the position of a different bit. To achieve $k+2$, we use the following protocol. The players alternate in sending one bit each, from the beginning of their inputs (i.e., the A-player sends his bit 1, then the B-player sends his bit 2, then the A-player sends his bit 3, etc.) However, if any player sees that the received bit is not equal to his corresponding bit, he "raises his hand" as follows: in the next round he sends 1, and in all the following rounds he sends 0. This way the players exchange the total of k bits. Now, each of them sends a bit which is 0 if he raised his hand before, and 1 otherwise. If no player raised the hand, the only difference must be the bit k (using the promise that the inputs are not equal). If one of the players raised his hand, there is a difference just before the last 1 sent by him. If both players raised their hands, we take the earlier of the two positions where they last sent 1. We can read out the answer from the communicated $k+2$ bits. (For more results on the complexity of this game see Tardos & Zwick 1997.)

The communication complexity of R_f for a random function f is almost the same as for the Universal Game: $k - o(k)$. Basically, one player needs to communicate his entire input to the other player. The difference between the games is that in the Universal Game, the players are merely promised that their inputs are different, whereas in R_f, they are also told a particular way in which their inputs differ. Since the games are of comparable difficulty, the moral would seem to be that being told that, for your inputs x and y, $f(x) \neq f(y)$ for a random function f does not convey any more useful information than being told $x \neq y$.

The Universal Composition Relation is obtained by applying this intuition in an analogous way to the Karchmer-Wigderson game for a composition of d random functions $f_1, \ldots, f_d$.

The composition function has k^d inputs, which can be thought of as labeling the leaves of a complete ordered k-ary tree of depth $d+1$. Similarly as for the Iterated Multiplexor function, any input induces a labeling of every node of the tree. If a node u at level $i \leq d$ has children with labels $b_1, \ldots, b_k$, from left to right, then the induced label for u is $f_i(b_1, \ldots, b_k)$. The two players in the Karchmer-Wigderson game are each given such a labeled tree, with the labels at the root different. Their goal is to find an input bit (i.e., leaf) where their inputs differ.

The obvious protocol for this problem is as follows. The protocol is divided into d rounds, each requiring $k + O(1)$ bits of communication. Initially, the players know that the root (level 1) has different labels in their two labelings. At the beginning of the r^{th} round, the players agree on a node at level r that has different labels. The players communicate $k + O(1)$ bits to find a difference in labels of children of this node. Such a difference must exist, since both players compute the node's label from the labels of the children in the same way. This begins the next round. After d rounds the players agree on a leaf labeled differently. Hence an upper bound on the communication complexity of the game for the composition of any d functions on k bit inputs is $kd + O(d)$.

The protocol only uses the facts that the two roots have different labels and that, if a node is labeled differently in the two inputs, then one of its children is also labeled differently. It follows that one of the leaves is labeled differently. Thus, this upper bound also holds for the following game.

Definition 1.1. *The* **Universal Composition Relation with parameters** d **and** k, *denoted* $UCR_{d,k}$, *is defined as follows. Each player has as input a boolean labeling of all the nodes of an ordered complete* k*-ary tree of depth* $d+1$, *counting the root as depth 1. The label of the root for the A-Player is 0 and for the B-Player is 1. The two inputs must satisfy the following condition: for every interior node of the tree, if the node is labeled differently in the two players' inputs, then at least one of the children of the node is labeled differently. The goal of the players is to agree on a leaf node which is labeled differently in their two trees.*

In this paper, we prove the following lower bound:

Theorem 1.2. $CC(UCR_{d,k}) \geq dk - O(d^2(k \log k)^{1/2})$

This bound is almost tight for $d = o((k/logk)^{1/2})$. If this same bound were proven for the Iterated Multiplexor Function, it would yield a lower bound of $\Omega((\log n)^{3/2}(\log\log n)^{-1/2})$ on circuit depth by setting $d = \varepsilon(k/logk)^{1/2}$ for a suitable constant $\varepsilon > 0$. The goal now, in order to separate AC^1 from NC^1 is to use these techniques to prove the lower bound for the Iterated Multiplexor function. This motivated us to re-examine the case for $d = 1$. We give a proof directly in terms of communication complexity that there is a function on k bits requiring $\Omega(k)$ circuit depth, i.e., $CC(R_f) = \Omega(k)$. Although this fact can be easily established using a simple counting argument, our proof has the advantage of being an "adversary argument," which is more in line with the proof techniques used for the Universal Composition Relation. We hope our results will provide a step towards a theory for circuit depth.

An extended abstract of this paper appeared in Edmonds *et al.* (1991). Subsequently, using different (and highly interesting) techniques, Håstad & Wigderson (1993) obtained a lower bound of $dk - 2^{O(d)}$ for $UCR_{d,k}$. This is better for small values of d. However, even if extended to the actual composition of functions, this would give a depth bound of at most $\Omega(\log n \log\log n)$, whereas ours would give one of the form $\Omega((\log n)^{3/2}/(\log\log n)^{1/2})$. Recently Raz & McKenzie (1999) used a notion similar to predictability (our main concept) to prove a separation of monotone P from monotone NC. Unfortunately, no progress has been made in using our techniques for their original purpose, proving circuit depth lower bounds. It would be interesting to see if this is because our techniques are "natural" in the sense of Razborov & Rudich (1997). While we do not see how to use a general lower bound on compositions to derive a "natural" property, such a property might be implicit in our proofs.

In Section 2, we give a formal definition of communication complexity, and review some of its basic properties and connections to circuit complexity. In Sections 3 to 6, Theorem 1.2 is proved. First, in Section 3, we describe the argument for the depth two informally. In Section 4 we develop our main technical tool, the concept of predictability. The formal proof is presented in Sections 5 and 6, separately for the depth two and general depth. Section 7 gives the new proof of the existence of a function requiring linear depth circuits to compute.

2. Communication Complexity

For a general reference on communication complexity see Kushilevitz & Nisan (1996). A communication protocol is a method by which two parties, each with a private input, compute a function of the two inputs by sending messages to each other. Each message should be a single bit. Who speaks next should be determined by the conversation so far, and the output of the protocol should be determined by the total conversation. Formally,

DEFINITION 2.1. *Let X, Y, and Z be finite sets. Then a* **two-person t-message communication protocol** *consists of a function T from $\{0,1\}^{<t}$ to $\{$"A-player", "B-player"$\}$, a function P^A from $X \times \{0,1\}^{<t}$ to $\{0,1\}$, a function P^B from $Y \times \{0,1\}^{<t}$ to $\{0,1\}$, and a function Out from $\{0,1\}^t$ to Z. For $x \in X, y \in Y$, the* **conversation up to message i** *determined by the protocol on inputs x, y is given by $C_i = C_i(x,y)$ where $C_0 = \varepsilon$ and $C_{i+1} = (C_i, P^A(x, C_i))$ if $T(C_i) =$ "A-player", and $C_{i+1} = (C_i, P^B(y, C_i))$ otherwise. The* **output** *of the protocol on inputs x, y is $Out(C_t(x,y))$.*

We will use the following general property of communication protocols.

DEFINITION 2.2. *Let P be a t-message communication protocol on input sets X and Y. Let $i \leq t$, and let $C \in \{0,1\}^i$. Let $x \in X, y \in Y$. We say that the pair $\langle x, y\rangle$ is consistent with C if $C_i(x,y) = C$. We will denote the set of pairs consistent with C by S_C. We will let S_C^A be the set of $x \in X$ so that there is a $y \in Y$ with $\langle x,y\rangle \in S_C$, and similarly $S_C^B = \{y \mid \exists x \in X, \langle x,y\rangle \in S_C\}$.*

LEMMA 2.3 (Kushilevitz & Nisan 1996,Proposition 1.14). *For any communication protocol and any $C \in \{0,1\}^i$, $S_C = S_C^A \times S_C^B$*

We are concerned with communication protocols for problems of the following form. We are promised that a certain relationship holds between the two inputs. If the relationship holds, the protocol should output a string which bears a certain relationship to the pair. Formally, Definition 2.4. *A* **communication task T** *on sets X, Y, Z consists of a relation $R : X \times Y \to \{0,1\}$ and a relation $R' : X \times Y \times Z \to \{0,1\}$, so that for every x, y satisfying $R(x,y)$, there exists a $z \in Z$ with $R'(x,y,z)$. A protocol* **performs** *communication task T if, for every x, y satisfying $R(x,y)$, the output satisfies $R'(x,y,Out(C_t(x,y)))$. The* **communication complexity** *of a communication task T, denoted $CC(T)$, is the least t such that there is a t-message protocol performing the task.*

For every Boolean function f on k bit strings, we can associate a communication task whose complexity is exactly the minimum depth, $D(f)$, of a circuit over the basis $\{\wedge, \vee, \neg\}$ that computes f. The task is, given inputs $\langle x, y\rangle$, for which $f(x) = 0$ and $f(y) = 1$, to find a bit position where x and y differ.

Definition 2.5. *Let f be a function from $\{0,1\}^k$ to $\{0,1\}$. The* **communication task for** *f, denoted R_f, is given by $X = Y = \{0,1\}^k$, $Z = \{1, \ldots, k\}$, $R(x,y)$ holds if and only if $f(x) = 0$ and $f(y) = 1$, and $R'(x,y,z)$ holds if and only if $x_z \neq y_z$, i.e., x and y differ in the bit position z.*

This task characterizes the depth of a formula computing f:

Theorem 2.6 (Karchmer & Wigderson 1990). *$CC(R_f) = D(f)$.*

The following theorem is proved using a counting argument, comparing the number of Boolean circuits of depth d to the number of functions on $\{0,1\}^k$.

Theorem 2.7 (Riordan & Shannon 1942). *Let f be a Boolean function on $\{0,1\}^k$, chosen randomly. Then, with high probability, $D(f) \geq k - O(\log k)$.*

Finally, we recast Definition 1.1 in this formal setting.

Definition 2.8. *Let k, d be positive integers. Let τ be the complete k-ary tree of depth $d+1$, rooted and ordered. Let τ' be the set of all nodes of τ except for the root, $X = \{0,1\}^{\tau'}$, and $L^A, L^B \in X$. We say the pair L^A, L^B is* **as promised***, if (i) there exist a node i on the second level of τ (i.e., a child of the root) such that $L^A(i) \neq L^B(i) = 1$, and (ii) for every interior node $i \in \tau$ satisfying $L^A(i) \neq L^B(i)$, there is a child i' of i with $L^A(i') \neq L^B(i')$.*

Let Z be the set of leaves of τ. The **Universal Composition Relation** *with parameters d, k, denoted $UCR_{d,k}$, is the communication task on X, X, Z with R being the "as promised" relation and $R'(L^A, L^B, j)$ if and only if $L^A(j) \neq L^B(j)$.*

3. The Case $d = 2$: Overview

The proof for general d involves a great deal of complicated notation and messy details, which tend to obscure the basic intuition. Therefore, we will first present the proof for the special case of $d = 2$. In this section, we give an intuitive overview to motivate the next two sections, which present the proof.

Let us consider the task $UCR_{2,k}$. Each player is given a $\{0, 1\}$ labeling L of the complete k-ary tree of depth 3. Since the root is always labeled 0 in the A-Player's input, and always labeled 1 in the B-Player's, we can ignore the labeling of the root. Let τ_1 be the root of the tree, τ_2 be the k children of the root and τ_3 be the k^2 leaves. For such a labeling L, let L_1 be the induced labeling on τ_2 and L_2 on τ_3. The bit labeling L_2 can either be thought of as k^2 bits indexed by τ_3 or as k vectors indexed by τ_2. For node $i \in \tau_2$, b_i is used to denote the i^{th} bit of L_1 and $\vec{cv}_i$ is used to denote the i^{th} vector of L_2 which labels the children of i. This is called the **child vector of** i. See Figure 3.1.

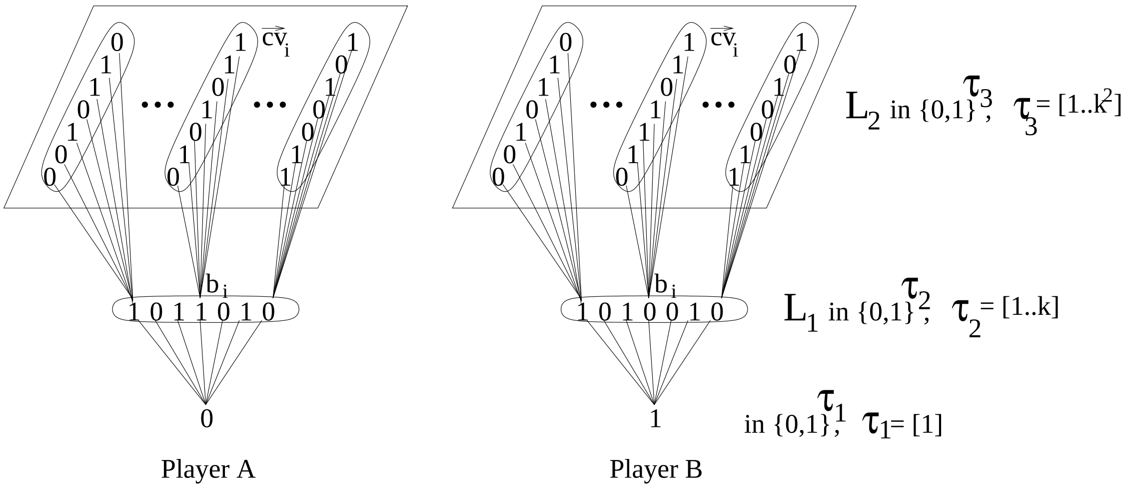

Figure 3.1: The inputs for $UCR_{2,k}$

We would like to show that any reasonable protocol first communicates enough bits to find a difference in the players' L_1 vectors and then finds a difference in the child vector of this differently labeled node. This will take twice as many communication bits as to find a difference in two distinct k bit vectors, i.e., twice as long as the **Universal Game**. A simple argument that the Universal Game requires $k - 1$ bits is as follows.

Each player is given a k bit vector, with the only requirement being that they are different. They must find an index where the vectors are different.

The lower bound will be proved by a simple adversary argument. Given a fixed protocol that is assumed to solve the task in fewer steps, we give a procedure for an adversary to find an input pair on which the protocol fails. The adversary restricts the set of inputs in each round in order to fix the conversation between the players up to the current time step. After a certain number of bits have been communicated, let S^A be the set of vectors x that can be given to the A-Player and be consistent with the conversation so far. Define S^B similarly. The adversary will maximize $|S^A \cap S^B|$ in order to maintain the symmetry between the players. A bit communicated by the A-Player partitions S^A into those vectors on which he communicates a 0 and those on which a 1 is communicated. The adversary chooses the half that keeps the intersection $S^A \cap S^B$ as large as possible. This at most halves the intersection. Similarly, for bits communicated by the B-Player. After t bits, $|S^A \cap S^B| \geq 2^{k-t}$. Hence, after $k-2$ bits $|S^A \cap S^B|$ contains at least four vectors. If the protocol ends at this point and outputs a bit position i, then the adversary can find at least two of the four vectors which are the same at this index (actually, only three vectors are necessary for this). These two vectors are given to the two players. Therefore, the protocol fails on this input. So any protocol solving this task requires $k-1$ bits of communication.

This adversary strategy can be summarized by saying that the adversary delays breaking the symmetry between the players for as long as possible, in order to make it difficult for the players to find a difference. At the last minute, this symmetry has to be broken in order to keep the promise of giving them different vectors.

The same tension between making the players' inputs the same and making them different occurs in our lower bound. The lower bound is broken into stages, one stage per level. In the first stage, the adversary maintains the symmetry between the players' values for both L_1 and L_2. Just before the players can communicate the k bits describing L_1, their L_1 vectors are fixed to be different. In the second stage, the same is done with L_2. The symmetry is kept until just before the players can communicate one child vector in L_2.

The lower bound is more complicated than above, because the players may communicate bits about L_2 during the first stage of the protocol, including hybrid bits involving both the L_1 and L_2 parts of the input or bits which involve many places in the L_2 vector. For simplicity, we will ignore the last two complications and come back to them later.

The intuition is that bits communicated about L_2 during the first stage of the protocol are wasted. The players do not know which of the k^2 bits in L_2 to communicate. On the other hand, if the players wait until a difference in L_1 is known, then they can concentrate on finding a difference in the corresponding child vector within L_2.

Suppose, pessimistically, that the players communicate k bits about L_2 during the first stage. They could either communicate a lot of bits about a few child vectors or a few bits about lots of child vectors. Either way this is not useful for the players. Assume lots of bits have been communicated about a few child vectors. Remember that the adversary only needs to make these child vectors different for the two players if the corresponding bits in L_1 are different. Thus, the adversary will simply make the corresponding bits in L_1 the same, freeing her to fix the entire child vectors in the same way for both players. Thus, the many bits communicated about L_2 could have been replaced by only a few bits about L_1. On the other hand, if only a few bits about any particular child vectors are known, these bits will not significantly help the players once a difference in the L_1's is discovered, and the players focus on one child vector. The dividing line between lots of bits communicated and a few will be set to k/l bits; the parameter l will be set to $\sqrt{k}$ in order to optimize the lower bound. Note there are at most l child vectors for which at least k/l bits have been communicated.

Therefore, the general strategy for the adversary is as follows. During the first $k - l - f$ bits of communication, where f is a small "fudge factor", we assume pessimistically that the players learn $k-l-f$ bits about L_1 and another $k - l - f$ bits about L_2. After this period, the adversary performs a "clean-up" which further restricts the allowable input pairs. The adversary first identifies at most l child vectors for which the players know at least k/l bits, and fixes the corresponding bits in L_1. This leaves only f bits of L_1 unknown; the adversary must now break the symmetry at the L_1 level. If no hybrid bits concerning both L_1 and L_2 have been communicated, this is simple: the adversary picks a different L_1 vector for each player from the 2^f possible vectors. In the real proof, this will be more complicated. Once this symmetry is broken, the parties may know many places in which their L_1's differ, and hence know that the corresponding child vectors are different. But they will not know more than k/l bits about any one of these child vectors. Thus, the adversary will be able to force the protocol to go for about $k - k/l$ more steps before any of these rows are in danger of being forced to be the same for both players. This yields a lower bound of $2k - O(l + k/l) = 2k - O(k^{1/2})$ bits for any protocol solving the task.

The players have more options than in the above discussion. Although it does not appear to help them, instead of communicating a bit about L_1 or a bit about L_2, they could communicate a bit about the combination. Our lower bound gets around this complication by (more or less) converting one combination bit into one bit about L_1 and one bit about L_2. Another complication is that, if the players communicate "hybrid" bits, the number of bits known about a child vector is not well-defined. For this purpose, we need to define a formal measure.

For example, suppose over a sequence of communication bits, the A-Player tells the B-Player, "I am not telling you any of my values, but I will tell you that if the first bit of the first child vector in my L_2 is 0, then all k bits in the second child vector are zero. On the other hand, if this first bit is 1, then I reveal no information about this second child vector." The question now is whether the B-Player is considered to know k or zero bits about this second child vector. A useful information measure for many applications is Entropy. Because half the time k bits about the second child vector are revealed and half the time 0 bits are revealed, Entropy measures the number of bits revealed as the average $k/2$. Our adversary, however, wants to be more cautious by assuming that the B-Player knows more than this. We define a measure of "predictability" to be the probability of guessing the value. If the B-Player completely knows the child vector then the probability of guessing it is 1 and if he knows nothing about it, then the probability is 2^{-k}. The measure is the average of these, $(1+2^{-k})/2 \approx 1/2$. A predictability of $1/2$ is then interpreted to mean that the B-Player knows everything but "1 bit" about this child vector. The next section formally defines this measure.

4. Predictability

Consider a set of vectors $S \subseteq W^\pi$ indexed by the set π, and whose elements lie in set W. For our purposes, W will be the set of k-bit vectors $\{0,1\}^k$, making $v \in S$ a vector of vectors of bits, or, alternately, a matrix of bits. However, we will only use the nature of W in this section to provide some motivating examples of how the Lemmas proved here will be used.

Definition 4.1. *For every vector $v \in S$ and index $i \in \pi$, let $v_i \in W$ be the element of the vector indexed by i.*

For every subset of indices $\rho \subseteq \pi$, let $Proj(v, \rho) \in W^{\pi-\rho}$ be the sub-vector of v indexed by the indices not in ρ (i.e., a projection which removes the elements of v indexed by ρ). Similarly let $Proj(S, \rho) = \{Proj(v, \rho) \mid v \in S\}$ be the projection of S removing the elements indexed by ρ from every vector in S. We write $Proj(v, i)$ instead of $Proj(v, \{i\})$ and $Proj(S, i)$ instead of $Proj(S, \{i\})$.

For every $w \in Proj(S,\rho)$, let $Cons(S,w) = \{v \in S \mid Proj(v,\rho) = w\}$ be those vectors in S consistently extending the vector w to the elements indexed by ρ. A function E from $Proj(S,\rho)$ to S is called an **extension function** *if, for all $w \in Proj(S,\rho)$, $E(w) \in Cons(S,w)$. Note that any extension function is 1-1, since $Proj(E(w),\rho) = w$.*

Suppose t bits have been communicated about an input in W^π and let S be the set of inputs consistent with this conversation. We can safely assume $|S| \geq 2^{-t} \cdot |W^\pi|$. An interesting question is, for $i \in \pi$, how many of these bits were communicated "about the i^{th} element". Since the actual bits communicated could depend on all the elements, this is not a clear-cut issue. Furthermore, if for example the bitwise parity of all the elements were revealed, each element, given the communication, would still be completely uniformly distributed, and so its conditional entropy, given the conversation, would still be large. Thus, we must also condition on the other elements. Note that this gets us into the reverse situation, in that if the protocol communicates the parity of the elements, it simultaneously reduces all of their conditional entropies given the other elements to 0. We will discuss this problem later in this section.

More precisely, look at the following distribution: choose a random vector $v \in S$, and reveal $Proj(v,i)$. We want to measure the conditional randomness of v_i. The set $Cons(S, Proj(v,i))$ contains all vectors v that are consistent with the revealed information. Therefore, the probability of being able to predict the value of v_i is $1/|Cons(S, Proj(v,i))|$. We define the predictability as the expectation of this probability.

DEFINITION 4.2. *The* **predictability** *of the i^{th} element in S is*

$$Pred_i(S) = Exp_{v \in S}\left[\frac{1}{|Cons(S, Proj(v,i))|}\right].$$

If the i^{th} element v_i of $v \in S$ is fixed as a function of the other elements, then $Pred_i(S) = 1$. If this element is completely undetermined, then $Pred_i(S) = 1/|W|$. We can think of $\log(Pred_i(S) \cdot |W|)$ as the "number of bits known about element i", since if S is the set of inputs consistent with t independent bits communicated about v_i, $Pred_i(S) = 2^t/|W|$.

Think of S as being partitioned according to $Proj(v,i)$. The number of equivalence classes is $|Proj(S,i)|$. The size of the equivalence class of v is $|Cons(S, Proj(v,i))|$. The contribution of v towards the expectation in the definition of the predictability is $1/|Cons(S, Proj(v,i))|$. So the total contribution of the vectors in any equivalence class is 1 and the expectation is just the number of classes normalized by the number of vectors. This gives the following lemma which can be used as an alternate definition of predictability.

Lemma 4.3. $Pred_i(S) = |Proj(S,i)|/|S|$. □

Another measure of the amount of information that has been communicated about v_i is called the **collision probability** and is denoted $CP_i(S)$. It is the probability that, for two vectors randomly chosen (with replacement) from S, the i^{th} element is the same, i.e., $CP_i(S) = Prob_{v,v' \in S}[v_i = v'_i]$. The predictability of i is an upper bound on this probability.

Lemma 4.4. $CP_i(S) \leq Pred_i(S)$.

Proof. For any $v \in S$, there are at most $|Proj(S,i)|$ vectors $v' \in S$ such that $v_i = v'_i$. Hence $Prob_{v' \in S}[v_i = v'_i] \leq |Proj(S,i)|/|S| = Pred_i(S)$, and we average over $v \in S$. □

How does throwing away vectors from S affect the predictability?

Lemma 4.5. *Let $S' \subseteq S$. Then*

$$Pred_i(S') \leq \frac{|S|}{|S'|} Pred_i(S).$$

Proof. Note that $Proj(S',i) \subseteq Proj(S,i)$. Therefore,

$$Pred_i(S') = \frac{|Proj(S',i)|}{|S'|} \leq \frac{|Proj(S,i)|}{|S'|} = \frac{|S|}{|S'|} Pred_i(S). \qquad \square$$

Suppose t bits have been communicated about the vectors v in S, i.e., $|S| = 2^{-t} \cdot |W^\pi|$. A natural property to want is that, for all l, at most l elements of v can have more than t/l bits "revealed about it". The problem is that after only $\log(|W|)$ bits have been communicated, it is possible that, for all $i \in \pi$, $Pred_i(S) = 1$, implying that $\log(|W|)$ bits have been communicated "about each of the elements". Consider the following example, for $W = \{0,1\}^k$. Suppose that for $j \in \{1,\ldots,k\}$, the bit j of communication reveals that the parity of the column j is 0. Then for each $i \in \pi$, the row v_i is uniquely determined by the other rows $Proj(v,i)$. Therefore, for all $i \in \pi$, $Pred_i(S) = 1$.

We will get around this problem as follows. When an element becomes highly predictable in S, this element is fixed as a function of the other elements and then ignored. From then on, the set of vectors $Proj(S,i)$ is considered in place of S. In our previous example, if any one of the rows v_i is fixed as a function of the other rows, then no information at all is known about the remaining elements $Proj(v, i)$. The v_i value is fixed as a function of $Proj(v, i)$, by choosing one vector $v \in Cons(S, w)$ for each $w \in Proj(S, i)$. (I.e., we are fixing an extension function which maps $Proj(S, i)$ 1-1 to $S' \subseteq S$.) This is equivalent to the condition that the set of all chosen vectors S' satisfies $|S'| = |Proj(S', i)| = |Proj(S, i)|$, we will use this concise formulation later in the paper. The following lemma says that if at most t bits have been revealed about S, then there exists a set of at most l elements such that, if we fixed them in this way, then no more than t/l bits have been "revealed about" any of the other elements.

LEMMA 4.6. *Let* $|S| \geq 2^{-t} \cdot |W^\pi|$ *and* $|\pi| \geq l > 0$. *Then there exists a subset of elements* $\sigma \subseteq \pi$ *such that*

- $|\sigma| \leq l$, *and*
- $\forall i \in \pi - \sigma,\ Pred_i(Proj(S, \sigma)) < 2^{t/l}/|W|$.

PROOF. Initially, let $\sigma = \emptyset$. We will keep adding indices to σ, maintaining the property that

$$\frac{|Proj(S, \sigma)|}{|W^{\pi-\sigma}|} \geq 2^{-t\left(1-\frac{|\sigma|}{l}\right)}.$$

For $\sigma = \emptyset$, this property is true because $|S| \geq 2^{-t} \cdot |W^\pi|$. Now assume for $\sigma \subseteq \pi$ the property holds and that there is an index $i \in \pi - \sigma$ for which $Pred_i(Proj(S, \sigma)) \geq 2^{t/l}/|W|$. By Lemma 4.3,

$$\frac{|Proj(S, \sigma \cup \{i\})|}{|Proj(S, \sigma)|} = Pred_i(Proj(S, \sigma)) \geq \frac{2^{\frac{t}{l}}}{|W|}.$$

It follows that

$$\frac{|Proj(S, \sigma \cup \{i\})|}{|W^{\pi-\sigma-\{i\}}|} \geq \frac{2^{\frac{t}{l}}}{|W|} \cdot \frac{|Proj(S, \sigma)|}{|W^{\pi-\sigma-\{i\}}|} =$$

$$2^{\frac{t}{l}} \cdot \frac{|Proj(S, \sigma)|}{|W^{\pi-\sigma}|} \geq 2^{\frac{t}{l}} \cdot 2^{-t\left(1-\frac{|\sigma|}{l}\right)} = 2^{-t\left(1-\frac{|\sigma \cup \{i\}|}{l}\right)}.$$

Thus the property holds for $\sigma \cup \{i\}$. Eventually, for all $i \in \pi - \sigma$, we have $Pred_i(Proj(S, \sigma)) < 2^{t/l}/|W|$. Since $Proj(S, \sigma) \subseteq W^{\pi-\sigma}$, it follows that

$$1 \geq \frac{|Proj(S, \sigma)|}{|W^{\pi-\sigma}|} \geq 2^{-t\left(1-\frac{|\sigma|}{l}\right)}$$

and thus $|\sigma| \leq l$. □

At certain points in the argument, we will want to ensure that the players are given vectors that are distinct at many locations. We will do this by partitioning W for each $i \in \pi$ with a random function $g_i : W \to \{0,1\}$, and promising that, for every $i \in \pi$, $g_i(v_i^A) = 0$ and $g_i(v_i^B) = 1$, where v^A and v^B are the inputs for the A-player and the B-Player.

Lemma 4.7. *Suppose for all $i \in \pi$, $Pred_i(S) \leq 1/(4|\pi|m)$. For each $i \in \pi$, pick uniformly at random the functions $g_i : W \to \{0,1\}$. Then $Prob[(\exists v \in S)(\forall i \in \pi) g_i(v_i) = 0] > 1 - |\pi| \cdot 2^{-m}$.*

Proof. For notational convenience, assume, without loss of generality, that $\pi = \{1, \ldots, |\pi|\}$. Let $S^0 = S$ and $S^i = \{v \in S^{i-1} \mid |Cons(S^{i-1}, Proj(v,i))| > m\}$, for $i = 1, \ldots, |\pi|$.

First we prove by induction on i that

$$|S^i| \geq \left(1 - \frac{i}{2|\pi|}\right)|S|.$$

For $i = 0$, it is trivial. Let $1 \leq i \leq |\pi|$. By the induction assumption $|S^{i-1}| \geq |S|/2$. Thus, by Lemma 4.5, $Pred_i(S^{i-1}) \leq 1/(2|\pi|m)$. By Markov inequality,

$$\begin{aligned} &Prob_{v \in S^{i-1}}[|Cons(S^{i-1}, Proj(v,i))| \leq m] \leq \\ &\qquad m \cdot Exp_{v \in S^{i-1}}\left[\frac{1}{|Cons(S^{i-1}, Proj(v,i))|}\right] = m \cdot Pred_i(S^{i-1}) \leq \frac{1}{2|\pi|}. \end{aligned}$$

Thus for a random v chosen uniformly from S^{i-1} the probability of the event that $|Cons(S, Proj(v,i))| \leq m$ (and hence $v \notin S^i$) is at most $1/(2|\pi|)$. Hence

$$|S^i| \geq \left(1 - \frac{1}{2|\pi|}\right)|S^{i-1}| \geq \left(1 - \frac{1}{2|\pi|}\right)\left(1 - \frac{i-1}{2|\pi|}\right)|S| > \left(1 - \frac{i}{2|\pi|}\right)|S|$$

and our claim follows by induction. Applying the claim to $i = |\pi|$, we have that $S^{|\pi|}$ is nonempty.

Now we will construct a sequence of vectors $v^{(|\pi|)}, \ldots, v^{(0)}$ as follows: Let $v^{(|\pi|)} \in S^{|\pi|}$. For $i = |\pi|, \ldots, 1$, select $v^{(i-1)} \in Cons(S^{i-1}, Proj(v^{(i)}, i))$ such that $g_i(v_i^{(i-1)}) = 0$, if possible. Let $v = v^{(0)}$. Since $v^{(i)} \in S^i$, we have $|Cons(S^{i-1}, Proj(v^{(i)}, i))| > m$ and these (more then m) vectors have distinct i^{th} entries. So the probability that g_i will be 1 on all of them is less than 2^{-m}. Therefore, $v^{(0)}$ exists with probability at least $1 - |\pi| \cdot 2^{-m}$, and $g_i(v_i^{(0)}) = 0$, for all $i \in \pi$, since $v_i^{(0)} = v_i^{(i-1)}$ by the construction. □

The following discussion and lemmas formalize the idea that "a hybrid bit communicated concerning all levels of the tree is no worse than a single bit communicated for each level." We consider the general situation where the indices in π are partitioned, as $\pi = \bigcup_{i=r}^{d} \pi_i$. We would like to view S in a way that presents the uncertainty in picking an element of S hierarchically, first picking the information indexed by π_d, then by π_{d-1}, ..., π_r. This gives the following **hierarchy tree** representation of the set S. (One confusion that should be avoided is that, in our example, π itself has a tree structure. The hierarchy tree has the same depth, but it is much larger and the choices in the hierarchy tree are in reverse order, largest index depth first to smallest last.)

Definition 4.8. *Let $r < d$ be integers, let $\pi = \bigcup_{i=r}^{d} \pi_i$ be an ordered partition of π, and let $S \subseteq W^{\pi}$. Define $\pi_{\leq j} = \bigcup_{i=r}^{j} \pi_i$, and $\pi_{\geq j} = \pi - \pi_{\leq j-1}$. Let $S^{\geq j} = Proj(S, \pi_{\leq j-1}) \subseteq W^{\pi_{\geq j}}$. In the degenerate case we set $S^{\geq d+1} = \{\Lambda\}$, where Λ is the single "null" vector (i.e., a vector with 0 coordinates).*

We define the **extension set** *for $L_{\geq j} \in S^{\geq j}$, denoted $X^S(L_{\geq j})$, as the set of all labeling of π_{j-1} which can be consistently added to $L_{\geq j}$, i.e., $X^S(L_{\geq j}) = Proj(Cons(S^{\geq j-1}, L_{\geq j}), \pi_{\geq j})$. (Note that the extension set contains only labelings of a single level, L_{j-1}, even if S contains labelings of more levels.)*

The **hierarchy tree** *H^S for S with respect to this partition is a tree of total depth $d+2-r$. See Figure 4.1. Each node at depth $d+2-j$ is labeled with an element $L_j \in W^{\pi_j}$. Hence, the path from the root to this node is labeled with an element $L_{\geq j} = \langle L_j, \ldots, L_d \rangle \in W^{\pi_{\geq j}}$. The hierarchy tree, however, is pruned so that at this depth it only contains those nodes corresponding to the elements $L_{\geq j} \in S^{\geq j} \subseteq W^{\pi_{\geq j}}$. (Note that the root at depth 1 is Λ, the unique element of $S^{\geq d+1}$, and the leaves at depth $d+2-r$ are the elements of $S^{\geq r} = S$.) The parent of a node $L_{\geq j} \in S^{\geq j}$ for $r \leq j \leq d$ is $L_{\geq j-1} = Proj(L_{\geq j}, \pi_j) \in S^{\geq j+1}$. Its the children are labeled with the elements $L_{j-1} \in X^S(L_{\geq j})$ and correspond to the elements $L_{\geq j-1} = \langle L_{j-1}, L_j, \ldots, L_d \rangle \in Cons(S^{\geq j-1}, L_{\geq j})$.*

We now formalize the intuition that a hybrid bit about a $v \in S$ can be replaced by a simple bit about each level of labeling L_j.

Lemma 4.9. *Let H be a finite tree rooted at Λ, and let $H_1, \ldots, H_p$ be a partition of the leaves of H. Then there is b, $1 \leq b \leq p$ and a non-empty subtree H' of H also rooted at Λ, so that all leaves of H' are in H_b, and, for every interior node $h \in H'$, if h has c children in H, h has at least c/p children in H'.*

Proof. Mark every leaf by a number i between 1 and p according to which set H_i it belongs to. Mark interior nodes in order of decreasing depth, so that

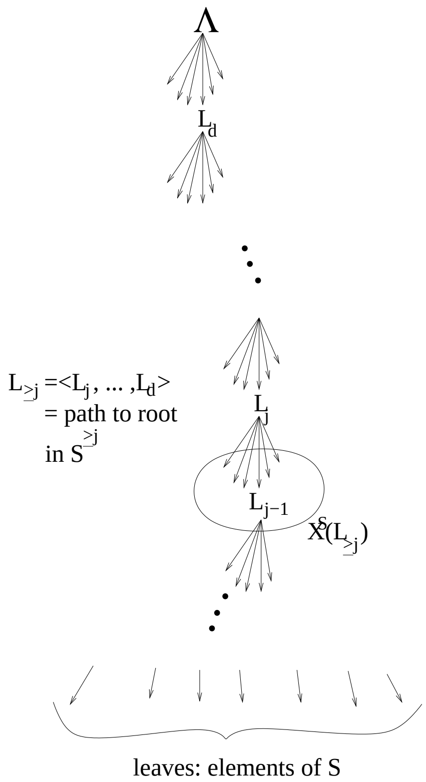

Figure 4.1: The hierarchy tree

each node has the label that appears most frequently at its children (breaking ties arbitrarily). Let b be the label of the root Λ. We define H' as the tree containing all the nodes such that all the nodes on the path from it to the root have label b, i.e., the tree containing the root, its children labeled by b, their children labeled by b, etc. $\square$

Lemma 4.10. *Let $S \subseteq W^\pi$ and let $S_1, \ldots, S_p$ be a partition of S. Then there is b, $1 \leq b \leq p$ and a subset S' of S_b such that for any j, $r \leq j \leq d+1$ and any $L_{\geq j} \in (S')^{\geq j}$, $|X^{S'}(L_{\geq j})| \geq |X^S(L_{\geq j})|/p$.*

Proof. Apply Lemma 4.9 to the hierarchy tree H^S to get a subtree H'. Define S' as the set of all leaves of H'. $\square$

5. Proof for $d = 2$

Now we prove the lower bound. We start with the special case of $d = 2$, since it illustrates the technique while avoiding certain complications. We prove that the players need to communicate at least $2k - 3\sqrt{k} - O(\log k)$ bits.

5.1. Notation. We use the notation from Section 3 and Section 4.

There are two equivalent representations for the inputs to the players. See Figure 5.1. In the **bit** representation, we think of the inputs as bits indexed by a complete k-ary depth 3 tree, $\tau = \tau_1 \cup \tau_2 \cup \tau_3$. The bit of the root which is the sole element of τ_1 is 0 for the A-player and 1 for the B-player. The bit labeling node $i \in \tau$ is denoted by b_i. Then L_1 is the sub-array of bits indexed by τ_2 and L_2 the sub-array of bits indexed by τ_3.

In the **child vector** representation, we think of the inputs as instead labeling the depth 2 tree τ_1, τ_2 with elements from $W = \{0,1\}^k$, the child vectors of the interior nodes in the tree. Thus, the label of the root τ_1 is L_1, and the labels of the leaves τ_2 describe L_2. (The label of the original root is not significant, since this is always 0 for the A-player and 1 for the B-player.) The set of indices is then $\tau' = \tau_1 \cup \tau_2$, the root and its k children respectively. We use the notation $\vec{cv}_i$ to denote the child vector of node $i \in \tau'$. Hence, if i' is the j^{th} child of i in τ, $b_{i'} = \vec{cv}_i(j)$. Throughout the paper we use the child vector representation, unless we explicitly say otherwise.

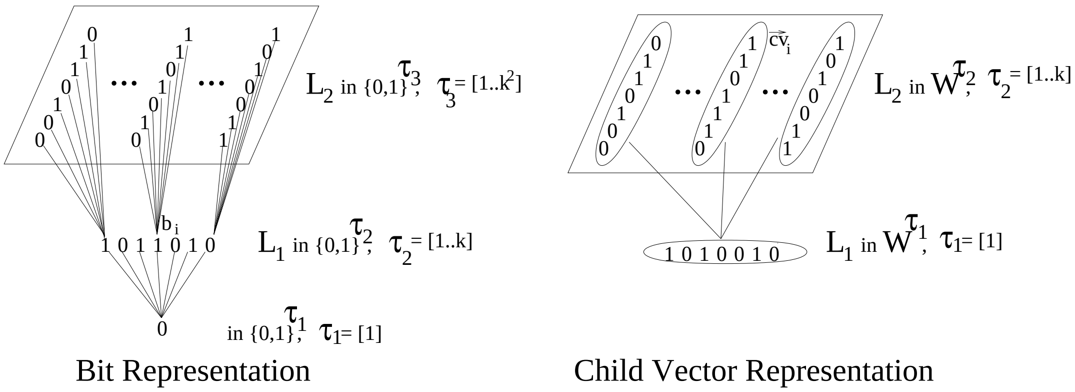

Figure 5.1: The bit and the child vector representation of the inputs

Let $b_i, b'_i, i \in \tau$ be two inputs in bit representation, and let $\vec{cv}_i, \vec{cv}'_i, i \in \tau'$ be the same labelings in child vector representation. We say **the promise is kept at node** $i \in \tau'$ if either $b_i = b'_i$ or $\vec{cv}_i \neq \vec{cv}'_i$. The pair is **as promised** if it satisfies the relation R, or, equivalently, if the promise is kept at all nodes i.

Let us fix a protocol supposedly solving the problem with fewer than $2k - 3\sqrt{k} - \log(6k^3)$ bits of communication. Let $l = \sqrt{k}$ and $t = k - l - \log(2k^2)$. (We assume that l and t are integers, otherwise we round l up and t down.)

For C a partial conversation, let S_C^A be the set of $\langle L_1, L_2 \rangle$ labelings that can be given to the A-Player and be consistent with the conversation C so far. Define S_C^B similarly.

5.2. The First Stage Conversation. During the first stage, the adversary will pick C to maximize the symmetry between the players. More precisely, the adversary will find a conversation C and a set $S \subseteq S_C^A \cap S_C^B$ that is "full" in the following sense. Let H^S be the hierarchy tree for S with nodes $S^{\geq 3} \cup S^{\geq 2} \cup S^{\geq 1}$. See Figure 5.2. The set $S^{\geq 3}$ consists of a single root Λ. Its children are $S^{\geq 2} =$

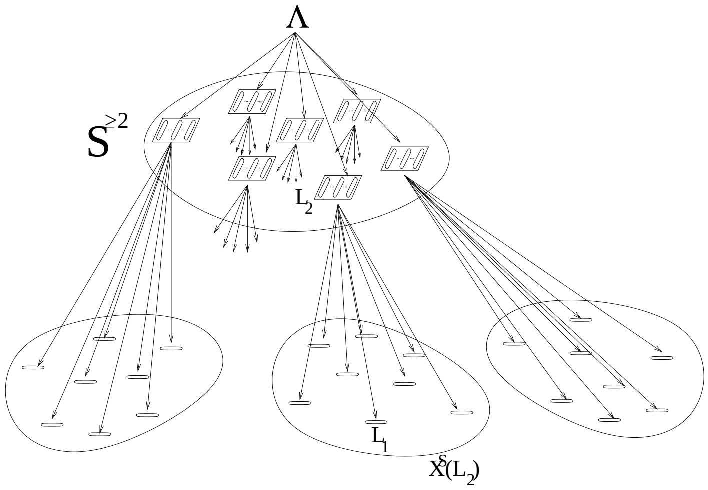

Figure 5.2: The hierarchy tree H^S for S

$X^S(\Lambda) = Proj(S, \tau_1)$ is the set of possible L_2 which might arise from a labeling in S. For a given $L_2 \in S^{\geq 2}$, $X^S(L_2)$ is the set of all L_1 with $\langle L_1, L_2 \rangle \in S$. After almost k bits have been communicated, the adversary wants many possible L_2's that could be given to either player, so that it will be hard to find a difference in the leaves. There may still be a few vectors in L_2 that have been the subject of significant communication, so the adversary will want to make the corresponding bits of L_1 identical, while allowing the players to be given distinct L_1's to keep the promise at the root. To do this, the adversary also wants as many as possible L_1's for each possible L_2.

Lemma 5.1. *Fix any protocol for $UCR_{2,k}$. Then there exists a $C \in \{0,1\}^t$ and an $S \subseteq S_C^A \cap S_C^B$ so that $|S^{\geq 2}| \geq 2^{k^2-t}$; and, for each $L_2 \in S^{\geq 2}$, $|X^S(L_2)| \geq 2^{k-t} = 2k^2(2^l)$.*

Proof. Partition the set $W^{\tau'}$ of labelings $\langle L_1, L_2\rangle$ according to the first t bits of communication that would result if both players are given this labeling. (This choice of inputs is not as promised, but the protocol is still defined.) If $\langle L_1, L_2\rangle$ is in S_C, the class of the partition corresponding to partial conversation C, then $\langle L_1, L_2\rangle \in S_C^A \cap S_C^B$. Obtain the conversation C and subset $S \subseteq S_C$ by Lemma 4.10; the guarantee on size of extension sets is exactly as above, since the number of classes in the partition is at most $p = 2^t$, $|S^{\geq 2}| = |X^S(\Lambda)| \geq |X^{W^{\tau'}}(\Lambda)|/p = 2^{|\tau_3|}/p = 2^{k^2-t}$, and for every $L_2 \in S^{\geq 2}$, $|X^S(L_2)| \geq |X^{W^{\tau'}}(L_2)|/p = 2^{|\tau_2|}/p = 2^{k-t}$. □

5.3. Breaking Symmetry for L_1. Between the stages, the adversary does some "cleaning up". At this point, enough information has been communicated about L_1 for the players to start finding a difference. On the other hand, intuitively, only a few child vectors in L_2 have had significant amounts of information revealed about them. The adversary wants to identify the child vectors of L_2 for which more than k/l bits have been communicated, fix these as functions of the remaining child vectors, and ensure the corresponding bits of L_1 are fixed to be the same for both players. Then the adversary can break the symmetry for L_1, by partitioning possible values for L_1 between the two players, ensuring that the players are given distinct values as promised. This process is described formally as the proof of the following lemma:

Lemma 5.2. *Fix any protocol for $UCR_{2,k}$. Then there are:*

(i) a partial conversation $C \in \{0,1\}^t$;

(ii) a set of nodes $\sigma \subseteq \tau_2$, $|\sigma| \leq l$;

(iii) a set $T \subseteq (\{0,1\}^k)^{\tau_2-\sigma}$;

(iv) an extension function $E_1 : T \to (\{0,1\}^k)^{\tau_2}$

(v) extension functions $E^A, E^B : E_1(T) \to \{0,1\}^\tau$

satisfying

(i) For all $L_2 \in E_1(T)$, $E^A(L_2) \in S_C^A$ and $E^B(L_2) \in S_C^B$.

(ii) For all $L_2, L_2' \in E_1(T)$, the promise is kept for the pair of inputs $E^A(L_2)$ and $E^B(L_2')$ at all $i \in \sigma \cup \tau_1$.

(iii) For all $i \in \tau_2 - \sigma$, $Pred_i(T) \leq 2^{k/l+l} \cdot 2^{-k}$;

PROOF. Let C and S be as guaranteed by Lemma 5.1.

First, we identify the child vectors in L_2 where "a lot of information has been communicated". By Lemma 4.6 since $|S^{\geq 2}| \geq 2^{k^2-t} > 2^{k^2-k}$ (i.e., no more than k bits have been communicated about L_2), there exists a subset $\sigma \subseteq \tau_2$, $|\sigma| \leq l$, such that if these are ignored, then the remaining child vectors $i \in \tau_2 - \sigma$ have predictability $Pred_i(Proj(S^{\geq 2}, \sigma)) \leq 2^{k/l} \cdot 2^{-k}$ (i.e., no more than k/l bits have been communicated "about each of these child vectors"). For each element M of $Proj(S^{\geq 2}, \sigma)$, choose a single extension $E_1(M) \in S^{\geq 2}$; let $H' \subseteq S^{\geq 2}$ be the set of all values of $E_1(M)$. Note that E_1 and $Proj(L_2, \sigma)$ are one-to-one correspondences between H' and $Proj(S^{\geq 2}, \sigma)$. Let $S' = \{\langle L_1, L_2 \rangle \in S \mid L_2 \in H'\}$. Note that $(S')^{\geq 2} = H'$ and $X^{S'}(L_2) = X^S(L_2)$ for every $L_2 \in H'$.

As far as we know, the child vectors of nodes in σ might now be constant in S'. In order to have the promise kept for these nodes, the adversary now fixes the bits of L_1 labeling the nodes in σ. To choose values of bits indexed by σ, we again use Lemma 4.10, similarly as if $|\sigma|$ bits are communicated. More precisely, we partition S' into $p \leq 2^l$ sets according to the bits $b_i, i \in \sigma$; then by Lemma 4.10, there are $\bar{b}_i$, $i \in \sigma$ and $S'' \subseteq S'$ such that:

(i) For all labelings in S'', and all $i \in \sigma$, $b_i = \bar{b}_i$.

(ii) $|(S'')^{\geq 2}| \geq |(S')^{\geq 2}|/2^l = |Proj(S^{\geq 2}, \sigma)|/2^l$

(iii) For all $L_2 \in (S'')^{\geq 2}$, $|X^{S''}(L_2)| \geq |X^{S'}(L_2)|/2^l = |X^S(L_2)|/2^l \geq 2k^2$

Let $T = Proj((S'')^{\geq 2}, \sigma)$. Then since $(S'')^{\geq 2} \subseteq H'$ and the function $Proj(L_2, \sigma)$ is one-to-one on H', $|T| = |(S'')^{\geq 2}| \geq |H'|/2^l$. Therefore, by Lemma 4.5, the predictability of any child vector can go up by at most a factor of 2^l over its predictability in $Proj(H', \sigma)$. Therefore, for all $i \in \tau_2 - \sigma$, $Pred_i(T) \leq 2^l Pred_i(Proj(S^{\geq 2}, \sigma)) \leq 2^{k/l+l} \cdot 2^{-k}$ (i.e., no more than $k/l+l$ bits have been communicated "about each of these child vectors"), so condition (iii) of the lemma is proved.

For the other conditions, we need to define the extension functions E^A and E^B. We start by using E_1 to go from a labeling of $\tau_2 - \sigma$ to an entire L_2. Now we need to find L_1's. The tricky part is to ensure the promise at the root is kept, i.e. that the L_1 vectors are different. Thus, the adversary must now break the symmetry between the two players' possible values for L_1. For each $L_2 \in T$, one L_1 vector is chosen for the A-Player from the remaining set $X^S(L_2)$ and one is chosen for the B-Player. In this way, the L_2 chosen for each player at the end of the protocol will fix the L_1 vectors for the two players in an asymmetric way as a function of the L_2. In order to ensure that the players get different L_1 vectors special care needs to be taken. The adversary chooses a function $G : \{0,1\}^k \to \{0,1\}$ randomly. The A-Player will be given an L_1 from $G^{-1}(0)$ and the B-Player from $G^{-1}(1)$. We must ensure that for each $L_2 \in E_1(T)$, $X^{S''}(L_2) \cap G^{-1}(0)$ and $X^{S''}(L_2) \cap G^{-1}(1)$ contain at least one vector each. To see that this is the case with high probability, note that for each such L_2, $L_2 \in (S'')^{\geq 2}$, thus $|X^{S''}(L_2)| \geq 2k^2$ and for a random function G, the probability that all elements of this set are 1's of the function, $Prob_G[X^{S''}(L_2) \cap G^{-1}(0) = \emptyset]$, is at most 2^{-2k^2}, and likewise for all 0's. Therefore, since there are at most 2^{k^2} such M, the probability that there is such an M is small; namely

$$Prob_G[(\exists L_2 \in E_1(T))(X^{S''}(L_2) \cap G^{-1}(0) = \emptyset \vee X^{S''}(L_2) \cap G^{-1}(1) = \emptyset)] \leq 2|T| \cdot 2^{-2k^2} \leq 2^{1-k^2} < 1.$$

So we pick a function G where this never occurs, and for each $L_2 \in E_1(T)$, we select $L_1^A \in X^{S''}(L_2) \cup G^{-1}(0)$ and $L_1^B \in X^{S''}(L_2) \cup G^{-1}(1)$ and set $E^A(L_2) = \langle L_1^A, L_2\rangle$ and $E^B(L_2) = \langle L_1^B, L_2\rangle$.

Then for any $L_2, L_2' \in E_1(T)$, let $L = E^A(L_2) \in S''$ and $L' = E^B(L_2') \in S''$. Then since $G(L_1) = 0$ and $G(L_1') = 1$, $L_1 \neq L_1'$, and so the promise is kept at the root for the pair L and L'. For $i \in \sigma$, since $L, L' \in S''$, $b_i = \bar{b}_i = b_i'$, so the promise is kept at each $i \in \sigma$. So condition (ii) of the lemma holds.

Since $S'' \subseteq S \subseteq S_C^A \cap S_C^B$, $L \in S_C^A$ and $L' \in S_C^B$. Thus condition (i) of the lemma holds. □

In the general case, $d > 2$, we will need to use Lemma 4.7 to handle this step, because we will need to break the symmetry simultaneously for many vectors.

5.4. The Second Stage. During the second stage of the lower bound the players continue to communicate bits. This stage continues for $t' = k - k/l - l - \log(3k)$ bits of communication. Since, for each player, we have fixed L_1 as the functions E^A, E^B of L_2, and we have fixed the rest of L_2 as the function E_1 of $Proj(L_2, \sigma)$, bits communicated partition the set T of possible $Proj(L_2, \sigma)$'s. More precisely, look at the conversation produced on inputs $E^A(M)$ and $E^B(M)$ for each $M \in T$ for $t + t'$ bits. (As before, such a pair need not be as promised for the protocol to be defined.) The first t bits are always C. So this partitions the set T into at most $2^{t'}$ subsets, at least one of which $T' \subseteq T$, is of size at least $|T|/2^{t'}$. Now by Lemma 4.5 the predictability of any child vector has gone up by at most the same factor. Since for T we had $\forall i \in \tau_2 - \sigma$, $Pred_i(T) \leq 2^{k/l+l} \cdot 2^{-k}$, we now have $\forall i \in \tau_2 - \sigma$, $Pred_i(T') \leq 2^{k/l+l} \cdot 2^{t'} \cdot 2^{-k} = 2^{-\log(3k)} = 1/(3k)$.

Summarizing:

Lemma 5.3. *Fix any protocol for $UCR_{d,k}$. Let $t'' = t + t' = 2k - k/l - 2l - \log(6k^3)$. Then there are:*

(i) a conversation $C' \in \{0,1\}^{t''}$;

(ii) a set of nodes $\sigma \subseteq \tau_2$, $|\sigma| \leq l$;

(iii) a set $T' \subseteq (\{0,1\}^k)^{\tau_2 - \sigma}$;

(iv) an extension function $E_1 : T \rightarrow (\{0,1\}^k)^{\tau_2}$

(v) extension functions $E^A, E^B : E_1(T) \rightarrow \{0,1\}^{\tau}$

satisfying

(i) For all $L_2 \in E_1(T')$, $E^A(L_2) \in S^A_{C'}$ and $E^B(L_2) \in S^B_{C'}$.

(ii) For all $L_2, L_2' \in E_1(T')$, the promise is kept for the pair of inputs $E^A(L_2)$ and $E^B(L_2')$ at all $i \in \sigma \cup \tau_1$.

(iii) For all $i \in \tau_2 - \sigma$, $Pred_i(T') \leq 1/(3k)$

5.5. Choosing The Inputs. The protocol ends at this point and outputs $Out(C') \in \tau_3$, claiming that $b_{Out(C')} \neq b'_{Out(C')}$. The adversary must find a pair $\langle L_1^A, L_2^A \rangle$ and a pair $\langle L_1^B, L_2^B \rangle$ which are consistent with C', which are as promised at all nodes, and for which the specified leaf $Out(C')$ is labeled the same.

The adversary chooses these inputs by choosing two vectors L_2^A, L_2^B independently and uniformly at random from the set $E_1(T')$ (since E_1 is one-to-one, this is equivalent to sampling from T'). This fixes the inputs $\langle L_1^A, L_2^A \rangle = E^A(L_2^A)$ and $\langle L_1^B, L_2^B \rangle = E^B(L_2^B)$.

The claim is that with probability at least $1/6$, these inputs meet the adversary's requirements.

First, the pair is consistent with C' by Lemma 5.3 (i).

We now compute a lower bound on the probability that the pair of inputs is as promised. By Lemma 5.3 (ii) the promise is kept for all nodes in $\tau_1 \cup \sigma$.

So we just need to look at the probability they keep the promise at nodes $i \in \tau_2 - \sigma$. For the promise to be kept at such nodes, it suffices that, for each $i \in \tau_2 - \sigma$, $\vec{cv}_i \neq \vec{cv}'_i$. Now, $\vec{cv}_i$ and $\vec{cv}'_i$ are parts of two uniform and independent samples from T'. Lemma 4.4 states that the probability, for two samples randomly chosen (with replacement) from T', that the i^{th} component is the same, is no more than $Pred_i(T')$, which is bounded by $Pred_i(T') \leq 1/(3k)$ using Lemma 5.3 (iii). Thus, the probability that at least one of the k such child vectors is the same is no more than $1/3$ and the input pair is as promised with probability at least $2/3$

Now let us calculate the probability that the specified index of τ_2 is labeled the same. Let p be the probability that for a random $L_2 \in E_1(T')$, $b_{Out(C')} = 1$. Because both players' L_2's are identically and independently chosen from $E_1(T')$, this probability is the same for both players and the probability that the label is different is $2p(1-p) \leq 1/2$.

Thus, the probability that the pair is as promised and the bits $b_{Out(C')}$ are the same is at least $2/3 - 1/2 = 1/6$. Therefore, there exists a choice of inputs which causes the protocol to fail. Since this holds for an arbitrary protocol, we get

THEOREM 5.4. $CC(UCR_{2,k}) \geq 2k - 3\sqrt{k} - \log(6k^3)$.

6. Proof for $d \geq 3$

6.1. Overview. Although the proof is considerably more complicated, the intuition behind the case $d \geq 3$ is the same as that for $d = 2$. In both, the adversary tries to delay progress on the part of the protocol as long as possible, where progress means finding a difference in the players' labeling deep in the tree. If the players know a difference at some node at level i they can find a difference at level $i+1$ within k bits of communication. Thus approximately every k bits we begin a new "stage" of the lower bound. During the stage r, the adversary assumes that the players have already found a difference in the level $r-1$ and tries to prevent them from finding a difference in the level r.

The way this is modeled in the proof is for the adversary to find a conversation for the first r stages and a set of possible inputs for each player where each pair of possible inputs leads to this conversation. Furthermore, the way we model "the players know no difference beyond depth r in the tree", is to make these sets identical as far as their possible $L_{r+1}, \ldots, L_d$ parts. This means that we cannot guarantee to fulfill the promises in these parts, since we cannot guarantee any differences at these levels. However, we make sure that this set is large, so that two random elements of the set are likely to have enough differences to fulfill the promises. By fixing $L_1, \ldots, L_r$ as functions of $L_{r+1}, \ldots, L_d$ in different ways for the two players, we ensure the promises are kept for these levels.

More precisely, at the end of stage r, the adversary has fixed a conversation C_r and a small set $\sigma_r \subseteq \tau_{r+1}$ which are the child vectors in L_{r+1} about which "significant communication" has occurred. The adversary maintains a set of labelings T_r of $\tau_{\geq r+1} - \sigma_r$ which are possible for both players. This specifies possible labelings $L_{r+2}, \ldots, L_d$ symmetrically for the players, as well as the "unpredictable" parts of L_{r+1}. Then the labeling of σ_r , and hence L_{r+1} is determined by an extension function E_r symmetrically for both players, which are then asymmetrically extended to complete labelings by functions E_r^A and E_r^B. Intuitively, since the sets of inputs for the two players are the same as far as their labelings of $\tau_{\geq r+1}$ go, the players do not yet know a difference at the $(r+1)^{st}$ level.

The delicate part is when the players have revealed enough information to almost find a difference at the r^{th} level. In order not to violate a promise, we need to break symmetry at the r^{th} level, but maintain it at the $(r+1)^{st}$. The steps towards doing this are:

1. Identify the set of nodes σ_{r+1} at level $r+1$ where "significant information has been communicated".

2. Fix the child vectors of these nodes as functions of the rest of L_{r+1}.

3. Fix the bits labeling these nodes.

4. For each $i \in \tau_{r+1} - \sigma_r$, partition the elements of $\{0,1\}^k$ between the players to make sure that these child vectors differ.

6.2. Notation. Let $l = \sqrt{k/\log k}$, $a_1 = dk/l$, $a_2 = dl \log k$, $a_3 = 4d \log k + 4$, and $s = k - a_1 - a_2 - a_3 - 2l = k - O(d(k \log k)^{1/2})$. (Note that $a_1 = a_2$, to optimize the bound.) Fix any protocol which supposedly solves $UCR_{d,k}$ using at most sd bits of communication. We call the time from the bit $(r-1)s+1$ to bit rs of the conversation stage r of the protocol.

For j, $1 \leq j \leq d+1$, let τ_j represent the nodes of the complete, depth $d+1$, k-ary tree at depth j, so τ_1 is a single root, τ_2 its k children, etc. Remember, in the **child vector representation**, L_i is an array of elements of $\{0,1\}^k$ indexed by τ_i; in the **bit representation** it is an array of bits indexed by τ_{i+1}. For notational convenience, we will think of a labeling as having both representations. In particular, we will call a function an extension function even if its input is in one format and output in the other, as long as the input and output are identical where they are both defined (if, say, both were converted to bit representation). For a labeling L, we use the notation b_i, $i \in \tau_{\geq 2}$, to represent the bit labeled by i in the bit representation of L and $\vec{cv}_i, i \in \tau_{\leq d}$, to represent the k bit string indexed by i in its child vector representation.

In the proof, after stage r we will have a set of partial labelings T_r, each member is a possible labeling $L_{r+2}, \ldots, L_d$, plus the part of L_{r+1} which is indexed by $\tau_{r+1} - \sigma_r$ (for a set σ_r representing those child vectors in L_{r+1} about which "too much information has been revealed"). We will use Lemma 4.10 on this set, with respect to the partition $\tau_{r+1} - \sigma_r$, τ_{r+2}, $\ldots$, τ_d, so all references to the hierarchy tree for T_r are with respect to this partition.

Let L^A and L^B be labelings of $\tau_{\geq 2}$, with bit representations b_i^A and b_i^B and child vector representations $\vec{cv}_i^A$ and $\vec{cv}_i^B$. We say the pair is **as promised for node** $i \in \tau_{\geq 2}$ if either $b_i^A = b_i^B$ or $\vec{cv}_i^A \neq \vec{cv}_i^B$. The pair is **as promised for the root** $i \in \tau_1$ if $\vec{cv}_i^A \neq \vec{cv}_i^B$.

6.3. Induction Assumption. The inductive statement is analogous to the case of $d = 2$, Lemma 5.2. We will prove, by induction on $r = 0, \ldots, d-1$, that:

LEMMA 6.1. *Let $0 \leq r \leq d-1$. Then there are:*

(i) a partial conversation $C_r \in \{0,1\}^{sr}$;

(ii) a set of nodes $\sigma_r \subseteq \tau_{r+1}$, $|\sigma_r| \leq l$;

(iii) a set $T_r \subseteq (\{0,1\}^k)^{\tau_{\geq r+1} - \sigma_r}$;

(iv) an extension function $E_r : T_r \rightarrow (\{0,1\}^k)^{\tau_{\geq r+1}}$

(v) extension functions $E_r^A, E_r^B : E_r(T_r) \rightarrow \{0,1\}^\tau$

satisfying

(IND-i) *For all $L_{\geq r+1} \in E_r(T_r)$, $E_r^A(L_{\geq r+1}) \in S_C^A$, and $E_r^B(L_{\geq r+1}) \in S_C^B$,*

(IND-ii) *For all $L_{\geq r+1}^A, L_{\geq r+1}^B \in E_r(T_r)$, the promise is kept for the two inputs $E_r^A(L_{\geq r+1}^A), E_r^B(L_{\geq r+1}^B)$ at all $i \in \sigma_r \cup \tau_{\leq r}$.*

(IND-iii) *For all $L_{\geq r+2} \in T_r^{\geq r+2}$, and for all $i \in \tau_{r+1} - \sigma_r$,*

$$Pred_i(X^{T_r}(L_{\geq r+2})) \leq 2^{-(s+a_3+l)} = 2^{a_1+a_2+l} \cdot 2^{-k};$$

(IND-iv) *For all $d+1 \geq j \geq r+3$ and for all $L_{\geq j} \in T_r^{\geq j}$, $|X^{T_r}(L_{\geq j})| \geq 2^{-kr} \cdot 2^{|\tau_j|}$.*

Intuitively, this lemma says that the adversary can maintain a symmetrical set of $L_{\geq r+1}$'s, $E_r(T_r)$, which can be inputs to either player. Any two inputs from this symmetrical set can be given to the two players without breaking promises in non-symmetrical parts, by suitable and asymmetrical extensions to a complete input. Any part of L_{r+1} that is predictable given the conversation so far has had the corresponding bit in L_r fixed to ensure the promise is kept there. So the players still need to work hard to find any difference in L_{r+1}. Finally, the number of possible labelings $L_{r+2}, \ldots, L_d$ in the set is large enough that, even given $L_j, \ldots, L_d$, the players know no more than kr bits about L_{j-1}, for any $j \geq r+3$.

6.4. The Base Case. For $r = 0$, we can pick $\sigma_0 = \emptyset$, and $T_0 = (\{0,1\}^k)^{\tau_{\geq 1}}$, the set of all labelings of all nodes except the root of the tree; $C_0 = \varepsilon$; E_0, E_0^A, and E_0^B are all the identity function. All conditions above are vacuously satisfied (note that in condition (IND-iii), we have a full set of labelings L_1 in the extension set, so the predictability is 2^{-k}, much less than required).

6.5. Induction Step. Assume we have $C_{r-1}, T_{r-1}, \sigma_{r-1}, E_{r-1}, E_{r-1}^A, E_{r-1}^B$ as in the induction claim. We want to find $C_r, T_r, \sigma_r, E_r^A, E_r^B$ that satisfy the claim at r. Remember, $E_{r-1}(T_{r-1})$ is a set of tuples $\langle L_r, \ldots, L_d \rangle$ that could be given to either player and be consistent with the conversation C_{r-1} when extended by E_{r-1}^A or E_{r-1}^B to get $L_0, \ldots, L_{r-1}$. These extensions are guaranteed to keep promises in the lower levels and at those parts of L_r that are indexed by σ_{r-1}. The other parts of L_r are unpredictable, and not much total information has been revealed about $L_{r+1}, \ldots, L_d$. So, while the players might have found a difference in level L_{r-1}, intuitively they have not found a difference at level L_r or above. In the induction step, we need to argue that, after s more bits of communication, while they might be able to find a difference in L_r, we can keep $L_{r+1}, \ldots, L_d$ symmetrical for the two players.

6.5.1. Picking the Conversation for Stage r. First, we must pick a conversation for stage r that does not give away too much information about any level of the labeling.

Partition T_{r-1} by associating each $L_{\geq r} \in T_{r-1}$ with the bits communicated on the pair of inputs $E_{r-1}^A(E_{r-1}(L_{\geq r}))$ for the A player and $E_{r-1}^B(E_{r-1}(L_{\geq r}))$ for the B player. (This pair is not as promised, but the protocol is still defined on it.) The first $(r-1)s$ bits communicated will be C_{r-1}. The s bits communicated in stage r thus create a partition of T_{r-1} into at most 2^s pieces.

The adversary, by Lemma 4.10, can find a subset $T' \subseteq T_{r-1}$ and a conversation C_r extending C_{r-1} so that all members of T' yield conversation C_r, and so that the following holds: for all j, $r+1 \le j \le d+1$, and for all $L_{\ge j} \in (T')^{\ge j}$, $|X^{T'}(L_{\ge j})| \ge 2^{-s} \cdot |X^{T_{r-1}}(L_{\ge j})|$.

In particular, for $j = r+1$ and any $i \in \tau_r - \sigma_{r-1}$, induction assumption (IND-iii) and Lemma 4.5 yield $Pred_i(X^{T'}(L_{\ge r+1})) \le 2^s \cdot Pred_i(X^{T_{r-1}}(L_{\ge r+1})) \le 2^s \cdot 2^{-(s+a_3+l)} = 2^{-(a_3+l)}$ (i.e., even given $L_{r+1}, \ldots, L_d$, any place in L_r not indexed by σ_{r-1} is still pretty unpredictable). For $j \ge r+2$, by induction assumption (IND-iv) we have $|X^{T'}(L_{\ge j})| \ge 2^{-s} \cdot |X^{T_{r-1}}(L_{\ge j})| \ge 2^{-(k(r-1)+s)} \cdot 2^{|\tau_j|}$. So at most s more bits have been revealed about any L_j, $r+1 \le j \le d$.

6.5.2. Finding Predictable Places in L_{r+1}. Intuitively, we want to argue that the adversary can arrange for the promises to be kept at level L_r without revealing any differences in L_{r+1}. The problem is that there are possibly a few places in L_{r+1} where significant communication has occurred. If we make those places different in L_r, the players will be on their way to finding a difference in L_{r+1}. So we must find those places, and argue that we have enough slack in L_r to make those places identical while still allowing promises to be kept.

For each $L_{\ge r+2} \in (T')^{\ge r+2}$, we know that $|X^{T'}(L_{\ge r+2})| \ge 2^{-(k(r-1)+s)} \cdot 2^{|\tau_{r+2}|} \ge 2^{-kd} \cdot (2^k)^{|\tau_{r+1}|}$. By Lemma 4.6, for each such $L_{\ge r+2}$ there is a subset $\sigma' \subseteq \tau_{r+1}$ of the indexes for L_{r+1}, $|\sigma'| \le l$, such that if we remove these child vectors, the other child vectors $i \in \tau_{r+1} - \sigma'$ are unpredictable, i.e., $Pred_i(Proj(X^{T'}(L_{\ge r+2}), \sigma')) \le 2^{kd/l} \cdot 2^{-k} = 2^{a_1} \cdot 2^{-k}$.

These predictable child vectors represent the places where the players might find a difference in L_{r+1} quickly, if they knew the corresponding bits in L_r were different. So we would like to fix the bits corresponding to σ' in L_r to be constant and identical for both players. Unfortunately, the set σ' depends on $L_{r+2}, \ldots, L_d$. Fortunately there are not too many possible sets.

We associate each $L_{\ge r+2} \in (T')^{\ge r+2}$ with the set σ' described above. This partitions $(T')^{\ge r+2}$ into at most $|\tau_{r+1}|^l \le k^{dl} = 2^{a_2}$ classes, each corresponding to one possible value of σ'. Now we use Lemma 4.10 (with $r+2$ in place of r) to choose one set $\sigma' \subseteq \tau_{r+1}$, $|\sigma'| \le l$, and a corresponding partition class $T'_{\sigma'} \subseteq (T')^{\ge r+2}$. Define $\sigma_r = \sigma'$ and T'' to be the set of all $L_{\ge r} \in T'$ such that $Proj(L_{\ge r}, \tau_{\le r+1}) \in T'_{\sigma'}$, i.e., the set of extensions of all elements of $T'_{\sigma'}$ by all consistent L_{r+1} and L_r. The following conditions are now satisfied:

- for all j, $r+3 \le j \le d+1$, and for all $L_{\ge j} \in (T'')^{\ge j}$, $|X^{T''}(L_{\ge j})| \ge 2^{-a_2} \cdot |X^{T'}(L_{\ge j})| \ge 2^{-((r-1)k+s+a_2)} \cdot 2^{|\tau_j|}$.

- For all $L_{\geq r+2} \in (T'')^{\geq r+2}$, and all $i \in \tau_{r+1} - \sigma_r$,

$$Pred_i(Proj(X^{T''}(L_{\geq r+2}), \sigma_r)) \leq 2^{a_1} \cdot 2^{-k}.$$

- For any $L_{\geq r+1} \in (T'')^{\geq r+1}$ and any $i \in \tau_r - \sigma_{r-1}$, we have

$$Pred_i(X^{T''}(L_{\geq r+1})) = Pred_i(X^{T'}(L_{\geq r+1})) \leq 2^{-(a_3+l)}.$$

The first two conditions are implied by Lemma 4.10 and our choice of $\sigma_r = \sigma'$, the last condition follows from Section 6.5.1.

To ensure that promises are kept for $i \in \sigma_r$, we fix the corresponding bits in L_r as follows: Partition T'' into at most 2^l sets according to the values of b_i, $i \in \sigma_r$, in $E_{r-1}(L)$. Applying Lemma 4.10, we get bits $\bar{b}_i$, $i \in \sigma_r$, and a set $T''' \subseteq T''$ with $|X^{T'''}(L_{\geq j})| \geq 2^{-l} \cdot |X^{T''}(L_{\geq j})|$, for every $L_{\geq j} \in T^{\geq j}$, and such that for every $L^{\geq r} \in E_{r-1}(T''')$, and every $i \in \sigma_r$, $b_i = \bar{b}_i$. Then we have (applying the similar conditions for T'' and Lemma 4.5):

Lemma 6.2.

(i) *For all $L_{\geq r} \in E_{r-1}(T''')$, $E^A_{r-1}(L_{\geq r}) \in S^A_{C_r}$ and $E^B_{r-1}(L_{\geq r}) \in S^B_{C_r}$.*

(ii) *For all $L^A_{\geq r}, L^B_{\geq r} \in E_{r-1}(T''')$, for $E^A_{r-1}(L^A_{\geq r})$ and $E^B_{r-1}(L^B_{\geq r})$ the promise is kept at all $i \in \tau_{\leq r-1} \cup \sigma_{r-1} \cup \sigma_r$.*

(iii) *For any $L_{\geq r+1} \in (T''')^{\geq r+1}$, and $i \in \tau_r - \sigma_{r-1}$, $Pred_i(X^{T'''}(L_{\geq r+1})) \leq 2^{-a_3}$.*

(iv) *For all $L_{\geq r+2} \in (T''')^{\geq r+2}$, and for all $i \in \tau_{r+1} - \sigma_r$,*

$$Pred_i(Proj(X^{T'''}(L_{\geq r+2}), \sigma_r)) \leq 2^{a_1+l} \cdot 2^{-k}.$$

(v) *For all j, $r+3 \leq j \leq d+1$, and for all $L_{\geq j} \in (T''')^{\geq j}$, $|X^{T'''}(L_{\geq j})| \geq 2^{-((r-1)k+s+a_2+l)} \cdot 2^{|\tau_j|} \geq 2^{-rk} \cdot 2^{|\tau_j|}$.*

6.5.3. Breaking Symmetry for L_r. At this point, the vectors in L_r corresponding to $\tau_r - \sigma_{r-1}$ not sufficiently unpredictable, so that the players can agree on a difference there with a few bits of communication. Moreover, such a difference has to exists, since the corresponding bits in L_{r-1} have been fixed as a function of L_r in an asymmetrical way, so if no such difference existed, the promise would be violated. So to ensure promises are kept, in the following construction, the adversary breaks the symmetry in a very strong sense, using the remaining unpredictability of the vectors L_r.

Randomly choose for each index $i \in \tau_r - \sigma_{r-1}$ a function g_i from child vectors $\{0,1\}^k$ to $\{0,1\}$. The adversary restricts inputs L_r for player A to those where $g_i(\vec{cv}_i) = 0$ for all such i and for B to those where $g_i(\vec{cv}_i) = 1$ for all such i. Call such an input L_r a **promise keeper** for the appropriate player. If L_r^A is a promise keeper for A and L_r^B for B, then L^A and L^B keep the promise at all $i \in \tau_r - \sigma_{r-1}$, since we must have $\vec{cv}_i^A \neq \vec{cv}_i^B$.

Let $L_{\geq r+1} \in (T''')^{\geq r+1}$. By Lemma 6.2 (iii), for all $i \in \tau_r - \sigma_{r-1}$,

$$Pred_i(X^{T'''}(L_{\geq r+1})) \leq 2^{-a_3} = \frac{1}{16k^{4d}} \leq \frac{1}{4|\tau_r|(|\tau| + \log|\tau_r| + 2)}$$

By Lemma 4.7, setting $m = |\tau_{\geq r}| + \log|\tau_r| + 2$, the probability that there is no promise keeper for A in $X^{T'''}(L_{\geq r+1})$, is at most $|\tau_r|/2^m = 1/(4 \cdot 2^{|\tau|}) \leq 1/(4|(T''')^{\geq r+1}|)$, and similarly for B. Therefore, for some choice of $\vec{g}$, there are such promise keepers $PK^A(L_{\geq r+1}), PK^B(L_{\geq r+1})$ for all $L_{\geq r+1} \in (T''')^{\geq r+1}$. We define $E_r^A(L_{\geq r+1}) = E_{r-1}^A(E_{r-1}((PK^A(L_{\geq r+1})))$ and similarly $E_r^B(L_{\geq r+1}) = E_{r-1}^B(E_{r-1}((PK^B(L_{\geq r+1})))$.

Let $T_r = Proj((T''')^{\geq r+1}, \sigma_r)$. For each $M \in T_r$, arbitrarily select an extension $E_r(M) \in (T''')^{\geq r+1}$. Note that $(T_r)^{\geq j} = (T''')^{\geq j}$ for all j, $r+2 \leq j \leq d$ and that $X^{T_r}(L_{\geq r+2}) = Proj(X^{T'''}(L_{\geq r+2}), \sigma_r)$ for all $L_{\geq r+2} \in (T_r)^{\geq r+2}$.

6.5.4. Verifying the Induction Hypothesis for r. We now have defined $C_r, \sigma_r, T_r, E_r, E_r^A$ and E_r^B. We need to verify that the induction hypothesis holds for these definitions:

(IND-i) By Lemma 6.2 (i), for all $L_{\geq r} \in E_{r-1}(T''')$, we have $E_{r-1}^A(L_{\geq r}) \in S_{C_r}^A$ and $E_{r-1}^B(L_{\geq r}) \in S_{C_r}^B$. Therefore, for $L_{\geq r+1} \in E_r(T_r)$ we let $L_{\geq r}^A = E_{r-1}(PK^A(L_{\geq r+1})) \in E_{r-1}(T''')$ and $L_{\geq r}^B = E_{r-1}(PK^B(L_{\geq r+1})) \in E_{r-1}(T''')$, and obtain $E_r^A(L_{\geq r+1}) = E_{r-1}^A(L_{\geq r}^A) \in S_{C_r}^A$ and $E_r^B(L_{\geq r+1}) = E_{r-1}^B(L_{\geq r}^B) \in S_{C_r}^B$.

(IND-ii) Defining $L_{\geq r}^A \in T'''$ and $L_{\geq r}^B \in T'''$ as above, by Lemma 6.2 (ii), the promise is kept at all $i \in \tau_{r-1} \cup \sigma_{r-1} \cup \sigma_r$. It is also kept at $i \in \tau_r - \sigma_{r-1}$ because $g_i(\vec{cv}_i^A) = 0$ and $g_i(\vec{cv}_i^B) = 1$. Thus for all $L_{\geq r+1}^A, L_{\geq r+1}^B \in E_r(T_r)$, the promise is kept for the two inputs $E_r^A(L_{\geq r+1}^A), E_r^B(L_{\geq r+1}^B)$ at all $i \in \sigma_r \cup \tau_{\leq r}$.

(IND-iii) By Lemma 6.2 (iv), for all $L_{\geq r+2} \in T_r^{\geq r+2}$ and all $i \in \tau_{r+1} - \sigma_r$. This follows since

$$Pred_i(X^{T_r}(L_{\geq r+2})) = Pred_i(Proj(X^{T'''}(L_{\geq r+2}), \sigma_r)) \leq 2^{a_1+l} \cdot 2^{-k}.$$

(IND-iv) By Lemma 6.2 (v), for all j, $r+3 \le j \le d+1$ and all $L_{\ge j} \in T_r^{\ge j}$,

$$|X^{T_r}(L_{\ge j})| = X^{T'''}(L_{\ge j}) \ge 2^{-kr} 2^{|\tau_j|}.$$

This finishes the proof of Lemma 6.1.

6.6. The Last Stage Conversation. We have proved the inductive assumption of Lemma 6.1 for r up to $d-1$. We need to deal with the last stage separately, as some of the induction hypothesis become degenerate when $r = d-1$. In this case, note that $T_{d-1} \subseteq (\{0,1\}^k)^{\tau_d - \sigma_{d-1}}$ and $E_{d-1}(T_{d-1}) \subseteq (\{0,1\}^k)^{\tau_d}$ is a symmetric set of possible labelings L_d of the last level that can be extended to both players. So we are in the analogous position to Section 5.4.

Applying Lemma 6.1 when $r = d-1$ gives us C_{d-1}, σ_{d-1}, T_{d-1}, E_{d-1}, E^A_{d-1}, and E^B_{d-1} so that:

(i) For all $L_d \in E_{d-1}(T_{d-1})$, $E^A_{d-1}(L_d) \in S^A_{C_{d-1}}$, and $E^B_{d-1}(L_d) \in S^B_{C_{d-1}}$.

(ii) For all $L^A_d, L^B_d \in E_{d-1}(T_{d-1})$, the promise is kept for the two inputs $E^A_{d-1}(L^A_d), E^B_{d-1}(L^B_d)$ at all $i \in \sigma_{d-1} \cup \tau_{\le d-1}$.

(iii) (Only applies to Λ, the trivial unique root of the hierarchy tree for T_{d-1}, and $X^{T_{d-1}}(\Lambda) = T_{d-1}$.) For all $i \in \tau_d - \sigma_{d-1}$, $Pred_i(T_{d-1}) \le 2^{-(s+a_3+l)}$.

(iv) Does not apply.

Then the last s bits of the protocol (on inputs $E^A_{d-1}(E_{d-1}(M))$ for player A and $E^B_{d-1}(E_{d-1}(M))$ for B) partition T_{d-1} into at most 2^s classes. (The first $s(d-1)$ bits are always C_{d-1}). We pick T_d to be the largest class, and C_d to be the corresponding conversation. Then we have:

Lemma 6.3.

(i) For all $L_d \in E_{d-1}(T_d)$, $E^A_{d-1}(L_d) \in S^A_{C_d}$ and $E^B_{d-1}(L_d) \in S^B_{C_d}$.

(ii) For all $L^A_d, L^B_d \in E_{d-1}(T_d)$, the promise is kept for the pair of inputs $E^A_{d-1}(L^A_d)$ and $E^B_{d-1}(L^B_d)$ at all $i \in \sigma_{d-1} \cup \tau_{\le d-1}$.

(iii) For all $i \in \tau_d - \sigma_{d-1}$, $Pred_i(T_d) \le 2^s \cdot 2^{-(s+a_3+l)} = 2^{-(a_3+l)} \le 1/(16k^{4d}) \le 1/(16|\tau_d|)$.

6.7. Picking the Inputs. The protocol ends after sd bits and outputs $I = Out(C_d) \in \tau_{d+1}$, claiming that $b_I^A \neq b_I^B$. We claim there exist L^A and L^B such that they keep all promises, are consistent with C_d, and $b_I^A = b_I^B$.

The adversary chooses these inputs by choosing two elements M^A, M^B independently and uniformly at random from the set T_d. This determines the input vectors, by $L_d^A = E_{d-1}(M^A)$, $L_d^B = E_{d-1}(M^B)$, $L^A = E_{d-1}^A(L_d^A)$ and $L^B = E_{d-1}^B(L_d^B)$. We show that, with non-zero probability, all promises are kept and $b_I^A = b_I^B$.

From Lemma 6.3, we know that the pair is always consistent with C_d and that all promises are kept at all $i \in \tau_{\leq d-1} \cup \sigma_{d-1}$.

We now compute a lower bound on the probability that the pair of inputs is as promised at nodes in $\tau_d - \sigma_{d-1}$. For the promise to be kept at such nodes, it suffices that, for each such i, $\vec{cv}_i^A \neq \vec{cv}_i^B$. Now, $\vec{cv}_i$ and $\vec{cv}_i'$ are parts of M^A, M^B, which are uniform and independent samples from T_d. Lemma 4.4 states that the probability, for two samples randomly chosen (with replacement) from T', that the i^{th} component is the same, is no more than $Pred_i(T_d) \leq 1/(16|\tau_d|)$ (by Lemma 6.3 (iii)). Thus, the probability that one of the $|\tau_d| - |\sigma_{d-1}|$ such child vectors is the same is no more than $1/16$. Thus, the input pair is as promised with probability at least $15/16$.

As in Section 5.5, we have the probability that $b_I^A = b_I^B$ is at least $1/2$, since they are the same bit in L_d^A and L_d^B, and these two are both uniformly and independently distributed in the same set $E_{d-1}(T_d)$.

Thus, the probability that the pair is as promised and $b_I^A = b_I^B$ is at least $15/16 - 1/2 = 3/16$. Therefore, there exists a choice of inputs which causes the protocol to fail. Thus, there is no valid protocol with communication complexity at most $sd = dk - O(d^2(k \log k)^{1/2})$.

7. The Multiplexor Game, $d = 1$

DEFINITION 7.1. *The* **Same Function 1-Multiplexor** *communication game is played on a fixed set of vectors $U \subseteq \{0,1\}^k$. An adversary chooses two vectors $v_0, v_1 \in U$ and one function $f \in \{0,1\}^U$, with the restriction that $f(v_0) = 0$ and $f(v_1) = 1$. Player P_0 is given (v_0, f) and player P_1 is given (v_1, f). The players' goal is to find an index $i \in [1, k]$ for which the vectors are different.*

In fact, this is the Iterated Multiplexor Game for $d = 1$ in which the function f is known to the players (instead of the adversary only promising that there is such a function). It is the Karchmer-Wigderson R_f game, when f is part of the input.

THEOREM 7.2. *The Same Function 1-Multiplexor Game requires* $\Omega(\log |U|)$ *bits of communication. In particular, for* $U = \{0,1\}^k$, $\Omega(k)$ *bits are needed.*

Note that Theorem 7.2 can be proved by a simple counting argument. Riordan & Shannon (1942) proved there are many more functions in $\{0,1\}^{\{0,1\}^k}$ than there are circuits with depth $o(k)$. Since $CC(R_f) = Depth(f)$, it follows that there are functions with high communication complexity. If the adversary gives both the players such a function, the lower bound follows. However, the counting argument of Riordan & Shannon (1942) is notorious for yielding no intuition for constructing hard functions. Hopefully, our new techniques will provide new insight especially when combined with other communication complexity type proofs.

The following proof also provides techniques which might be useful in translating our lower bound for the Universal Composition Relation into an actual circuit lower bound. A major tool used in the lower bound for the Universal Composition Relation is to keep symmetry between the players to as large an extent as possible. As a bit communicated by one player restricts that player's set of possible inputs, we want to restrict the other player's inputs in the same way. However, in the communication game for any real function, this symmetry is lost from the very start, since one player has only inputs from $f^{-1}(0)$ and the other has inputs from $f^{-1}(1)$.

This lack of symmetry is even a problem in the simple Same Function 1-Multiplexor game. For any f, the set of inputs that the two players can have are disjoint, and we must give the players the same function f. Our lower bound has to get around this obstacle, which is also a major problem for converting the Universal Composition Relation bound into a real circuit bound. We do this by viewing the entire input as being composed of two parts, the function and the vector. Although the set of vector-function pairs allowed for the two players are disjoint, we keep both the set of possible vectors and the set of possible functions symmetrical for the two players. We also need to maintain a fullness property, which is that for any vector still in our set and any function still in our set, the pair can be given as an input to one of the two players. In effect, we find a smaller version of the Same Function 1-Multiplexor game after each bit of communication.

This separation of the input into several parts and maintaining symmetry for each part might be useful in proving circuit lower bounds for larger depth Multiplexors, but the situation is certainly much more complicated. A somewhat simpler game where these techniques might apply is the direct sum of two Same Function 1-Multiplexor games. To get a lower bound significantly more than k for this direct sum game is an open problem.

PROOF OF THEOREM 7.2. Given a fixed communication protocol, the adversary goes through the protocol round by round maintaining a set of vectors $V \subseteq \{0,1\}^k$ and a set of functions $F \subseteq \{0,1\}^{\{0,1\}^k}$ from which v_0, v_1, and f are chosen. We prove by induction on t that:

LEMMA 7.3. *Fix any protocol for the Same Function 1-Multiplexor game on U. Let $c = 12.27$. Let $0 \le t < \log_c(|U|)$. Then there is a partial conversation $C \in \{0,1\}^t$, a set $V \subseteq U$, and a set of functions $F \subseteq \{0,1\}^U$ so that:*

(i) $\forall v \in V$, $\forall f \in F$, *if* $f(v) = 0$, *then* $\langle v, f\rangle \in S_C^A$ *and if* $f(v) = 1$, *then* $\langle v, f\rangle \in S_C^B$.

(ii) $|V| \ge \frac{|U|}{c^{t-1}}$

(iii) $F_V = \{0,1\}^V$

Let $F_V \subseteq \{0,1\}^V$ denote the set of projections of elements of F onto V, i.e., the set of functions mapping vectors in V to $\{0,1\}$ which are consistent with some function in F. Because we are no longer considering the vectors not in V, we do not care what value f takes on these vectors. Property 3 states that all functions defined on V are possible. In fact, the proof is easier if we assume that for every function in F_V, there is one and only one function consistent with it in F. In this case, $|F| = 2^{|V|}$.

If we can prove Lemma 7.3, the theorem follows. Applying it to $U = \{0,1\}^k$ and $t = (k-2)/\log c$ bits of communication, by property 2, V still contains at least three vectors. At this point, the communication game is over and the protocol must specify an index $\in [1,k]$ for which the vectors v_0 and v_1 are different. At least two of the three vectors in V have the same bit at the specified index. One of these two vectors is given to the A-player and the other to the B-player. By Property 3, there exists a function $f \in F$ for which $f(v_0) = 0$ and $f(v_1) = 1$. This function is given to both players. By Property 1, the protocol's communication pattern on (v_0, v_1, f) is as stated. It follows that the protocol fails to find a difference in the vectors given these inputs. So it remains to prove Lemma 7.3.

Clearly, when $t = 0$, $V = U$, $F = \{0,1\}^U$, and the three properties hold. Assume that C, V, F meet the three properties for time $t-1$ and that given the conversation C, the A- player communicates during round t. For every pair (v, f) for which $f(v) = 0$, the protocol determines the bit the A-player will communicate if given the pair. We will denote this bit by $b^{v,f}$. The rules of the game ensure that the A-player will never be given a pair (v, f) for which $f(v) = 1$. Therefore, the protocol does not specify a bit for this pair. In this case, define $b^{v,f} = *$. (This represents the fact that the adversary is able to set this bit to 0 or 1 as needed.) With this notation, the statement "If the conversation so far is C, and $f(v) = 0$, then the A-player communicates the bit a given the pair (v, f) in round t", becomes simply "$[b^{v,f} \in \{a, *\}]$".

The adversary proceeds in three steps to obtain the new values for C, V, and F.

1. First, she finds $F' \subseteq F$ of size $(3/2)^{|V|}$ such that $\forall v \in V$, $\exists a^v \in \{0,1\}$, $\forall f \in F'$, $[b^{v,f} \in \{a^v, *\}]$.
2. Next, she finds $V' \subseteq V$ of size $\frac{2|V|}{c}$ such that $F'_{V'} = \{0,1\}^{V'}$.
3. Finally, she finds a bit $C_t \in \{0,1\}$ and a subset $V'' \subseteq V'$ of size $\frac{|V|}{c}$ such that $\forall v \in V''$, $\forall f \in F'$, $[b^{v,f} \in \{C_t, *\}]$.

It should be clear that, after these steps, C, C_t, V'' and F' satisfy the three properties of the Lemma.

Before we give a formal proof that step 1 is possible, we give an intuitive argument for motivation. The goal of step 1 is to find a subset F' of the functions for which $b^{v,f}$ is constant with respect to f for each v. Consider any $v \in V$. This vector partitions F into 3 parts: those f's for which $b^{v,f}$ equals 0, 1, and *, i.e 0 is communicated, 1 is communicated, and $f(v) \neq 0$. The adversary is free to choose between the 0 and the 1 part, by setting a^v appropriately and, either way, is able to keep the entire * part. The * part (i.e., those $f \in F$ for which $f(v) = 1$) is likely to make up about half of F, and the adversary is able to choose the larger of the 0 and the 1 parts; so meeting the goal for any particular v, should be possible while restricting F by at most a factor of 3/4. Initially, F is of size $2^{|V|}$. Therefore, we expect $|F'|$ to be of size $2^{|V|}(3/4)^{|V|} = (3/2)^{|V|}$. The problem with this argument is the statement that half the v, f pairs should give a *, which is true initially, but is not guaranteed to remain true as we restrict F.

A formal argument goes as follows. The vector $\vec{a} = a^{v_1} \dots a^{v_{|V|}} \in \{0,1\}^V$ is said to be consistent with the function $f \in F$, if $\forall v \in V$, $[b^{v,f} \in \{a^v, *\}]$. In other words, $\vec{a}$ and f are consistent, if $\vec{a}$ agrees with the bit the A-player will communicate during time step t if given v, f, whenever this last is actually possible. Step 1 can be rephrased as the adversary's finding a $\vec{a}$ which is consistent with $(3/2)^{|V|}$ functions in F. Towards this goal, she constructs a $2^{|V|} \times F_V = 2^{|V|} \times 2^{|V|}$ Boolean matrix M. Each row is labeled by an $\vec{a} \in \{0,1\}^V$

and each column is labeled with a function $f \in F$ with the f_V's taking on all possible values. The entry $M_{\vec{a},f}$ is set to be 1 if and only if $\vec{a}$ and f are consistent.

For a given $f \in F$, let $l \in [0, 2^{|V|}]$ be the number of $v \in V$ for which $f(v) = 1$, i.e., for which $b^{v,f} = *$. Then the number of $\vec{a}$ consistent with f is exactly 2^l, since we are free to pick $a(v) \in \{0,1\}$ for each such v. This is also the number of 1's in the column labeled by f. Because $F_V = \{0,1\}^V$, there are $\binom{|V|}{l}$ functions $f \in F$ where the number of $v \in V$ for which $f(v) = 1$ is l. It follows that the total number of 1's in the matrix is $\sum_{l=0}^{|V|} \binom{|V|}{l} 2^l = (1+2)^{|V|} = 3^{|V|}$.

There are $2^{|V|}$ rows. Therefore, there exists an $\vec{a}$ for which its row contains $\frac{3^{|V|}}{2^{|V|}} = (3/2)^{|V|}$ 1's. Let F' be those $f \in F$ that have a 1 in this row. It follows that $\forall v \in V,\ \exists a^v \in \{0,1\},\ \forall f \in F',\ [b^{v,f} \in \{a^v, *\}]$. This completes step 1.

For step 2, we use the following result from extremal set theory:

LEMMA 7.4 (Sauer 1972). *Let $S \subseteq \{0,1\}^n$, $|S| \geq \sum_{j=0}^{d} \binom{n}{j}$. Then there exist a set of d coordinates on which the projection of S is the whole d-cube.*

We view F' as a subset of $\{0,1\}^V$. Now, $|F'| \geq (3/2)^{|V|} \geq \sum_{l=0}^{2|V|/c} \binom{|V|}{l}$, for some suitable constant c ($c = 12.27$ works). So, applying Lemma 7.4 to the family F' and letting V' be the guaranteed set of coordinates gives us step 2.

Step 3 is very straightforward. From step 1, the bit communicated by A does not depend on the function $f \in F'$, but does depend on v. Label each $v \in V'$ with the bit a^v in the vector $\vec{a}$ chosen above. Partition V' according to this labeling and let V'' be the larger half and C_t the corresponding bit. It follows that $\forall v \in V''$, $\forall f \in F'$, if $f(v) = 0$ then, after conversation C, the A-player communicates bit C_t if given input (v, f). This gives us Property 3. $\square$

Acknowledgements

We would like to thank Toni Pitassi, Dan Simon, and Faith Fich for their insightful conversations and support.

Partially supported by cooperative research grant INT-9600919/ME-103 from the NSF (USA) and the MŠMT (Czech republic). Jeff Edmonds partially supported by grant NSERC A9176. Russell Impagliazzo partially supported by NSERC Post-doctoral fellowship at Univ. of Toronto, NSF YI Award CCR92-570979, NSF Award CCR-9734911, Sloan Research Fellowship BR-3311, grant #93025 of the joint US-Czechoslovak Science and Technology Program, and USA-Israel BSF Grant 97-00188. Jiří Sgall partially supported by grants A1019602 and A1019901 of GA AV ČR, and grants 201/97/P038 and 201/01/1195 of GA ČR.

References

JEFF EDMONDS, RUSSELL IMPAGLIAZZO, STEVEN RUDICH & JIŘÍ SGALL (1991). Communication Complexity Towards Lower Bounds on Circuit Depth. In *Proc. of the 32nd Ann. IEEE Symp. on Foundations of Computer Sci.*, 249–257. IEEE.

JOHAN HÅSTAD & AVI WIGDERSON (1993). Composition of the Universal Relation. In *Advances in Computational Complexity Theory*, JIN-YI CAI, editor, volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 119–134. AMS.

MAURICIO KARCHMER, RAN RAZ & AVI WIGDERSON (1995). Super-Logarithmic Depth Lower Bounds via Direct Sum in Communication Complexity. *Comput. Complexity* **5**(3/4), 191–204.

MAURICIO KARCHMER & AVI WIGDERSON (1990). Monotone Circuits for Connectivity Require Super-Logarithmic Depth. *SIAM J. Disc. Math.* **3**(2), 255–265.

EYAL KUSHILEVITZ & NOAM NISAN (1996). *Communication Complexity*. Cambridge University Press.

RAN RAZ & PIERRE MCKENZIE (1999). Separation of the Monotone NC Hierarchy. *Combinatorica* **19**, 403–435.

ALEXANDER A. RAZBOROV & STEVEN RUDICH (1997). Natural Proofs. *J. Comput. Syst. Sci.* **57**, 127–143.

JOHN RIORDAN & CLAUDE E. SHANNON (1942). The Number of Two-Terminal Series-Parallel Networks. *J. of Mathematics and Physics* **21**, 83–93.

NORBERT SAUER (1972). On the Density of Families of Sets. *J. of Combinatorial Theory (A)* **13**, 145–147.

GÁBOR TARDOS & URI ZWICK (1997). The Communication Complexity of the Universal Relation. In *Proc. of the 12th Ann. IEEE Conf. on Computational Complexity*, 247–259. IEEE.

Manuscript received July 30, 1999

JEFF EDMONDS
Department of Computer Science
York University
North York, Ontario M3J 1P3, Canada
`jeff@cs.yorku.ca`

RUSSELL IMPAGLIAZZO
Department of Computer Science
UC San Diego
La Jolla CA 92093, U.S.A.
`russell@cs.ucsd.edu`

Steven Rudich
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, U.S.A.
rudich@theory.cs.cmu.edu

Jiří Sgall
Mathematical Institute, AS CR
Žitná 25
115 67 Praha 1, Czech Republic
sgall@math.cas.cz
http://www.math.cas.cz/~sgall/