The Communication Complexity of the Universal Relation

Gábor Tardos * Uri Zwick †

Abstract

Consider the following communication problem. Alice gets a word $x \in \{0,1\}^n$ and Bob gets a word $y \in \{0,1\}^n$. Alice and Bob are told that $x \neq y$. Their goal is to find an index $1 \leq i \leq n$ such that $x_i \neq y_i$ (the index i should be known to both *of them). This problem is one of the most basic communication problems. It arises naturally from the correspondence between circuit depth and communication complexity discovered by Karchmer and Wigderson.*

We present three protocols using which Alice and Bob can solve the problem by exchanging at most $n+2$ bits. One of this protocols is due to Rudich and Tardos. These protocols improve the previous upper bound of $n + \log^* n$, obtained by Karchmer. We also show that any protocol for solving the problem must exchange, in the worst case, at least $n+1$ bits. This improves a simple lower bound of $n-1$ obtained by Karchmer. Our protocols, therefore, are at most one bit away from optimality.

*The three $n+2$ bit protocols use two completely different ideas and they each have some additional interesting properties. The simplest protocol (*SIMPLE*) always finds the* first *difference between x and y. It uses, however, about n rounds of communication. A more complicated version of this protocol (*LOGSTAR*) finds the first difference between x and y by exchanging at most $n+2$ bits in about $\log^* n$ rounds of communication. Our most surprising protocol (*HAM$_3$*) finds a difference, not necessarily the first one, between x and y by exchanging at most $n+2$ bits in at most 3 rounds of communication. Protocol* HAM$_3$ *uses the Hamming error-correcting code.*

We next consider protocols for finding the first difference using a limited number of rounds. For every $c \geq 2$, we present an oblivious *protocol that finds the first difference by exchanging $n + \lceil \log^{(c-1)} n \rceil + 1$ bits in c rounds of communication. We also show that any protocol that finds the first difference using at most c rounds must exchange at least $n + \lceil \log^{(c-1)} n \rceil - 2$ bits. These protocols are, therefore, at most 3 bits away from being optimal.*

Finally, we consider protocols for variants of the above communication problem. Our most surprising results are perhaps the following. Alice and Bob can exchange at most $n - \lfloor \log n \rfloor + 2$ bits, in only 2 rounds, after which Alice will know and index i such that $x_i \neq y_i$. Alice and Bob can exchange at most $n - \lfloor \log n \rfloor + 4$ bits, in at most 4 rounds, after which Alice will know and index i such that $x_i \neq y_i$ and Bob will know and index j such that $x_j \neq y_j$. Furthermore, $i = j$ unless x and y differ in exactly two places.

1 Introduction

Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. The *depth* of f, denoted by $D(f)$, is the minimal depth of a fanin-2 circuit, over the basis $\{\wedge, \vee, \neg\}$, computing f. Karchmer and Wigderson [KW90] established an elegant correspondence between the depth of a Boolean function f and the complexity of the following communication problem. Alice gets a word $x \in \{0,1\}^n$ such that $f(x) = 0$ while Bob gets a word $y \in \{0,1\}^n$ such that $f(y) = 1$. Clearly $x \neq y$. Alice and Bob must find an index $1 \leq i \leq n$ such that $x_i \neq y_i$ (the index i should be known to *both* of them). Let $C(f)$ be the communication complexity of this problem, i.e., the number of bits exchanged, in the worst case, by the best (deterministic) communication protocol that solves the problem. Karchmer and Wigderson [KW90] show that for every Boolean function f, the communication complexity of the communication problem corresponding to f is exactly equal to the depth of f. In other words, $D(f) = C(f)$.

Though the proof of the correspondence between circuit depth and communication complexity is extremely simple, the correspondence is an extremely powerful tool for studying circuit depth as it allows arguing in a 'top-down' manner. Karchmer and Wigderson [KW90] utilized the correspondence to obtain an $\Omega(\log^2 n)$ lower bound on the monotone depth of the st-connectivity problem. Raz

*Mathematical Institute of the Hungarian Academy of Sciences, Pf. 127, Budapest, H-1364 Hungary. Supported by NSF grants CCR-95-03254 and DMS-9304580, a grant from Fuji Bank and the grant OTKA-F014919. This work was done while the author was visiting the Institute for Advanced Study, Princeton, NJ 08540. E-mail address: tardos@cs.elte.hu.

† Department of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel, International Computer Science Institute, 1947 Center Street, Suite 600, Berkeley, CA 94704, U.S.A, and Department of Computer Science, University of California, Berkeley, CA 94720, U.S.A. E-mail address: zwick@math.tau.ac.il.

protocol	bits	rounds	first difference?
trivial	$n + \lceil \log n \rceil$	2	yes
logstar	$n + \log^* n$	$\log^* n$	yes
SIMPLE	$n + 2$	n	yes
LOGSTAR	$n + 2$	$\log^* n$	yes
HAM_3	$n + 2$	3	no
LOG(c)	$n + \lceil \log^{(c-1)} n \rceil + 1$	c	yes

Table 1. Protocols for the universal relation.

and Wigderson [RW92] used it to obtain an $\Omega(n)$ lower bound on the monotone depth of the perfect matching problem. Subsequently, many other papers dealt with or utilized this and similar relations between circuit depth and communication complexity. Some of them are Edmonds *et al.* [EIRS91], Karchmer, Raz and Wigderson [KRW91], Krause and Waack [KW91], Goldmann and Håstad [GH92], Håstad and Wigderson [HW93], Szegedy [Sze93], Goldmann [Gol94], Roychowdhury *et al.* [ROS94], and Grigni and Sipser [GS95]. Chin [Chi90] used the correspondence to obtain an upper bound on the depth of counting functions.

The communication problem described in the abstract is usually referred to as the communication problem of the *universal relation*. A solution to this communication problem gives a solution to the communication problem of *any* Boolean function.

A trivial upper bound on the communication complexity of the universal relation is $n + \lceil \log n \rceil$. Alice sends Bob her word and Bob replies with a $\lceil \log n \rceil$ bit index. In his thesis, Karchmer [Kar89] presents a slightly less trivial upper bound of $n + \log^* n$. Karchmer also presents a simple $n - 1$ lower bound on the communication complexity of the universal relation.

The thesis of Karchmer [Kar89] leaves a gap of $\log^* n$ between the upper and lower bounds on the communication complexity of the universal relation. This gap was closed by the unpublished $(n+2)$-bit n-round protocol of [RT96]. We present their protocol here. We then describe a way of reducing the number of rounds used by this protocol from n to $\log^* n$. We also describe, for every $c \geq 2$, a c-round protocol for the universal relation that exchanges at most $n + \lceil \log^{(c-1)} n \rceil + 1$ bits. These protocols always find the *first* difference between the input words. We also present a completely different $(n+2)$-bit 3-round protocol based on the Hamming error-correcting code. The difference found by this protocol is not necessarily the first difference between the inputs.

Improving the $n - 1$ lower bound of Karchmer, we show that any protocol for the universal relation must exchange, in the work case, at least $n + 1$ bits.

The old and new protocols for the universal relation are compared in Table 1. We think that the existence of a protocol for the universal relation that exchanges at most $n + 2$ bits in a *fixed* number of rounds is quite surprising. Note that our protocol that achieves this does not necessarily find the first difference between x and y. This is not a coincidence. We show that any $(n + O(1))$-bit protocol that always finds the first difference between x and y must be composed of at least $\log^* n - O(1)$ rounds. Furthermore, we show that each such protocol which is composed of only c rounds must exchange at least $n + \lceil \log^{(c-1)} n \rceil - 2$ bits.

Using the Karchmer-Wigderson correspondece of communication length and formula depth, our protocols yield a depth-4 balanced formula of size 2^{n+3} for the lookup function. Our protocols also imply the existence of *formula schemes* (see [MP77]) of depth $n + 2$, slightly simplifying and improving a result of McColl and Paterson [MP77].

See the surveys of Lengauer [Len90] and Lovász [Lov90], and the forthcoming book by Kushilevitz and Nisan [KN95], for excellent introductions to communication complexity.

2 Protocols based on the Hamming error correcting code

The protocols described in this section are based on the Hamming error-correcting code (see van Lint [vL91]). Similar coding ideas were employed by Lupanov [Lup73] and Gaskov [Gas78].

For every $x, y \in \{0,1\}^n$, we let $d(x,y)$ be the Hamming distance between x and y, i.e., the number of positions in which x and y differ. For every $x \in \{0,1\}^n$, we let

$$\begin{aligned} ball(x) &= \{y \in \{0,1\}^n \mid d(x,y) \leq 1\} , \\ sphere(x) &= \{y \in \{0,1\}^n \mid d(x,y) = 1\} \\ &= ball(x) - \{x\} . \end{aligned}$$

Let $n = 2^r - 1$, for some $r \geq 1$. The binary Hamming code of length n is a collection C_n of 2^{n-r} binary words

Alice:

send $sph(X_1)$
receive bit
if $bit = 0$ then
  receive Y_2
  if $X_2 \neq Y_2$ then
    $i \leftarrow k_1 + \mathit{diff}(X_2, Y_2)$
  else ($X_2 = Y_2$)
    $i \leftarrow ind(X_1)$
else ($bit = 1$)
  receive Y_1
  $i \leftarrow \mathit{diff}(X_1, Y_1)$

Bob:

receive $sph(X_1)$
if $sph(X_1) = sph(Y_1)$ then
  send 0
  send Y_2
else
  send 1
  send Y_1

Figure 1. Protocol HAM_2**.**

of length n such that the distance between any two words $x, y \in C_n$ is at least three. The Hamming code is therefore a single-error correcting code. Furthermore, the Hamming code is a *perfect* code, the 2^{n-r} balls of radius one, centered at the words of C_n define a partition of $\{0,1\}^n$.

Let $n = 2^r$ for some $r > 1$. From the Hamming code of length $n-1$, we can easily construct a collection $\hat{C}_n \subseteq \{0,1\}^n$, $|\hat{C}_n| = 2^{n-r}$ such that the 2^{n-r} spheres centered at the words of $\hat{C}_n$ define a partition of $\{0,1\}^n$. In other words, for every $x \in \{0,1\}^n$, there is a *unique* $c \in \hat{C}_n$ such that $d(x,c) = 1$. If C_{n-1} is the Hamming code of length $n-1$, we let $\hat{C}_n = \{0x, 1x \mid x \in C_{n-1}\}$ (to each codeword from C_{n-1} we affix a zero, and a one, and add the two words to $\hat{C}_n$). The required property of $\hat{C}_n$ follows easily from the fact that the Hamming code $\hat{C}_{n-1}$ is a perfect code.

The collection $\hat{C}_n$ can be defined directly as follows. Let M be an $n \times \log n$ matrix with all the different $\log n$ bit strings as rows (in some order). Then

$$\hat{C}_n = \{\, x \in \{0,1\}^n \mid xM = 0 \,\} .$$

We assign each word of $\hat{C}_n$ a unique $n-r$ bit label. Let $x \in \{0,1\}^n$. We let $sph(x)$ be the label of the unique $c \in \hat{C}_n$ such that $d(x,c) = 1$. We let $ind(x)$ be the position in which x and c differ.

We now describe several interesting protocols based on the partition of $\{0,1\}^n$ into disjoint spheres.

2.1 An $(n - \lfloor \log n \rfloor + 2)$-bit 2-round protocol using which Alice can find a difference

We start by describing a very simple $(n - \lfloor \log n \rfloor + 2)$-bit 2-round protocol, called HAM_2, using which Alice can find an index i such that $x_i \neq y_i$. Protocol HAM_2 is described in Figure 1. The words x and y, of length n, are broken into two blocks X_1, X_2 and Y_1, Y_2 of lengths k_1 and k_2 respectively. If $n = 2^r + s$, where $0 \le s < 2^r$, then $k_1 = 2^{r-1}$ and $k_2 = 2^{r-1} + s$. Note that $\log k_1 = \lfloor \log n \rfloor - 1$. If x and y are two distinct words of the same length, we let $\mathit{diff}(x,y)$ be the index of first position in which x and y differ.

Alice begins by sending $sph(X_1)$, the $k_1 - \log k_1$ bit label of the sphere to which X_1 belongs. Bob compares $sph(X_1)$ and $sph(Y_1)$. If $sph(X_1) = sph(Y_1)$, then Bob sends the bit 0, followed by Y_2. Alice now compares X_2 and Y_2. If $X_2 \neq Y_2$, then Alice clearly knows at least one position in which X_2 and Y_2 differ. She sets $i \leftarrow k_1 + \mathit{diff}(X_2, Y_2)$. The more interesting case is when Alice finds out that $X_2 = Y_2$. She then knows that $X_1 \neq Y_1$ while $sph(X_1) = sph(Y_1)$. This means that X_1 and Y_1 are two different words belonging to the same sphere. The Hamming distance between X_1 and Y_1 is exactly 2. The two positions in which X_1 and Y_1 differ are exactly $ind(X_1)$ and $ind(Y_1)$. Alice knows $ind(X_1)$ and she sets $i \leftarrow ind(X_1)$.

Suppose now that Bob finds out that $sph(X_1) \neq sph(Y_1)$. This clearly means that $X_1 \neq Y_1$. Bob sends the bit 1, followed by Y_1. Alice can now find the first position in which X_1 and Y_1 differ.

At the end of the protocol, i will always correspond to a position in which x and y differ. In fact, i will always be one of the first *three* positions in which x and y differ.

How many bits are exchanged? Alice sends $k_1 - \log k_1$ bits. Bob replies with either $k_1 + 1$ or $k_2 + 1$ bits. As $k_1 \le k_2$, the worst case is $k_1 + k_2 - \log k_1 + 1 = n - \log k_1 + 1 = n - \lfloor \log n \rfloor + 2$. We have thus established the following theorem:

Alice:	Bob:
send X_0	receive X_0
send $sph(X_1)$	receive $sph(X_1)$
receive bit_1	`if` $X_0 = Y_0$ `and` $sph(X_1) = sph(Y_1)$ `then`
`if` $bit_1 = 0$ `then`	send 0
receive Y_2	send Y_2
`if` $X_2 \neq Y_2$ `then`	receive bit_2
send 0	`if` $bit_2 = 0$ `then`
send $d_2 \leftarrow \mathit{diff}(X_2, Y_2)$	receive d_2
`else` $(X_2 = Y_2)$	`else` $(bit_2 = 1)$
send 1	receive d_1
send $d_1 \leftarrow ind(X_1)$	`else if` $X_0 \neq Y_0$ `then`
`else` $(bit_1 = 1)$	send 10
receive bit_3	send $d_0 \leftarrow \mathit{diff}(X_0, Y_0)$
`if` $bit_3 = 0$ `then`	`else` ($X_0 = Y_0$ but $sph(X_1) \neq sph(Y_1)$)
receive d_0	send 11
`else` $(bit_3 = 1)$	send Y_1
receive Y_1	receive d_1
send $d_1 \leftarrow \mathit{diff}(X_1, Y_1)$	

Figure 2. Protocol HAM$_3$**.**

Theorem 2.1 *Protocol* HAM$_2$ *is composed of* 2 *rounds in which at most* $n - \lfloor \log n \rfloor + 2$ *bits are exchanged. If* $x \neq y$*, then after running the protocol Alice knows a position in which* x *and* y *differ. Furthermore, this position is one of the first three positions in which* x *and* y *differ.*

It is possible to obtain a variant of HAM$_2$ in which at most $n - \lfloor \log n \rfloor + 1$ bits are exchanged, for every n that satisfies $n \geq 2^{\lfloor \log n \rfloor} + 2\lfloor \log n \rfloor$. Note that most values of n satisfy this condition.

2.2 An $(n+2)$-bit 3-round protocol for the universal relation

By supplementing HAM$_2$ with a third round in which Alice sends the $\lceil \log n \rceil$-bit index of the position she found to Bob, we get a three round protocol, HAM$'_3$, for the universal relation. The total number of bits exchanged by HAM$'_3$ is at most $(n - \lfloor \log n \rfloor + 2) + \lceil \log n \rceil \leq n + 3$.

A 3-round protocol, HAM$_3$, for the universal relation which exchanges at most $n + 2$ bits, one bit less than HAM$'_3$, is decribed in Figure 2. Protocol HAM$_3$ is also schematically described in Figure 3. Suppose that $n = 2^r + s$, where $0 \leq s < 2^r$. The words x and y, of length n, are broken this time into three blocks X_0, X_1, X_2 and Y_0, Y_1, Y_2 of lengths k_0, k_1 and k_2, respectively, where $k_0 = s$ and $k_1 = k_2 = 2^{r-1}$.

Alice begins by sending X_0 and $sph(X_1)$. Bob compares these blocks to Y_0 and $sph(Y_1)$.

If $X_0 = Y_0$ and $sph(X_1) = sph(Y_1)$ then Bob sends the bit 0 followed by Y_2. Alice now compares X_2 and Y_2. If $X_2 \neq Y_2$ then she sends the bit 0 followed by the $\log k_2 = r - 1$ bits of $\mathit{diff}(X_2, Y_2)$. If $X_2 = Y_2$ the Alice sends the bit 1 followed by the $\log k_1 = r - 1$ bits of $ind(X_1)$. Note that, as X_1 and Y_1 belong to the same sphere, these bits describe a position in which X_1 and Y_1 differ.

Suppose now that Bob finds out that $X_0 \neq Y_0$. He sends the pair 10 followed by the $\lceil \log k_0 \rceil$ bits of $\mathit{diff}(X_1, Y_1)$.

Finally, suppose that Bob finds out that $X_0 = Y_0$ but $sph(X_1) \neq sph(Y_1)$. He then sends the pair 11 followed by Y_1. Alice then responds with $\mathit{diff}(X_1, Y_1)$.

In any case, Alice and Bob agree on a position in which x and y agree. A closer look shows that this position must be one of the first three positions in which x and y differ.

How many bits are exchanged? If $X_0 = Y_0$ and $sph(X_1) = sph(Y_1)$ (the left branch in Figure 3), then the number of bits exchanged is $k_0 + (k_1 - \log k_1) + (1 + k_2) + (1 + \log k_1)$ (note that $k_1 = k_2$). This is exactly $k_0 + k_1 + k_2 + 2 = n + 2$. If $X_0 \neq Y_0$, then the number of bits exchanged is $k_0 + (k_1 - \log k_1) + 2 + \lceil \log k_0 \rceil \leq n + 2$. Finally, the number of bits exchanged in the remaining case is $k_0 + (k_1 - \log k_1) + 2 + k_2 + \log k_1$ which is again $n + 2$.

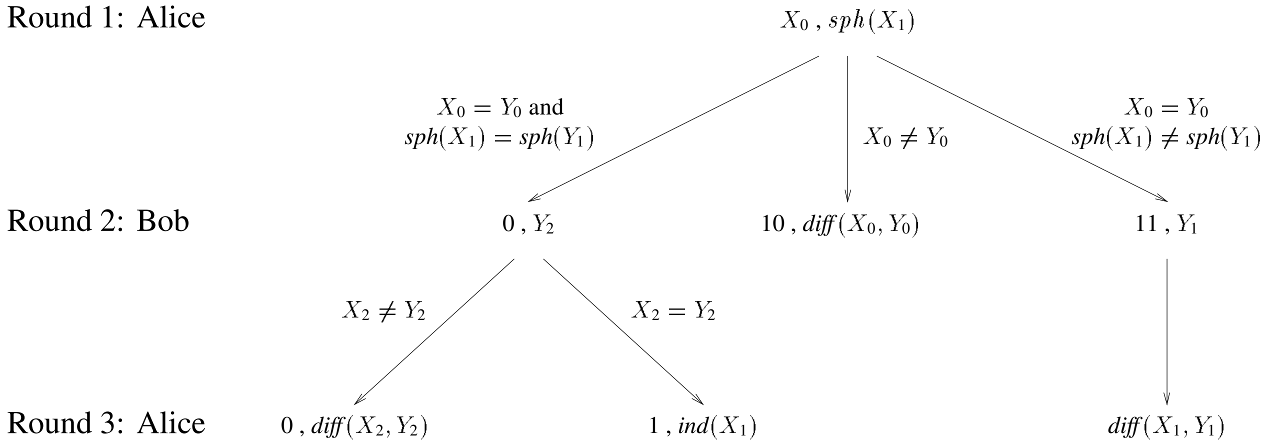

Figure 3. The communication pattern of protocol HAM$_3$.

Theorem 2.2 *Protocol* HAM$_3$ *is composed of* 3 *rounds in which at most* $n + 2$ *bits are exchanged. If* $x \neq y$*, then after running the protocol Alice and Bob agree on a position in which* x *and* y *differ. Furthermore, this position is one of the first three positions in which* x *and* y *differ.*

Protocol HAM$_3$ is the third protocol mentioned in the abstract. Note that in the third case of protocol HAM$_3$ (the case $X_0 = Y_0$ but $sph(X_1) \neq sph(Y_1)$), Bob does not have to send the last bit of Y_1, Alice can compute $diff(X_1, Y_1)$ without it. With this modification, the number of bits sent in the second round of the protocol is at most $k_1 + 1 = k_2 + 1 = 2^{r-1} + 1$, the only exception being the case $n = 7$. The number of bits sent in the third round is always at most $\log k_1 + 1 = \log k_2 + 1 = r$. With this modification, HAM$_3$ is therefore an *oblivious* 3-round $(n+2)$-bit protocol for the universal relation finding one of the first three differences. For the $n = 7$ case, a similar protocol is possible by partitioning the input into two blocks of sizes $k_1 = 4$ and $k_2 = 3$.

2.3 An $(n + 3)$-bit 4-round protocol for the strong universal relation

We now turn our attention to the communication problem in which Alice and Bob have to *decide* whether their inputs are equal, and agree on a position in which they differ, if they are not. A protocol solving this problem is said to be a protocol for the *strong universal relation*. It is easy to see that any protocol for the universal relation can be turned into a protocol for the strong universal relation by increasing the number of bits exchanged by at most two and increasing the number of rounds by at most one. At the end of protocol HAM$_3$, Bob knows whether $x = y$, so this specific protocol can be turned into a 4-round $(n + 3)$-bit protocol for the strong universal relation by adding a final round in which Bob sends one bit to Alice, telling her whether the two inputs are equal. We call the protocol so obtained HAM$_4$.

Theorem 2.3 *Protocol* HAM$_4$ *is a* 4*-round* $(n+3)$*-bit protocol for the strong universal relation.*

If the inputs of Alice and Bob differ, then HAM$_4$, as HAM$_3$, finds one of the first three differences between them. Protocol HAM$_4$, like HAM$_3$, can also be made oblivious.

2.4 An $(n - \lfloor \log n \rfloor + 4)$-bit 4-round protocol using which both Alice and Bob can find differences

In this subsection we describe a surprising $(n - \lfloor \log n \rfloor + 4)$-bit 4-round protocol using which Alice can find a position i such that $x_i \neq y_i$ and Bob can find a position j such that $x_j \neq y_j$. Note that such a protocol is not a porotocol for the universal relation as i and j are not necessarily the same.

We begin by describing an $(n - \lfloor \log n \rfloor + 5)$-bit 5-round protocol, HAM$_5$, for the job. Later we describe a slightly more complicated protocol that does the same with only 4 rounds of communication. Protocol HAM$_5$, invokes protocol HAM$_4$ for the strong universal relation. Protocol HAM$_4$ is composed of 4 rounds of communication. We assume this time that Bob starts the communication (otherwise an additional round would be required). We also assume that HAM$_4$ returns the index 0 if $x = y$. A description of HAM$_5$ is given in Figure 4.

Alice and Bob partition their words x and y into blocks X_1, X_2 and Y_1, Y_2, as they did before running HAM$_2$. Protocol HAM$_5$, as protocol HAM$_2$, starts with Alice sending

Alice:

send $sph(X_1)$
receive bit
`if` $bit = 0$ `then`
    $p \leftarrow \text{HAM}_4(X_2, Y_2)$
    `if` $p > 0$ `then`
        $i \leftarrow k_1 + p$
    `else` $(X_2 = Y_2)$
        $i \leftarrow ind(X_1)$
`else` $(bit = 1)$
    receive Y_1
    $i \leftarrow diff(X_1, Y_1)$
    send i

Bob:

receive $sph(X_1)$
`if` $sph(X_1) = sph(Y_1)$ `then`
    send 0
    $p \leftarrow \text{HAM}_4(X_2, Y_2)$
    `if` $p > 0$ `then`
        $j \leftarrow k_1 + p$
    `else` $(X_2 = Y_2)$
        $j \leftarrow ind(Y_1)$
`else`
    send 1
    send Y_1
    receive j

Figure 4. Protocol HAM$_5$**.**

$sph(X_1)$ to Bob. Bob compares $sph(X_1)$ and $sph(Y_1)$. If $sph(X_1) = sph(Y_1)$ then Bob sends the bit 0 and Alice and Bob run protocol HAM$_4$ on X_2 and Y_2. If $X_2 \neq Y_2$ then Alice and Bob agree on a position in which X_2 and Y_2 differ. If $X_2 = Y_2$, then as $x \neq y$, it must be the case that $X_1 \neq Y_1$. As X_1 and Y_1 belong to the same sphere, X_1 and Y_1 differ in exactly two positions, namely, in positions $ind(X_1)$ and $ind(Y_1)$. Alice knows $ind(X_1)$ and Bob knows $ind(Y_1)$. Finally, if $sph(X_1) \neq sph(Y_1)$, then Bob sends the bit 1 followed by the block Y_1. Alice finds the first difference between X_1 and Y_1 and sends it to Bob.

It is easy to verify that HAM$_5$ is indeed composed of at most 5 rounds and that the maximal number of bits exchanged is bounded by either $(k_1 - \log k_1) + 1 + (k_2 + 3)$ or $(k_1 - \log k_1) + 1 + 2\log k_1$. Both expressions are at most $n - \lfloor \log n \rfloor + 5$. We have thus established the following theorem:

Theorem 2.4 *Protocol* HAM$_5$ *is composed of at most* 5 *rounds in which at most* $n - \lfloor \log n \rfloor + 5$ *bits are exchanged. If* $x \neq y$*, then after running the protocol Alice knows an index* i *such that* $x_i \neq y_i$ *and Bob knows an index* j *such that* $x_j \neq y_j$*. Furthermore,* $i = j$ *unless* x *and* y *differ in exactly two positions.*

Protocol HAM$_5$ invokes HAM$_4$ on X_2 and Y_2. Protocol HAM$_4$ divides each of the blocks X_2 and Y_2 into two sub-blocks. By initially dividing x of y into three blocks X_1, X_2, X_3 and Y_1, Y_2, Y_3, of lengths k_1, k_2 and k_3, we can obtain a 4-round protocol, HAM$'_4$, which also exchanges at most $n - \lfloor \log n \rfloor + 4$ bits using which both Alice and Bob can find indices of positions in which x and y differ.

Protocol HAM$'_4$ is described in Figure 5. For sake of conciseness, we have not separated the roles of Alice and Bob in the protocol. It is easy to check that HAM$'_4$ is composed of at most 4 rounds of communication. If $n = 2^r + s$, where $0 \leq s < 2^r$, we let $k_1 = 2^{r-1}$, $k_2 = 2^{r-2}$ and $k_3 = 2^{r-2} + s$. It is easy to see that the number of bits exchanged by HAM$'_4$ is either $n - \log k_1 + 3$, or $n - \log k_1 - \log k_2 + \lceil \log k_3 \rceil + 2$, or at most $n - k_3 + 2$. All these expressions are at most $n - \lfloor \log n \rfloor + 4$. We have thus obtained:

Theorem 2.5 *Protocol* HAM$'_4$ *is composed of at most* 4 *rounds in which at most* $n - \lfloor \log n \rfloor + 4$ *bits are exchanged. If* $x \neq y$*, then after running the protocol Alice knows an index* i *such that* $x_i \neq y_i$ *and Bob knows an index* j *such that* $x_j \neq y_j$*. Furthermore,* $i = j$ *unless* x *and* y *differ in exactly two positions.*

For most values of n, it is possible to choose the values of k_1,k_2 and k_3 a little bit better so that the total number of bits exchanged by HAM$'_4$ is at most $n - \lfloor \log n \rfloor + 3$.

3 Protocols for finding the first difference

In this section we describe two $(n+2)$-bit protocols for finding the first difference between the inputs. The first protocol, described in Subsection 3.1, is extremely simple. It uses, however, a very large number of rounds. In Subsection 3.2 we reduce the number of rounds used from n to $\log^* n + 2$, without increasing the total number of bits exchanged. We show in the next section that the number of rounds cannot be reduced further without increasing the number of bits exchanged. We end the section with a family of protocols that presents an essentially optimal tradeoff between the number of rounds and the number of bits exchanged.

```
Alice sends sph(X_1)
Alice sends sph(X_2)
if sph(X_1) = sph(Y_1) and sph(X_2) = sph(Y_2)
then
    Bob sends 0
    Bob sends Y_3
    if X_3 = Y_3 then
        Alice sends 0
        Alice sends ind(X_2)
        if ind(X_2) = ind(Y_2) then
            Bob sends 0
            Alice sets i ← ind(X_1)
            Bob sets j ← ind(Y_1)
        else (ind(X_2) ≠ ind(Y_2))
            Bob sends 1
            Alice and Bob set i ← j ← k_1 + ind(X_2)
    else (X_3 ≠ Y_3)
        Alice sends 1
        Alice sends diff(X_3, Y_3)
        Alice and Bob set
            i ← j ← k_1 + k_2 + diff(X_3, Y_3)
else if sph(X_1) ≠ sph(Y_1) then
    Bob sends 10
    Alice sends ind(X_1)
    Bob sends diff(X_1, Y_1)
    Alice and Bob set i ← j ← diff(X_1, Y_1)
else (sph(X_1) = sph(Y_1) , sph(X_2) ≠ sph(Y_2))
    Bob sends 11
    Alice sends ind(X_2)
    Bob sends diff(X_2, Y_2)
    Alice and Bob set i ← j ← k_1 + diff(X_2, Y_2)
```

Figure 5. Protocol HAM'_4**.**

3.1 A simple $(n+2)$-bit protocol finding the first difference

In this subsection we describe an elementary $(n+2)$-bit protocol for the universal relation that always finds the first difference. This protocol is due to Rudich and Tardos [RT96]. It is included here as it has not been published yet.

Protocol SIMPLE is the first protocol mentioned in the abstract. It is described in Figure 6. Although the pseudo-code of SIMPLE is not as concise as that of HAM_3, protocol SIMPLE is conceptually simpler.

Protocol SIMPLE is composed of two phases. Excatly n bits are exchanged in the first phase and excatly 2 bits in the second. Alice and Bob trasmit their bits interchangingly. We let $a_1, a_3, \ldots$ be the bits sent by Alice and $b_1, b_3, \ldots$ be the bits sent by Bob. Alice begins by sending $a_1 \leftarrow x_1$. The subsequent bits sent by Alice are determined by the following rules:

(a) *No difference found yet* If $x_2x_4 \ldots x_i = b_2b_4 \ldots b_i$ then Alice sends $a_{i+1} \leftarrow x_{i+1}$.

(b) *Difference just found* If $x_2x_4 \ldots x_{i-2} = b_2b_4 \ldots b_{i-2}$ but $x_i \neq b_i$ then Alice sends $a_{i+1} \leftarrow 1$.

(c) *Difference found earlier* If $x_2x_4 \ldots x_{i-2} \neq b_2b_4 \ldots b_{i-2}$ then Alice sends $a_{i+1} \leftarrow 0$.

Bob follows an analogous set of rules. In the pseudo-code given in Figure 6, the variable $lock_A$ has the value 0 (Alice is *'unlocked'*) as long as no difference is discovered by Alice. When Alice discovers a difference, she sets $lock_A \leftarrow 1$ and becomes *'locked'*. She then transmits a 1. All the subsequent bits sent by Alice, if there are any, will be 0's.

We let $last1(c_1, c_2, \ldots)$ be the index of the last 1 in the sequence $c_1, c_2, \ldots$. Note that if $i = 2last1(a_1, a_3, \ldots) - 1$, i.e., $a_i = 1$ but $a_j = 0$ for $j > i$, and if Alice is locked, then Alice discovered her first difference, i.e., became locked, in position $i - 1$. Similarly, if $j = 2last1(b_2, b_4, \ldots)$ and Bob is locked, then Bob became locked in position $j - 1$. The streams $a_1, a_3, \ldots$ and $b_2, b_4, \ldots$ are known to both Alice and Bob. If Alice, for example, is told that Bob became locked, she can therefore identify the position in which he became locked.

At the end of the first stage, there are three candidates for the position of the first difference between x and y. This position can either be the position in which Alice locked, if she did, the position in which Bob locked, if he did, or, the last position.

In the second stage of the protocol Alice and Bob inform each other whether they became locked. It is easy to see that if at least one of Alice and Bob did lock, then the first difference between x and y corresponds to the position in which the first of them locked. If none of them locked, then $x_1x_2 \ldots x_{n-1} = y_1y_2 \ldots y_{n-1}$. As Alice and Bob are promised that $x \neq y$, they can deduce that $x_n \neq y_n$.

We have thus established the following theorem:

Theorem 3.1 *Protocol* SIMPLE *finds the first difference between x and y by exchanging exactly $n+2$ bits.*

Any protocol PROT that finds the *first* difference between x and y can be easily transfored into a protocol PROT^* for the strong universal relation (see Subsection 2.3 for a definition).

Alice:
$lock_A \leftarrow 0$
send $a_1 \leftarrow x_1$
`for` $i \leftarrow 2$ `to` $n-1$ `by` 2 `do`
{
  receive b_i
  `if` $lock_A = 1$ `then`
    send $a_{i+1} \leftarrow 0$
  `else` $(lock_A = 0)$
    `if` $x_i = b_i$ `then`
      send $a_{i+1} \leftarrow x_{i+1}$
    `else` $(x_i \neq b_i)$
      $lock_A \leftarrow 1$
      send $a_{i+1} \leftarrow 1$
}
if n is even then receive b_n
send $lock_A$
receive $lock_B$
$$i_A \leftarrow \begin{cases} 2 \cdot last1(a_1, a_3, \ldots) - 2 & \text{if } lock_A = 1 \\ n & \text{otherwise} \end{cases}$$
$$i_B \leftarrow \begin{cases} 2 \cdot last1(b_2, b_4, \ldots) - 1 & \text{if } lock_B = 1 \\ n & \text{otherwise} \end{cases}$$
$i \leftarrow \min\{i_A, i_B\}$

Bob:
$lock_B \leftarrow 0$
`for` $i \leftarrow 1$ `to` $n-1$ `by` 2 `do`
{
  receive a_i
  `if` $lock_B = 1$ `then`
    send $b_{i+1} \leftarrow 0$
  `else` $(lock_B = 0)$
    `if` $y_i = a_i$ `then`
      send $b_{i+1} \leftarrow y_{i+1}$
    `else` $(y_i \neq a_i)$
      $lock_B \leftarrow 1$
      send $b_{i+1} \leftarrow 1$
}
if n is odd then receive a_n
receive $lock_A$
send $lock_B$
$$i_A \leftarrow \begin{cases} 2 \cdot last1(a_1, a_3, \ldots) - 2 & \text{if } lock_A = 1 \\ n & \text{otherwise} \end{cases}$$
$$i_B \leftarrow \begin{cases} 2 \cdot last1(b_2, b_4, \ldots) - 1 & \text{if } lock_B = 1 \\ n & \text{otherwise} \end{cases}$$
$i \leftarrow \min\{i_A, i_B\}$

Figure 6. Protocol SIMPLE**.**

Protocol PROT* simply runs PROT on the inputs $x0$ and $y1$. If the first difference occurs in position $n+1$, Alice and Bob infer that $x = y$. We thus obtain:

Theorem 3.2 *Protocol* SIMPLE* *checks whether $x = y$, and finds the first difference between them, if they are not equal, by exchanging exactly $n + 3$ bits.*

As described, protocol SIMPLE requires $n + 1$ rounds, if n is odd, or $n + 2$ rounds, if n is even. It is possible to reduce the number of rounds to n as follows. Assume, for concreteness, that n is odd. The first stage then ends with Bob receiving the bit a_n from Alice. Bob, however, does not compare a_n with y_n and never locks on the n-th position. Bob may therefore send $lock_B$ together with b_{n-1}, before receiving a_n. Alice may then send $lock_A$ together with a_n. It is interesting to note that the last two bits of Alice have to distinguish between only three possibilities: (i) Alice became locked before position $n-1$; (ii) Alice became locked at position $n-1$; (iii) Alice did not become locked. If Alice is not locked while sending a_n, then a_n may be arbitrarily set to 0. With this modification Alice and Bob never examine x_n and y_n.

3.2 Finding the first difference in a limited number of rounds.

The protocol described in the previous subsection was very simple. It uses, however, n rounds of communication. In this subsection we describe more complicated protocols that find the first difference in a limited number of rounds. Surprisingly, using the slack in the last round of SIMPLE (two bits are used to distinguish three possibilities) one can reduce the number of rounds to $\log^* n + 2$ without increasing the length. The result is a protocol, named LOGSTAR, which is the second protocol mentioned in the abstract.

The number of rounds cannot be reduced below $\log^* n$ without increasing the number of bits sent. For every $c \geq 2$, we present a c-round protocol, LOG(c), for the universal relation that exchanges at most $n + \lceil \log^{(c-1)} n \rceil + 1$ bits. LOG(c) always finds the first difference between the inputs.

The length of LOG(c) is within three of the lower bound for this problem presented in the next section.

We start with a description, for any $s \geq 2$, of a protocol LOG$_s$. The protocol LOGSTAR and the protocols LOG(c) are obtained by chooing appropriate values of the parameter s.

Define a sequence $a_1 = 2^s - 2$ and $a_{i+1} = 2^{a_i} - 1$. For any $n \geq 1$, protocol LOG$_s$ starts by splitting x and y into $k + 1$ blocks as follows. Let k be the smallest number for which $\sum_{i=1}^{k} a_i \geq n - 1$. We break x into blocks $X_1, \ldots, X_k, X_{k+1}$, where X_i, for $2 \leq i \leq k$ is of length a_{k+1-i}, X_{k+1} is composed of a single bit, and X_1 is of length at most a_k. The word y is broken into blocks $Y_1, \ldots, Y_k, Y_{k+1}$ of corresponding lengths. Note that the lengths of the blocks $X_2, \ldots, X_{k+1}$ and $Y_2, \ldots, Y_k, Y_{k+1}$ decrease dramatically (the blocks X_1 and Y_1 are leftover blocks and may be of any size in the range 1 to a_k).

Protocol LOG$_s$ is similar to the protocol SIMPLE. The players alternate this time, however, in sending blocks rather than just bits.

The first phase of the protocol consists of k rounds. The number of bits sent in the i-th round is $|X_i| = |Y_i| = a_{k+1-i}$. As in SIMPLE, both Alice and Bob begin the protocol by being 'unlocked'. Alice transmits in the odd numbered rounds and Bob in the even numbered ones. Let A_i be the block sent by Alice in the i-th round, and let B_i be the block sent by Bob in the i-th round. Alice starts by sending $A_1 \leftarrow X_1$. The block sent by Alice in the $(i+1)$-st round of the first phase, where $i \geq 2$, is determined by the following rules:

(a) *No difference found yet* If $X_2X_4 \ldots X_i = B_2B_4 \ldots B_i$ then Alice sends $A_{i+1} \leftarrow X_{i+1}$.

(b) *Difference just found* If $X_2X_4 \ldots X_{i-2} = B_2B_4 \ldots B_{i-2}$ but $X_i \neq B_i$ then, Alice becomes locked, and instead of sending X_{i+1}, she sends $t \leftarrow \mathit{diff}(X_i, B_i)$, the position of the first difference between X_i and B_i. Here $1 \leq t \leq |X_i| \leq a_{k+1-i} < 2^{a_{k-i}} = 2^{|X_{i+1}|}$, thus t can be sent as an a_{k-i}-bit block. Note that $t \neq 0$, so the block sent by Alice when she becomes locked in a *non-zero* block.

(c) *Difference found earlier* If $X_2X_4 \ldots X_{i-2} \neq B_2B_4 \ldots B_{i-2}$ (Alice is locked), then Alice sends an all-zero block of the appropriate length.

The blocks sent by Bob are determined by analogous rules.

It is easy to see that the last non-zero block sent by Alice/Bob marks the position in which she/he became locked, if they did. The block size a_{i+1} is defined to be $2^{a_i} - 1$, and not 2^{a_i}, to ensure this property.

After the first phase, at most two positions in the blocks $X_1 \ldots X_{k-1}$ and $Y_1 \ldots Y_{k-1}$ are candidates for being the first difference between x and y. If A_i is the last non-zero block sent by Alice, then the position in X_{i-1} whose index is coded in A_i is the first of these candidates. If B_j is the last non-zero block sent by Bob, then the position in X_{j-1} whose index is coded in B_j is the second candidate. This is so, since if the first difference between x and y is the t-th bit of X_i, where $i < k$, then $A_{i+1} = t$ or $B_{i+1} = t$, by case (b), and all subsequent blocks sent by the player that sent A_{i+1} or B_{i+1} are all-zero blocks. Unfortunately all positions in the next to last block X_k, as well the last bit, are also candidates.

The second phase of the protocol consists again of two rounds, as in protocol SIMPLE. The first of these rounds coincides with the last round of the first phase. Let us assume, for concretness, that Alice was the last to transmit in the first phase. Otherwise, the roles of Alice and Bob are reversed. First, Alice sends a 0 if she is unlocked, and a 1 otherwise. Next, Bob sends a bits to describe one of the following 2^a possibilities:

(1) Bob sends 0^a to say that he is locked.

(2) Bob sends the binary representation of t, where $1 \leq t \leq |X_k| \leq a_1 = 2^a - 2$ to say that he is unlocked but $Y_k \neq A_k$ and t is the position of the first difference between these blocks.

(3) Bob sends 1^a to say that he is unlocked and $Y_k = A_k$.

It is easy to see that after the second round, both players can deduce the position of the first difference.

Protocol LOG$_s$ finds the first difference in $k + 1$ oblivious rounds of communication in which exactly $n + s$ bits are exchanged (k is defined above).

First we take $s = 2$ and let LOGSTAR = LOG$_2$. In this case $a_i > \exp^{(i-1)}(1)$ where $\exp^{(i)}$ is the exponentiation function 2^x iterated i times. Thus $n > a_{k-1} > \exp^{(k-2)}(1)$ and therefore $\log^* n \geq k - 1$. Thus we have

Theorem 3.3 LOGSTAR *is an oblivious protocol for the universal relation that finds the first position of difference by exchanging* $n + 2$ *bits in at most* $\log^* n + 2$ *rounds.*

In order to have fewer than $\log^* n$ rounds, we increase the parameter s and thus the length of the protocol. For an arbitrary integer parameter $c \geq 2$ we define LOG(c) to be LOG_s for the smallest $s \geq 2$ such that it has at most c rounds. If $s > 2$ we have $a_i > \exp^{(i)}(s - 1)$ thus

Theorem 3.4 *If* $\log^{(c-1)} n > 1$ *then* LOG(c) *is an oblivious protocol for the universal relation that finds the first position of difference by exchanging at most* $n + \lceil \log^{(c-1)} n \rceil + 1$ *bits in* c *rounds.*

4 Lower bounds

In this section we present some simple lower bounds that show that the protocols obtained in the previous sections are only a few bits away from being optimal.

We start with an $n + 1$ lower bound for the length of any protocol for the universal relation, for $n > 2$. This is a slight improvement over the $n - 1$ lower bound of Karchmer [Kar89] and comes within 1 of the upper bound we presented. For $n = 1$ there is no need for communication. For $n = 2$, Alice and Bob simply need to exchange the first bits of their inputs. For $3 \le n \le 6$ the lower bound of $n + 1$ can be achieved by a simple protocol, while for $n \ge 7$ we do not know whether optimal protocols for the universal relation use $n + 1$ or $n + 2$ bits.

Before we go ahead and prove the lower bounds, we call attention to a subtle point. A protocol for the universal relation has to work only for pairs of inputs (x, y), where $x \ne y$. All the protocols for the universal relation that we presented in this paper work even if $x = y$. Alice and Bob always agree on the same index i. If $x = y$, then this index is of course an index of a position in which x and y agree. It is not difficult to see that any protocol for the universal relation can be modified, if necessary, to have this property, without increasing the number of bits exchanged or the number of rounds used. All the protocols we consider in this section are therefore assumed to be of this form.

Theorem 4.1 *Any protocol for the n-bit universal relation uses in the worst case at least $n + 1$ bits, for $n > 2$.*

Proof: Let us consider a protocol P for the universal relation. As discussed above, we allow Alice and Bob to have an arbitrary pair (x, y) of n-bit inputs, including the case $x = y$. For any specific final transcript T of the conversation the set of pairs (x, y) resulting in T must be of the form $A_T \times B_T$ where $A_T, B_T \subset \{0, 1\}^n$.

First we claim that for any final transcript T we have

(i) $|A_T \cap B_T| \le 2$,

(ii) if $|A_T \cap B_T| = 2$ then $|A_T| = |B_T| = 2$, and

(iii) if $|A_T \cap B_T| = 1$ then either $|A_T| = 1$ or $|B_T| = 1$.

Indeed, any set $A_T \times B_T$ violating the above conditions would have three different n bit strings x, y, and z with $x, y \in A_T$ and $y, z \in B_T$. But then there is no consistent way Alice and Bob can find the position in which the input pairs (x, y), (x, z) and (y, z) differ. Note that each of these pairs result in the same transcript T.

Now suppose the protocol P has length n. We may suppose, without loss of generality, that each full transcript has length n. Let us consider a partial transcript T of length $n - 1$. It determines the set $A_T \times B_T$ of input pairs resulting in this partial transcript. Note that this set can be partitioned into two sets satisfying (i)–(iii) above, as T has two possible extensions to a final transcript. This observation is enough to verify the following claim.

For any partial transcript T of length $n - 1$ we have

(iv) $|A_T \cap B_T| \le 2$ and

(v) if $|A_T \cap B_T| = 2$ then $|A_T| = 2$ or $|B_T| = 2$.

The sets $A_T \times B_T$ corresponding to the 2^{n-1} partial transcripts T of length $n - 1$ partition the set $\{0, 1\}^n \times \{0, 1\}^n$ of all inputs. We see from (iv) that at most two of the 2^n diagonal elements can be in one class of the partition. By counting we get that each class has to contain exactly two diagonal elements. By (v), we get that the size of each set $A_T \times B_T$ is at most 2^{n+1}. Counting gives, now, that the size of each such set is exactly 2^{n+1} and thus $A_T = \{0, 1\}^n$ and $|B_T| = 2$ or $B_T = \{0, 1\}^n$ and $|A_T| = 2$. It is clear, though, that, if $n > 2$, and from such a state, a position in which the two inputs differ cannot be found by exchanging only a single additional bit.

The contradiction proves the theorem. □

It is easy to see that any protocol in which Alice finds an index i such that $x_i \ne y_i$ when such an index exists, must exchange in the worst case, at least $n - \lceil \log n \rceil + 1$ bits. Indeed, after finding such an index Alice can send it to Bob to get a protocol for the universal relation, which must exchange at least $n + 1$ bits. Our $(n - \lfloor \log n \rfloor + 2)$-bit protocol HAM_2 comes within two of this bound. The modified protocol mentioned after Theorem 2.1 comes within one of this bound, for most values of n.

It is interesting to note that HAM_2 does not necessarily find the first difference. This is not a coincidence. Any protocol after which Alice knows the first position of difference if one exists, must exchange, in the worst case, at least $n - 1$ bits. Indeed, after such a protocol Alice knows which of the two inputs is lexicographically first, and can send this information to Bob. But deciding the order of two non-equal inputs requires at least n bits in the worst case. We remark that the $n - 1$ lower bound is achieved by the protocol in which Bob sends all of his input but the last bit.

Finally we prove a lower bound for protocols for the universal relation finding the first position of difference in a limited number of rounds.

A protocol for the universal relation is said to be *symmetric*, if it for any pair of inputs x and y, where $x \ne y$, the index found by the protocol when Alice receives x and Bob receives y is equal to the index found when Alice receives y

and Bob receives x. Clearly, every protocol for the universal function that finds the first difference is a symmetric protocol.

Theorem 4.2 *Let P be a symmetric protocol for the universal relation on n-bit strings. If P consists of c rounds of communication, then the worst case length of P is at least $n + \lceil \log^{(c-1)} n \rceil - 2$.*

Note that for any number c the length of the c round protocol LOG(c) comes within three of this lower bound.

We start with a folklore result on bipartite graph covers.

Lemma 4.3 *Suppose that the edges of the complete graph K_n are colored, using an arbitrary number of colors, in such a way that the subgraph determined by any one color is bipartite. Then, there is a vertex with at least $\log n$ edges of different colors incident to it.*

Proof: Let X be the set of colors, and for any color $c \in X$, let V_c be the set of vertices incident to an edge of color c. Let $f_c : V_c \to \{0,1\}$ be a good coloring of the subgraph determined by the edges of color c. For a vertex v let

$$G_v = \{\, g : X \to \{0,1\} \mid \begin{array}{l} g(c) = f_c(v) \text{ for every} \\ c \in X \text{ such that } v \in V_c \end{array} \,\} .$$

Clearly, if v has k edges of different color incident to it, then G_v contains a 2^{-k} fraction of all the functions $g : X \to \{0,1\}$. Suppose now that $v \neq w$ and that the edge (v, w) is colored by c. If $g \in G_v \cap G_w$, then $g(c) = f_c(v) \neq f_c(w) = g(c)$. This contradiction shows that the sets G_v are pairwise disjoint. Thus one of them has a relative size at most $1/n$ and thus the corresponding vertex has $k \geq \log n$ adjecent colors. □

Lemma 4.4 *Suppose that the edges of the complete graph K_n are colored, using an arbitrary number of colors, in such a way that the subgraph determined by any one color is bipartite. Suppose that Alice receives a vertex v and that Bob receives a vertex w of this graph and that their goal is to find the color of the edge (v, w), if $v \neq w$. If P is a deterministic c-round m-bit protocol for solving this problem, and* $\log^{(c-1)} n > 1$, *then*

$$\log n < m - \lfloor \log^{(c-1)} m \rfloor + 2 .$$

Proof: We prove by induction on c the stronger inequality

$$n \leq 2^{m - \lfloor \log^{(c-1)} m \rfloor + 2} - 2 .$$

For the base case $c = 1$, we have $n \leq 2$, since one round of communication is as effective as none.

For the inductive step suppose the protocol P has $c + 1$ rounds. Suppose Alice starts the communication. Based on her input vertex v, she sends a string x_v to Bob. The length of this string may depend on v. We distinguish vertices with short initial message from vertices with long initial message by defining

$$\begin{aligned} S &= \{\, v \mid |x_v| \leq m - t \,\} , \\ L &= \{\, v \mid |x_v| > m - t \,\} , \end{aligned}$$

where $t = \lfloor \log m \rfloor$.

Let us consider the vertices in L first. Clearly, for any such vertex Alice is to receive at most $t - 1$ bits from Bob, thus she finally decides on one of 2^{t-1} possible colors. Thus, no vertex in L is adjacent to more than 2^{t-1} differently colored edges, so by Lemma 4.3 we have $|L| \leq 2^{2^{t-1}} \leq 2^{m/2}$.

Now we turn to S. For an $m-t$ bit string x let $S_x = \{v \in S \mid$ x is a prefix of the conversation between Alice and Bob when they both get $v\,\}$. These sets clearly partition S and it is also clear that if Alice and Bob get two different vertices from S_x then their conversation is also a prefix of x. Thus they find the color of the connecting edge in the graph spanned by S_x in the remaining part of the protocol. In this part they use at most t bits, and since the first round of communication ended within x, they actually use at most c rounds. Thus we can bound the size of S_x by the inductive hypothesis: $|S_x| \leq 2^{t - \lfloor \log^{(c-1)} t \rfloor + 2} - 2$. Summing over all possible x we get

$$\begin{aligned} |S| &\leq 2^{m-t}(2^{t - \lfloor \log^{(c-1)} t \rfloor + 2} - 2) \\ &= 2^{m - \lfloor \log^{(c)} m \rfloor + 2} - 2^{m-t+1} . \end{aligned}$$

To finish the proof we only have to note that

$$\begin{aligned} n &= |S| + |L| \leq 2^{m - \lfloor \log^{(c)} m \rfloor + 2} - 2^{m-t+1} + 2^{m/2} \\ &\leq 2^{m - \lfloor \log^{(c)} m \rfloor + 2} - 2 . \end{aligned}$$

□

Proof of Theorem 4.2: We may suppose $\log^{(c-1)} n > 1$, since otherwise Theorem 4.1 implies our result. Consider the the complete graph whose vertices are the n-bit strings and color the edge between x and y with the position the protocol P finds when applied to x and y. This is well defined as P is symmetric. Clearly, each monochromatic subgraph is bipartite thus Lemma 4.4 is applicable and implies the theorem. □

5 Formulae for the lookup function

The lookup function is a function of $2^n + n$ inputs defined as follows: $L_n(x_1, \ldots, x_n, y_0, \ldots, y_{2^n-1}) = y_{x_1 \ldots x_n}$. By the seminal result of Karchmer and Wigderson [KW90], any protocol for the strong universal relation yields a fanin-2 Boolean formula for the lookup function. The depth of the formula is the maximal number of bits exchanged by the protocol. (Note, however, that not every formula for the lookup function is obtained from such a protocol, as any formula that corresponds to a protocol reads every y input exactly once.)

Protocol SIMPLE* yields, therefore, a depth $n + 3$ fan-in 2 formula of size at most 2^{n+3} for comuting L_n. This formula can be made to consist of alternating levels of AND and OR gates.

Protocol HAM$_4$ yields a depth 4 *unbounded* fan-in formula of size at most 2^{n+3} for L_n. As HAM$_4$ can be made oblivious, the gates at each level of this formula can be made to have the same fan-in. The gates at the bottom level all have fan-in 2.

6 Concluding remarks and open problems

We presented three protocols, HAM$_3$, SIMPLE and LOGSTAR, for the universal relation. Each of these three protocols exchanges at most $n + 2$ bits. Although our lower bound is only $n + 1$, we conjecture that the upper bound presented by these protocols is tight.

Conjecture 6.1 *For large enough n, any protocol for the n-bit universal relation must exchange, in the worst case, at least $n + 2$ bits.*

Our next conjecture is more subtle and a bit harder to state. The protocols SIMPLE and LOGSTAR both find the first position of difference by exchanging at most $n+2$ bits. The advantage of LOGSTAR is its small number of rounds. Notice, however, that SIMPLE has advantages too. At the end of SIMPLE, one of the players knows whether the playres received the same input. Another advantage is that if the playes received different inputs, then the transcript of the conversation determines *how* the inputs differ at the agreed upon position, which player has a 1 there. It is easy to see that these two statements are equivalent for any protocol for the universal relation and are also equivalent to the statement that each invalid (i.e., equal) pair of input results in a different transcript of communication. We call a protocol for the universal relation *robust* if it satisfies any of the three equivalent conditions above. Note that both SIMPLE and HAM$_3$ are robust protocols for the universal relation. The following conjecture asserts that the high number of rounds in SIMPLE cannot be reduced significantly without loosing one of robustness, being oblivious, or the property of always finding the first difference. We remark, that for $n > 5$, one can modify SIMPLE slightly to reduce the number of rounds to $n - 4$ without losing either good property.

Conjecture 6.2 *Any robust oblivious protocol for the n-bit universal relation that exchanges at most $n + 2$ bits and always finds the first difference must have at least $n - O(1)$ rounds of communication.*

Acknowledgment

We would like to thank Noga Alon for some helpful discussions and for making the cooperation between the two co-authors possible. We would also like to thank Steven Rudich for allowing us to include a description of the protocol from [RT96].

References

[Chi90] A. Chin. On the depth complexity of the counting functions. *Information Processing Letters*, 35:325–328, 1990.

[Dun88] P.E. Dunne. *The complexity of Boolean networks*. Academic Press, 1988.

[EIRS91] J. Edmonds, R. Impagliazzo, S. Rudich, and J. Sgall. Communication complexity towards lower bounds on circuit depth. In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science, San Juan, Puerto Rico*, pages 249–257, 1991.

[Gas78] S.B. Gaskov. The depth of Boolean functions. *Problemy Kibern.*, 34:265–268, 1978. (In Russian).

[GH92] M. Goldmann and J. Håstad. A simple lower bound for monotone clique using a communication game. *Information Processing Letters*, 41:221–226, 1992.

[Gol94] M. Goldmann. Communication complexity and lower bounds for threshold circuits. In Vwani Roychowdhury, Kai-Yeung Siu, and Alon Orlitsky, editors, *Theoretical Advances in Neural Computation and Learning*. Kluwer, 1994.

[GS95] M. Grigni and M. Sipser. Monotone separation of logarithmic space from logarithmic depth. *Journal of Computer and System Sciences*, 50, 1995.

[HW93] J. Håstad and A. Wigderson. Composition of the universal relation. In J.-Y. Cai, editor, *Advances in Computational Complexity Theory*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pages 119–134. American Mathematical Society, 1993.

[Kar89] M. Karchmer. *Communication Complexity: A New Approach to Circuit Depth*. PhD thesis, The MIT Press, 1989.

[KN95] E. Kushilevitz and N. Nisan. *Communication Complexity*. draft, 1995.

[KRW91] M. Karchmer, R. Raz, and A. Wigderson. On proving super-logarithmic depth lower bounds via the direct sum in communication complexity. In *Proceedings of the 6th Annual Structure in Complexity Theory Conference*, pages 299–304, 1991.

[KW90] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM Journal on Discrete Mathematics*, 3:255–265, 1990.

[KW91] M. Krause and S. Waack. Variation ranks of communication matrices and lower bounds for depth two circuits having symmetric gates with unbounded fan-in. In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science, San Juan, Puerto Rico*, pages 777–782, 1991.

[Len90] T. Lengauer. VLSI theory. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume A, Algorithms and Complexity*, chapter 16, pages 835–868. Elsevier and The MIT Press, 1990.

[Lov90] L. Lovász. Communication complexity: A survey. In B.H. Korte, editor, *Paths, Flows and VLSI Layout*. Springer Verlag, 1990.

[Lup73] O.B. Lupanov. Complexity of the universal parallel-series network of depth 3. *Trudy Matem. Inst. Steklov*, 133:127–131, 1973. (In Russian).

[MP77] W.F. McColl and M.S. Paterson. The depth of all Boolean functions. *SIAM Journal on Computing*, 6:373–380, 1977.

[PM71] F.P. Preparata and D.E. Muller. On the delay required to realize Boolean functions. *IEEE Transactions on Computers*, C-20:459–461, 1971.

[ROS94] V.P. Roychowdhury, A. Orlitsky, and K.Y. Siu. Lower bounds on threshold and related circuits via communication complexity. *IEEE Transactions on Information Theory*, 40:467–474, 1994.

[RT96] S. Rudich and G. Tardos. Private communication.

[RW92] R. Raz and A. Wigderson. Monotone circuits for matching require linear depth. *Journal of the ACM*, 39(3):736–744, July 1992.

[Spi71] P.M. Spira. On the time necessary to compute switching functions. *IEEE Transactions on Computers*, C-20:104–105, 1971.

[Sze93] M. Szegedy. Functions with bounded symmetric communication complexity, programs over commutative monoids, and ACC. *Journal of Computer and System Sciences*, 47, 1993.

[vL91] J.H. van Lint. *Introduction to Coding Theory*. Springer-Verlag, 1991. Second Edition.