Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 62 (2004)

On the P versus NP intersected with co-NP question in communication complexity

Stasys Jukna *†‡§

Abstract

We consider the analog of the P versus NP ∩ co-NP question for the classical two-party communication protocols where polynomial time is replaced by polylogarithmic communication: if both a boolean function f and its negation $\neg f$ have small (poly-logarithmic in the number of variables) nondeterministic communication complexity, what is then its deterministic and/or probabilistic communication complexity? In the *fixed* (worst) *partition* case this question was answered by Aho, Ullman and Yannakakis in 1983: here P = NP ∩ co-NP.

We show that in the *best-partition* case the situation is entirely different: here P is a proper subset even of RP ∩ co-RP, and NP ∩ co-NP is no longer a subset of BPP. This, in particular, resolves an open question raised by Papadimitriou and Sipser in 1982. We also extend to the best-partition case the result of Rabin and Yao that NP is not a subset of BPP in the fixed-partition model.

Key words: communication complexity, best-partition protocols, lower bounds
AMS subject classification: 03D15, 68Q17, 68Q15, 68Q10

1 Introduction

Understanding the relative power of determinism, nondeterminism, and randomization is fundamental in any model of computation. In the Turing machine model this leads to the well-known P versus NP versus BPP and similar questions. While in this model such questions remain widely open, some progress was made in several simpler (but still important) models, like decision trees or communication protocols.

* *Current Address*: Universität Frankfurt, Institut für Informatik, Robert-Mayer-Str. 11-15, D-60054 Frankfurt am Main, Germany
† On leave from Institute of Mathematics and Informatics, Akademijos 4, LT-2600 Vilnius, Lithuania
‡ *Email*: jukna@thi.informatik.uni-frankfurt.de
§ Research supported in part by a DFG grant SCHN 503/2-1.

ISSN 1433-8092

In the decision tree model when the complexity measure is the *depth* of a tree we have that P = NP ∩ co-NP [3, 7, 20]: if both f and $\neg f$ can be computed nondeterministic decision trees of depth at most d then f can be computed by a deterministic decision tree of depth at most d^2. Nisan [16] has shown that in this case also P = BPP holds: if f can be computed by a probabilistic bounded error decision tree of depth d then f can be computed by a deterministic decision tree of depth $O(d^3)$.

Interestingly, the situation is different if we measure the *size* of (the total number of vertices in) a tree instead of its depth—then P ≠ NP ∩ co-NP [11]: there are explicit boolean functions f such that both f and $\neg f$ can be computed by nondeterministic decision trees of size at most N but any deterministic decision tree for f has size $N^{\Omega(\log N)}$. A similar situation is when the complexity measure of a decision tree is the number of its non-isomorphic subtrees (this model corresponds to so called read-once branching programs—just merge isomorphic subtrees): here we also have that P ≠ NP ∩ co-NP [11], and even NP ∩ co-NP ⊈ BPP [19].

In this paper we consider similar questions in the classical model of two-party communication protocols introduced by Yao in [21, 22] (see a survey [14] or monographs [8, 13] for more information). There are two main types of such protocols: the **fixed-partition** type where the protocol must use some prescribed (by an adversary) "bad" partition of input variables between the players, and **best-partition** type where the protocol is allowed to choose the "most suitable" for a given function partition of its variables. There are also three natural modes of communication: deterministic, nondeterministic and probabilistic. Having these modes and having the (admittedly far-fetched) analogy with the P versus NP question, one may ask whether, say, NP = co-NP or P = NP ∩ co-NP or P = RP ∩ co-RP or NP ∩ co-NP ⊆ BPP in the context of communication protocols; a systematic study of these classes was started by Babai, Frankl, and Simon in [2]. Here for convenience (and added thrill, just like in [2]) we use the common names for the analogs of the complexity classes:

> P (resp., NP, BPP and RP) consists of all boolean functions in n variables whose deterministic (resp., nondeterministic, probabilistic bounded error, and probabilistic one-sided error) communication complexity is polynomial in $\log n$.

It is clear that P ⊆ RP ⊆ NP ∩ BPP in both the fixed- and the best-partition case.

In the **fixed-partition** case most of these problems are already solved. In particular, the following facts are known (of course, this has nothing to do with the relations between Turing machine classes):

1. NP ≠ co-NP, and hence, P ≠ NP. This can be easily shown using the equality function $\text{EQ}(x, y)$ which tests whether two given binary strings x and y of length n are equal.

2. BPP ⊈ NP and P ≠ RP. This was shown by Rabin and Yao (see [23]): $\neg\text{EQ}(x, y)$ can be computed by a probabilistic (even one-sided $1/n$-error) protocol with only $O(\log n)$ bits of communication.

3. NP $\not\subseteq$ BPP. This was proved by Babai, Frankl, and Simon [2] using the set-disjointness function $\text{DISJ}(x, y)$, which outputs 1 iff $\sum_{i=1}^{n} x_i y_i = 0$. They proved that this function has probabilistic communication complexity $\Omega(\sqrt{n})$. This lower bound was improved to $\Omega(n)$ by Kalyanasundaram and Schnitger [12]; a simpler proof was found by Razborov [18]. In [2] it was also shown that (just like in the case of Turing machines) BPP $\subseteq \Sigma_2 \cap \Pi_2$ where Σ_2 and Π_2 are the analogs of classes in the second level of the polynomial hierarchy. The question of [2] on whether $\Sigma_2 \neq \Pi_2$ remains open.

4. P = RP $\cap$ co-RP = NP $\cap$ co-NP $\subsetneq$ BPP. This is a direct consequence of the following well-known result of Aho, Ullman, and Yannakakis [1]: if f and $\neg f$ have nondeterministic communication complexities n_f and $n_{\neg f}$, then the deterministic communication complexity of f does not exceed $O\left(\max\{n_f, n_{\neg f}\}^2\right)$. This result was later improved in different ways. Halstenberg and Reischuk [6] have strengthened the upper bound to the form $O(n_f \cdot n_{\neg f})$. Lovász and Saks [15] have strengthened this last upper bound by showing that n_f can be replaced by the triangular rank of the corresponding communication matrix of f. The tightness of the theorem was proved by Fürer [5] using so-called "list inequality" function. This was recently improved by Jayram, Kumar and Sivakumar [9] by showing that, for the iterated set-disjointness function f, even the probabilistic bounded error communication complexity is $\Omega(n_f \cdot n_{\neg f})$.

The **best-partition** model is more difficult to analyze, and here the situation was less clear. In particular, such "clean" functions like $\text{EQ}(x, y)$ or $\text{DISJ}(x, y)$ cannot be used (at least directly) for separations anymore just because even the deterministic communication complexity of these functions is constant.

The first separation in the best-partition model was given by Papadimitriou and Sipser in [17]. Using the triangle-freeness property of graphs and an elegant combinatorial argument, they proved that NP $\neq$ co-NP also in this case. The proof was via a reduction to computing $\text{DISJ}(x, y)$ in the fixed-partition case. Using probabilistic arguments, this result was extended in [10] to the case where a protocol is allowed to use different partitions for different inputs (the logarithm of the number of used partitions is, however, a part of the complexity): even in this model the triangle-freeness function has exponentially high nondeterministic communication complexity. This, together with the fact, proved in [4], that using $k+1$ partitions instead of k partitions may exponentially decrease the number of communicated bits, shows that in the context of communication complexity the corresponding classes NP and co-NP are indeed very different.

In the same paper [17] (this was one of the first papers to study questions of this type in communication complexity) Papadimitriou and Sipser asked whether P $\neq$ NP $\cap$ co-NP for the best-partition protocols. The question is important because it exposes something about the power of lower bound arguments. That is, fooling set and other arguments, used for the best-partition protocols, apply not only to deterministic but also to nondeterministic protocols. We can prove a lower bound on the deterministic communication

complexity of a function f by arguing about either f or $\neg f$. But if *both* the function and its negation have low nondeterministic complexity under some partitions of variables, other arguments are needed to show that the deterministic communication complexity must be large for *any* partition.

To our best knowledge this question (as well as other similar questions for best-partition protocols) remained unsolved. The only result in this direction we are aware of is the claim in [1] that an appropriate modification of the triangle-freeness function should separate P from NP $\cap$ co-NP in the best-partition case. Unfortunately, the proof—which should (apparently) involve the argument for the triangle-freeness function used in [17]—was never published.

2 Our results

In this paper we answer the question of [17]: P $\neq$ NP $\cap$ co-NP for the best-partition protocols. Actually, we establish even stronger separations, implying that in the best-partition case the situation is entirely different: here we have that

$$\mathrm{P} \subsetneqq \mathrm{RP} \cap \text{co-RP} \subsetneqq \mathrm{NP} \cap \text{co-NP} \not\subseteq \mathrm{BPP}. \tag{1}$$

Moreover, we show that the result of Rabin and Yao ([23])—that BPP $\not\subseteq$ NP in the fixed-partition case—can be extended to the best partition case. Note that the result of Babai et al. [2]—that NP $\not\subseteq$ BPP in the fixed-partition case—is already extended to the best-partition case by the last relation in Eq. (1).

All this is a direct consequence of the following theorem. We adopt the following convention for discussing different communication complexity measures of f in the *best-partition* case: $D(f)$ for the deterministic, $N(f)$ for the nondeterministic, $R(f)$ for the probabilistic bounded error, and $R^1(f)$ for probabilistic one-sided error communication complexity.

Theorem 2.1. *There are explicit boolean functions f, g and h in n^2 variables such that:*

(i) *both $N(f)$ and $N(\neg f)$ are constants but $R(f) = \Omega(n)$;*

(ii) *both $R^1(g)$ and $R^1(\neg g)$ are $O(\log n)$ but $D(g) = \Omega(n)$.*

(iii) *$R(h) = O(\log n)$ but $N(h) = \Omega(n)$.*

The proofs of these three claims themselves are relatively simple—as it often happens with the results of this type, most of the work is done by a careful choice of separating functions.

3 Proofs

Recall that in the best-partition case the players can choose different (most suitable) partitions for a function and its negation. To visualize the effect of this choice, in all three cases we define the corresponding separating function $f(X)$ as boolean functions in n^2 variables, arranged into an $n \times n$ matrix. Hence, inputs for f are 0/1 matrices $A : X \to \{0,1\}$. We define $f(X)$ in such a way that a partition of X according to columns is suitable for computing f, and that according to rows is suitable for $\neg f$.

3.1 Proof of Theorem 2.1(i)

We define the boolean function $f(X)$ (let us call it the *good matrix function*), separating NP $\cap$ co-NP from BPP in the best-partition case, as follows. Say that a row/column x of such a matrix is *good* if it contains precisely two 1's, and *bad* otherwise. Let $f(A) = 1$ if and only if

(i) at least one row of A is good, and

(ii) all columns of A are bad.

Lemma 3.1. *Both $N(f)$ and $N(\neg f)$ are constants.*

Proof. To compute $f(X)$ the players take a partition of X where Alice gets the first half of the *columns* and Bob gets the rest. Given an input matrix $A : X \to \{0,1\}$, the protocol first guesses a row r (a candidate for a good row). Then, using 3 bits, Alice tells Bob whether all her columns are bad, and whether the first half of the row r contains none, one, two or more 1's. After that Bob has the whole information about the value $f(A)$, and can announce the answer.

In order to compute $\neg f(X)$ the players take a partition of X where Alice gets the first half of the *rows* and Bob gets the rest. Given an input matrix $A : X \to \{0,1\}$, the protocol first guesses a column c (a candidate for a good column). Then, using 3 bits, Alice tells Bob whether there is a good row among her rows, and whether the first half of the column c contains none, one, two or more 1's. After that Bob again has the whole information about the value $f(A)$, and can announce the answer. □

In the proof of a lower bound on $R(f)$ we will use the fact (mentioned in the Introduction) that in the fixed-partition case, where Alice gets x and Bob gets y, the set-disjointness function DISJ(x,y), which outputs 1 iff $\sum_{i=1}^{n} x_i y_i = 0$, has probabilistic bounded error communication complexity $\Omega(\sqrt{n})$ [2], and even $\Omega(n)$ [12, 18].

Lemma 3.2. $R(f) = \Omega(n)$.

Proof. Take an arbitrary probabilistic bounded error protocol for $f(X)$. The protocol uses some balanced partition of X into two halves where the first half is seen by Alice

and the second by Bob. Say that a column is seen by Alice (resp., Bob) if Alice (resp., Bob) can see all its entries. A column is *mixed* if it is seen by none of the two players, that is, if each player can see at least one its entry. Let m be the number of mixed columns. We consider two cases depending on how large this number m is. In both cases we describe a "hard" subset of inputs, i.e. a subset of input matrices on which the players need to communicate many bits.

Case 1: $m \leq n/2 - 1$. Since each player can see at most $n/2$ columns, we have that in this case each player will see at least $n - (n/2 + m) \geq 1$ columns. Take one column seen by Alice and another column seen by Bob, and let Y be the $(n-3) \times 2$ submatrix of X formed by these two columns without the last three rows. We restrict the protocol to input matrices $A : X \to \{0,1\}$ defined as follows. We first set all entries in the last three rows to 1. This way we ensure that all columns of A are already bad. Then we set all remaining entries of X outside Y to 0. The columns x and y of Y may take arbitrary values.

In each such matrix all columns are bad and, since $n \geq 3$, the last three all-1 rows are also bad. Thus, given such a matrix, the players must determine whether some of the remaining rows is good. Since all these rows have 0's outside the columns x and y, this means that the players must determine whether $x_i = y_i = 1$ for some $1 \leq i < n-3$. That is, they must compute $\neg\textsc{disj}(x,y)$ which requires $\Omega(n)$ bits of communication.

Case 2: $m \geq n/2$. Let Y be the $n \times m$ submatrix of Y formed by the mixed columns. Select from the i-th ($i = 1, \ldots, m$) column of Y one entry x_i seen by Alice and one entry y_i seen by Bob. Since $m \leq n$ and we select only $2m$ entries, there must be a row r with $t \leq 2$ selected entries. Let Y be the $n \times (m-t)$ submatrix consisting of the mixed columns with no selected entries in the row r. We may assume that $m - t$ is odd and that $m - t \leq n - 2$ (if not, then just include in Y fewer columns).

Now restrict the protocol to input matrices $A : X \to \{0,1\}$ defined as follows. First we set to 1 some two entries of the row r lying outside Y, and set to 0 all the remaining entries of r. This ensures that the obtained matrices will already contain a good row. After that we set all the remaining non-selected entries of X to 0. Since each obtained matrix A contains a good row (such is the row r) and all columns outside the submatrix Y are bad (each of them can have a 1 only in the row r), the players must determine whether all columns of A in Y are also bad. Since all non-selected entries of Y are set to 0, the players must determine whether $x_i + y_i \leq 1$ for all $i = 1, \ldots, m-t$. Hence, the players must decide whether $\sum_{i=1}^{m-t} x_i y_i = 0$, that is, to compute the set-disjointness function $\textsc{disj}(x,y)$), which again requires $\Omega(m-t) = \Omega(n)$ bits of communication.

This completes the proof of Lemma 3.2, and thus, the proof of Theorem 2.1(i). □

3.2 Proof of Theorem 2.1(ii)

We define the boolean function $g(X)$ (let us call it the *odd-even function*), separating RP $\cap$ co-RP from P in the best-partition case, as follows. As before, inputs for g are

$n \times n$ matrices; this time we require that n is even. Say that a row/column of such a matrix is *odd* (*even*) if it contains an odd (even) number of 1's. Let $g(A) = 1$ if and only if

(i) A has at least one odd row, and

(ii) all columns of A are odd.

The *non-equality function* is a function $\text{NE}(x, y)$ in $2n$ variables such that $\text{NE}(x, y) = 1$ iff $x \neq y$, i.e. is the strings x and y differ in at least one coordinate. The variables themselves may not be boolean—they can take their values in any range $\{0, 1, \ldots, m\}$ with $m \leq n^2$. We will use a well-known fact (see, e.g., Example 3.9 in [13]) that in the fixed-partition case, where Alice gets x and Bob gets y, the function $\text{NE}(x, y)$ can be computed with a probabilistic one-sided error protocol by communicating only $O(\log n)$ bits. (We will also use this fact later in the proof of Lemma 3.5.)

Lemma 3.3. *Both $R^1(g)$ and $R^1(\neg g)$ are $O(\log n)$.*

Proof. Using the same partitions of X as in the proof of Lemma 3.1, we see that the computation of the odd-even function g and its negation $\neg g$ reduces to the computation of the non-equality function $\text{NE}(x, y)$, where x is a string of parities of rows/columns seen by Alice and y is a string of parities of rows/columns seen by Bob. Indeed, to compute g it is enough to decide whether $x \neq y$ (there is an odd row) whereas for $\neg g$ it is enough to decide whether $x \neq y \oplus \mathbf{1}$ (there is an even column). □

Lemma 3.4. $D(g) = \Omega(n)$.

Proof. Take an arbitrary deterministic protocol for $g(X)$ using some balanced partition of the $n \times n$ matrix of variables X. Assume that n is even, and let (as before) m be the number of mixed columns.

Case 1: $m \leq n/2 - 1$. In this case each player can see at least one column. Take one column seen by Alice and another column seen by Bob, and let Y be the $(n-1) \times 2$ submatrix of X formed by these two columns without the last row r. We restrict the protocol to input matrices $A : X \to \{0, 1\}$ defined as follows. We set to 1 all entries in the last row r, and set to 0 all remaining entries of X outside Y. The columns x and y of Y may take arbitrary values such that the resulting vectors are even. This way we ensure that all columns of A are odd. Moreover, the last row r is even since n is even. Thus, given such a matrix A, the players must determine whether some of the remaining rows is odd. That is, they must determine whether $x \neq y$, which requires $\Omega(n)$ bits of communication.

Case 2: $m \geq n/2$. Let Y be the $n \times m$ submatrix of Y formed by the mixed columns. Select from the i-th ($i = 1, \ldots, m$) column of Y one entry x_i seen by Alice and one entry y_i seen by Bob. Since $m \leq n$ and we select only $2m$ entries, there must be a row r with

$t \leq 2$ selected entries. Let Y be the $n \times (m-t)$ submatrix ($t \leq 2$) consisting of the mixed columns with no selected entries in this row r. We may assume that $m-t$ is odd (if not, then just include one column less in Y).

Now restrict the protocol to input matrices $A : X \to \{0,1\}$ defined as follows. First we set the part of the row r lying in Y to 0's and the rest of r to 1's. Since n is even and $m-t$ is odd, this ensures that the obtained matrices will already contain an odd row. After that we set to 0 all the remaining non-selected entries of X. Since each obtained matrix A contains an odd row (the row r) and all columns outside the submatrix Y are odd (each of them has a 1 in the row r and 0's elsewhere), the players must determine whether all columns of A in Y are also odd. That is, they must determine whether $x_i \neq y_i$ for all $i = 1, \ldots, m-t$. Or equivalently, they must decide whether $x = y \oplus \mathbf{1}$ for vectors $x = (x_1, \ldots, x_{m-t})$ and $y = (y_1, \ldots, y_{m-t})$, which again requires $\Omega(m-t) = \Omega(n)$ bits of communication.

This completes the proof of Lemma 3.4, and thus, the proof of Theorem 2.1(ii). □

Note that the requirement that the partitions of the set of variables X are balanced is not crucial. The same argument works also in the case when the partitions are only "almost balanced" in a sense that each player gets access to at least a λ fraction of all variables, for some $0 < \lambda(n) \leq 1/2$. The proofs of Lemmas 3.2 and 3.4 remain the same with only one difference: this time we consider the two cases depending on whether $m \leq \lambda n - 1$ or not. Since each player can see at most $(1-\lambda)n$ rows, we have that in the first case each player will see at least $n - (1-\lambda)n - m \geq 1$ rows. The rest of the proof is the same. The obtained lower bounds on $R(f)$ and $D(g)$ are then of the form $\Omega(\lambda n)$.

3.3 Proof of Theorem 2.1(iii)

We define the boolean function $h(X)$ (let us call it the *polynomial function*), showing that BPP $\not\subseteq$ NP in the best-partition case, as follows. Let n be a sufficiently large prime number, and assume that the rows and columns of the matrix of variables $X = \{x_{i,j}\}$ are indexed by elements of the field $\mathrm{F} = \{0, 1, \ldots, n-1\}$ of residues modulo n. Set $d = \lfloor n/2 \rfloor - 1$ and let $P(n,d)$ be the set of polynomials $p(z)$ of degree at most d over F. Let $h(X) = 1$ if and only if there exists a polynomial $p \in P(n,d)$ such that for all $i, j \in \mathrm{F}$, $x_{i,j} = 1$ iff $j = p(i)$. That is, the function h accepts a given matrix $A : X \to \{0,1\}$ if and only if the 1's in A are consistent with some polynomial from $P(n,d)$ (represent the graph of this polynomial).

In the proof of both upper and lower bounds for h we will use the fact that no two distinct polynomials of degree at most d over F can take the same values on more than d distinct elements of F. This is a direct consequence of a fundamental theorem from algebra that the number of roots of a (non-zero) polynomial is bounded by its degree.

Lemma 3.5. $R(h) = O(\log n)$.

Proof. Take a balanced partition of X where Alice gets the first $\lfloor n/2 \rfloor$ of the columns and Bob gets the rest. Since $d = \lfloor n/2 \rfloor - 1$, each player has access to more than d columns. Given an input matrix $A : X \to \{0,1\}$ Alice checks whether there is a polynomial in $P(n,d)$ which is consistent with all the 1's in her part of A. If there is no such polynomial, then Alice tells this to Bob, and the game is over: the input matrix A is (correctly) rejected. If there is such a polynomial $p_A(z)$, then it is unique, because in this case the polynomial must take the prescribed (by the 1's in the Alice's part of A) values on more than d elements of F. Similarly, either Bob finds no polynomial in $P(n,d)$ which is consistent with all the 1's in his part of A (and the game is over) or he finds such a polynomial $p_B(z)$, and this polynomial is also unique. Thus, it remains to decide whether these two polynomials $p_A(z)$ and $p_B(z)$ are equal. That is, if $x, y \in \mathrm{F}^{d+1}$ are the strings of coefficients of p_A and p_B then the players have only to decide whether $x = y$. As mentioned right before Lemma 3.3, this can be done with a probabilistic bounded error protocol by communicating only $O(\log n)$ bits. □

Lemma 3.6. $N(h) = \Omega(n)$.

Proof. Take an arbitrary nondeterministic protocol for $h(X)$ whose communication complexity is $N(h)$. This protocol uses some balanced partition of X into two parts where the first part is seen by Alice and the second by Bob. Given an input matrix $A : X \to \{0,1\}$, let a and b denote the corresponding partial assignments to the variables seen, respectively, by Alice and by Bob; hence, $h(A) = h(a,b)$. Let also $|a|$ denote the number entries set to 1 by a.

To prove the desired lower bound on $N(h)$, let us consider the communication matrix M of h. This is a 0/1 matrix whose rows are labeled by assignments a to Alice's part and columns by assignments b to Bob's part; the (a,b)-th entry of M is $h(a,b)$. It is a "folklore observation" that the nondeterministic communication complexity of a boolean function is equal to the logarithm of the minimum number of all-1 sub-matrices covering all 1's of its communication matrix. The number of 1's in our matrix M is n^{d+1}, just because we have so many polynomials of degree at most d over F. Thus, it is enough to give an appropriate upper bound on the the maximum size of (number of entries in) an all-1 submatrix of M.

Claim 3.1. *Each all-*1 *submatrix of M lies entirely in one row or in one column of M.*

Proof. If $h(a,b) = 1$ then $|a| + |b| = n \geq 2(d+1)$. Hence, at least one of the assignments, say a, sets to 1 at least $d+1$ variables, implying that $h(a,b') = 0$ for all $b' \neq b$. □

By this claim, it remains to give a good upper bound on the maximal possible number of 1's in a line (row or column) of M. Since our polynomials have degree at most d, the number of 1's in every row (or column) a cannot not exceed $\max\{1, n^{d+1-|a|}\}$. However for $|a| = 0$ this trivial upper bound is useless. In order to get a better upper bound we

will use the fact that none of the players can see more than a half of the variables in X. Set

$$H(t) = \left(\frac{n^2}{2(n-t)}\right)^{d+1-t}.$$

Claim 3.2. *For every assignment a there are at most $H(|a|)$ assignments b to the remaining variables of X such that $h(a,b) = 1$.*

Proof. Let $t = |a|$. If $t \geq d+1$, then $h(a,b) = 1$ for at most one b. So, assume that $0 \leq t \leq d$, and that $h(a,b) = 1$ for at least one b. Say that a column of X is *free* if no its entry is set to 1 by a. Since the assignment a sets to 1 precisely t entries of X and cannot set to 1 more than one entry in any column (for otherwise there would be no extensions at all), we have $n-t$ free columns. Let s_i be the number of entries in the i-th free column seen by Bob. Assume w.l.o.g. that $s_1 \leq \ldots \leq s_{n-t}$.

Set $r = d+1-t$, and let $T(s_1, \ldots, s_r)$ be the number of all r-tuples $(j_1, \ldots, j_r)$ with $1 \leq j_i \leq s_i$, for $i = 1, \ldots, r$. Since $|a| + r = d+1$, the number $T(s_1, \ldots, s_r)$ is an upper bound on the number of possible extensions b for which $h(a,b) = 1$. The function $T(s_1, \ldots, s_r)$ achieves its maximum when $s_1 = \ldots = s_r$ and the s_i's take the maximum possible value. Since $s_1 \leq \ldots \leq s_{n-t}$ and $s_1 + \cdots + s_{n-t} \leq n^2/2$, this implies that $T(s_1, \ldots, s_r)$ does not exceed $\left(\frac{n^2}{2(n-t)}\right)^r = H(t)$. □

Since the function $H(t)$ is non-increasing, Claims 3.1 and 3.2 imply that none of the all-1 sub-matrices of M can have more than $H(0) = (n/2)^{d+1}$ entries. Hence, $N(h)$ is at least the logarithm of $n^{d+1}/(n/2)^{d+1}$, which is $d+1 = \Omega(n)$, as desired. □

Acknowledgments

I would like to thank Martin Sauerhoff for helpful comments on a prior draft and Georg Schnitger for interesting discussions.

References

[1] Aho, A., Ullman, J. and Yannakakis, M. (1983): On notions of information transfer in VLSI circuits. In *Proc. of 15th Ann. ACM Symp. on the Theory of Computing*, 133–139.

[2] Babai, L., Frankl, P. and Simon, J. (1986): Complexity classes in communication complexity theory. In *Proc. of 27th Ann. IEEE Symp. on Foundations of Comput. Sci.*, 337–347.

[3] Blum, M. and Impagliazzo, R. (1987): Generic oracles and oracle classes. In *Proc. of 28th Ann. IEEE Symp. on Foundations of Comput. Sci.*, 118–126.

[4] Ďuriš, P., Hromkovič, J., Jukna, S., Sauerhoff, M. and Schnitger, G. (2001): On multipartition communication complexity. In *Proc. of 18th STACS*, Springer Lecture Notes in Computer Science, vol. 2010, 206–217. Journal version in *Information and Computation* (to appear).

[5] Fürer, M. (1987): The power of randomness for communication complexity. In *Proc. of 19th Ann. ACM Symp. on the Theory of Computing*, 178–181.

[6] Halstenberg, B. and Reischuk, R. (1993): Different modes of computation, *SIAM J. Computing* **22**:5, 913–934.

[7] Hartmanis, J. and Hemachandra, L. A. (1987): One-way functions, robustness and non-isomorphism of NP-complete classes. Tech. Rep. DCS TR86-796, Cornell University.

[8] Hromkovič, J. (1997): *Communication Complexity and Parallel Computing*, Springer-Verlag.

[9] Jayram, T. S., Kumar, R. and Sivakumar, D. (2003): Two applications of information complexity. In *Proc. of 35th Ann. ACM Symp. on the Theory of Computing*, 673–682.

[10] Jukna, S. and Schnitger, G. (2002): Triangle-freeness is hard to detect, *Combinatorics, Probability & Computing* **11**, 549–569.

[11] Jukna, S., Razborov, A., Savický, P. and Wegener, I. (1999): On P versus NP∩co-NP for decision trees and read-once branching programs, *Computational Complexity* **8**:4, 357–370.

[12] Kalyanasundaram, B. and Schnitger, G. (1987): The probabilistic communication complexity of set intersection. In *Proc. of 2nd Structure in Complexity Theory*, 41–49. Journal version in: *SIAM J. Discrete Mathematics* **5**:4 (1992), 545–557.

[13] Kushilevitz, E. and Nisan, N. (1997): *Communication Complexity.* Cambridge University Press.

[14] Lovász, L. (1990): Communication complexity: A survey. In *Algorithms and Combinatorics* **9** (Springer-Verlag), 235–265.

[15] Lovász, L. and Saks, M. (1993): Lattices, Möbius functions and communication complexity, *J. Compu. Syst. Sci.* **47**, 322–349.

[16] Nisan, N. (1991): CREW PRAMs and decision trees, *SIAM Journal on Computing* **20**:6, 999–1007.

[17] Papadimitriou Ch. H. and Sipser M. (1982): Communication complexity. In *Proc. of 14th Ann. ACM Symp. on the Theory of Computing*, pp. 196–200. Journal version in: *J. Comput. Syst. Sci.*, **28**:2 (1984), 260–269.

[18] Razborov, A. (1990): On the distributional complexity of disjointness. In: *Proc. of 17th International Colloquium on Automata, Languages and Programming*, Springer Lecture Notes in Computer Science **443**, 249–253. Journal version in: *Theoretical Computer Science* **106**:2 (1992), 385–390.

[19] Sauerhoff, M. (2003): Randomness versus nondeterminism for read-once and read-k branching programs. In *Proc. of 20th Symposium on Theoretical Aspects in Computer Science*, Springer Lecture Notes in Computer Science **2607**, 307–318.

[20] Tardos, G. (1989) Query complexity, or why is it difficult to separate $\mathrm{NP}^A \cap \text{co-NP}^A$ from P^A by a random oracle A? *Combinatorica* **9**, 385-392.

[21] Yao, A. C. (1979): Some complexity questions related to distributed computing. In *Proc. of 11th Ann. ACM Symp. on the Theory of Computing*, 209–213.

[22] Yao, A. C. (1981): The entropic limitations of VLSI computations. In *Proc. of 13th Ann. ACM Symp. on the Theory of Computing*, 308–311.

[23] Yao, A. C. (1983): Lower bounds by probabilistic arguments. In *Proc. of 24th Ann. IEEE Symp. on Foundations of Comput. Sci.*, 420–428.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'