Title: Communication complexity in linearly ordered sets

Author: Małgorzata Serwecińska

Citation style: Serwecińska Małgorzata. (2004). Communication complexity in linearly ordered sets. "Bulletin of the Section of Logic" (Vol. 33, no. 4 (2004), s. 209-222).

Bulletin of the **S**ection of **L**ogic
Volume 33/4 (2004), pp. 209–222

Małgorzata Serwecińska

COMMUNICATION COMPLEXITY IN LINEARLY ORDERED SETS

Abstract

In this paper we consider a communication complexity of the minimum function defined over a linear ordered set. We construct a protocol, the cost of which give an upper bound on the deterministic communication complexity. A lower bound will be derived from the rank lower bound. It seems, that the presented protocol is almost optimal for large sets. Its simply modification give an upper bound on the average communication complexity of the minimum function.

1. Introduction

Communication complexity was introduced by Yao [1] in 1979 year. It was the simplest model of the communication complexity - two-party communication complexity. Let X, Y, Z be arbitrary nonempty, finite sets and let $f: X \times Y \to Z$ be an arbitrary function. We have two players Alice and Bob, who wish to evaluate $f(x, y)$, for some inputs $x \in X$ and $y \in Y$. The difficulty is that Alice knows only x and Bob knows only y. In order to evaluate the value of the function, they will need to communicate with each other according to some fixed protocol. We are only interested in the amount of the communication between Alice and Bob. Thus we assume, that they have unlimited computational power and that local computation is free. The central notion of the communication is a protocol. This is a set of rules specifying the order and the meaning of the sent messages. In protocol each message depends on the previous messages and on the current

players inputs. Additionally, the messages are prefix-free, i.e., no possible message is the beginning (prefix) of another one. This property assures, that the players recognize the end of the message. A protocol terminates, when both players know the value of a function f in a fixed point (x, y). The cost of a protocol is the number of bits transmitted in the worst case. The deterministic communication complexity of the function f, denoted as $D(f)$, is the cost of the best protocol of the function f, in other words, this is the minimum number of bits, which Alice and Bob have to exchange in order to compute the value of the function for the worst inputs.

There is always the simplest protocol, called the trivial protocol. Alice sends her input to Bob (this requires $\lceil \log_2 |X| \rceil$ bits). In this way he knows both inputs x and y and can compute $f(x, y)$ and sends the result back to Alice (this requires $\lceil \log_2 |Z| \rceil$ bits). The cost of this protocol is equal to $\lceil \log_2 |X| \rceil + \lceil \log_2 |Z| \rceil$ bits. If $|Y| < |X|$ then in order to attain the lower cost of the trivial protocol, we can exchange the roles of players. Moreover, if $\max\{|X|, |Y|\} < |Z|$ then Bob can reduce the cost of the protocol by sending the second argument instead of the value of the function.
We thus have:

PROPOSITION 1. *Let* $|X| \leq |Y|$. *Then, for every function* $f: X \times Y \to Z$

$$D(f) \leq \lceil \log_2 |X| \rceil + \min\{\lceil \log_2 |Y| \rceil, \lceil \log_2 |Z| \rceil\}.$$

The lower bound on the communication complexity was estimated by Mehlhorn and Schmidt [3] and is called the rank lower bound.
For each element $z \in Z$ we can assign a boolean matrix $M_z(f)$ to a function f, whose rows and columns are indexed by the input variables X and Y, respectively and

$$M_z(f)_{xy} = \begin{cases} 1, & \text{if } f(x, y) = z \\ 0, & \text{otherwise} \end{cases}.$$

PROPOSITION 2. *For every function* $f: X \times Y \to Z$

$$D(f) \geq \left\lceil \log_2 \sum_{z \in Z} \operatorname{rank}(M_z(f)) \right\rceil.$$

It seems to be obvious that if the sets X, Y, Z are equipped with algebraic structures then the communication complexity can be described more precisely.
Ahlswede, Cai and Tamm [2] considered lattices in role of sets X, Y, Z and three fundamental functions:
let X be a finite lattice,

- let x_{min} be the minimal element of X, then

$$f_1(x,y) = \begin{cases} 1, & \text{if } x \wedge y = x_{min} \\ 0, & \text{otherwise} \end{cases}, \quad (x,y) \in X \times X,$$

- $f_2(x,y) = x \wedge y, \quad (x,y) \in X \times X,$

- let r be the rank function of X, then

$$f_3(x,y) = r(x \wedge y), \quad (x,y) \in X \times X.$$

Under the assumption that above functions are defined over geometric lattices it is possible to estimate communication complexity up to at most one bit (cf. [2]).

In this paper we consider these functions defined over a lattice with a linear order. In this case the value of the first function is equal to one, if and only if one of the players has the minimal element. We can consider the following protocol: each of the players sends the other "1", if he has the minimal element, otherwise he sends "0". Hence $D(f_1) \leq 2$. Using the rank lower bound we conclude that $D(f_1) = 2$.
In a linear lattice the rank function determines uniquely the element of the lattice, hence the third function reduces to the second function, so in the next part of the paper we investigate only f_2. It seems, that the lattice with the linear order has the simplest structure, what should help us to compute the communication complexity of this function. Unfortunately in this case the well known techniques (see e.g. [2], [4]) to determine communication complexity disappoint, because a gap between above lower and upper bound is $O(\log_2 n)$. In the next section we consider a protocol, different from the trivial one, which come near the lower bound.

2. The deterministic communication complexity

In this section we investigate a deterministic communication complexity of the second function defined over a finite lattice X with a linear order. Without lost of generality we can assume, that $X = \{0, 1, \ldots, n-1\}$. In this way f_2 has the form

$$f_2(i,j) = \min\{i,j\}, \qquad 0 \leq i,j \leq n-1.$$

Each number $0, 1, \ldots, n-1$ can be represented by its binary expansion using $\lceil \log_2 n \rceil$ bits. In order to simplify the notation let us denote $M = \lceil \log_2 n \rceil$. For a given binary sequence x, let $N(x)$ denote the non-negative integer m such that x is the binary expansion of m. In order to obtain the upper bound on the deterministic communication complexity of f_2 we construct a protocol and estimate its cost.

Construction of the protocol

Alice and Bob hold some elements $r, s \in X$, respectively. Let $x, y \in \{0,1\}^M$ denote its binary expansion. In the first part of the protocol the players divide the binary expansion of their messages into k blocks of the lengths $v, k-1, k-2, \ldots, 1$, respectively, where $1 \leq v \leq k$. Note, that the lengths of the blocks, except for the first one, form a decreasing arithmetical progression with the last element equal to one. The length of the first block complete the sum of the elements of this progression to the length of the whole message. Hence, it can be equal at most the next element of this arithmetical progression, otherwise the message can be divided into more blocks. Considering that, in order to obtain the value of the number of blocks, it is enough to find the lowest positive integer k such that

$$k + (k-1) + \ldots + 1 \geq M.$$

Thus

$$k^2 + k - 2M \geq 0.$$

It is easy to see that this number is equal to

$$k = \left\lceil \frac{1}{2}(\sqrt{1+8M} - 1) \right\rceil \leq \left\lceil \sqrt{2M} \right\rceil.$$

The next part of the protocol will take place according to the following scenario: let us suppose that $i \in \{1, \ldots, k\}$ and the protocol is not finished yet. Without lost of generality we can assume that Alice begins the communication. At the i–th step Alice sends to Bob the i-th block x_i, so he knows already $x_1, x_2, \ldots, x_i$.
If $N(x_1 \ldots x_i) \leq N(y_1 \ldots y_i)$ then Bob sends zero to Alice, otherwise he sends one followed by $y_i, y_{i+1}, \ldots, y_k$ and the protocol stops.
The protocol can be summarized as follows:

Algorithm ALICE(r)
Input: The number $r \in X$.
Output: The number $f(r, s)$.

Alice divides the binary expansion of her input r into k blocks. She stores these blocks in a table $A[1..k]$.
$i \leftarrow 1$
while $i \leq k$ **do**
    SendBlock($A[i]$)
    AnswerFromBob $\leftarrow$ GetBit()
    if AnswerFromBob=0 **then**
        $i \leftarrow i + 1$
    else
        for $j \leftarrow i$ **to** k **do**
            $B[j] \leftarrow$ GetBlock() {Alice saves in a table $B[1..k]$ the blocks sent by Bob}
        end for
        return $N(A[1] \ldots A[i-1]B[i] \ldots B[k])$
    end if
end while
return r.

Algorithm BOB(s)
Input: The number $s \in X$.
Output: The number $f(r, s)$.

Bob divides the binary expansion of his input s into k blocks. He stores these blocks in a table $B[1..k]$.
$i \leftarrow 1$
while $i \leq k$ **do**
    $A[i] \leftarrow$ GetBlock() {Bob saves in a table $A[1..k]$ the blocks sent by Alice}

```
5:      if N(A[1]...A[i]) ≤ N(B[1]...B[i]) then
6:          SendBit(0)
7:          i ← i + 1
8:      else
9:          SendBit(1)
10:         for j ← i to k do
11:             SendBlock(B[i])
12:         end for
13:         return s
14:     end if
15: end while
16: return N(A[1]...A[k]).
```

We have given the protocol for computing the values of the function f_2. Next we analyze its cost. If in each step of the protocol Bob sends zero to Alice then the protocol consists of k steps and in the last step Bob obtains the value of the function. It is equal to $r = N(x)$. In this case the number of bits transmitted is equal to $M + k$.
If Bob sent one in the i-th step of the protocol then $x_l = y_l$ for all $l < i$ and $N(x_i) > N(y_i)$. In this case

$$x_1 x_2 \dots x_{i-1} y_i \dots y_k = y_1 y_2 \dots y_{i-1} y_i \dots y_k,$$

so after the i-th step both players know y and they set $s = N(y)$ as the value of f_2. The number of bits transmitted is equal to

$$M + i + \text{length}(y_i) = \begin{cases} M + v + 1, & \text{for } i = 1, \\ M + k + 1, & \text{for } 2 \le i \le k \end{cases}.$$

So the cost of the protocol equals

$$M + k + 1 = M + \left\lceil \frac{1}{2}(\sqrt{1 + 8M} - 1) \right\rceil + 1 \le M + \left\lceil \sqrt{2M} \right\rceil + 1$$

$$= \lceil \log_2 n \rceil + \left\lceil \sqrt{2\lceil \log_2 n \rceil} \right\rceil + 1.$$

The presented protocol is the best known protocol for the function f_2. Its cost gives an upper bound on the deterministic communication complexity for f_2. The following theorem estimates the deterministic communication complexity of the minimum function:

THEOREM 1. *Let X be the lattice with linear order and $n = |X| > 1$. Then*

$$\lceil \log_2 n \rceil + 1 \leq D(f_2) \leq \lceil \log_2 n \rceil + \left\lceil \sqrt{2\lceil \log_2 n \rceil} \right\rceil + 1.$$

PROOF. It remains to prove the lower bound. It follows from Proposition 2. Note, that the matrix of the values of the function f_2 has the form

$$M(f_2) = \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & \dots & n-1 \end{bmatrix}.$$

Hence

$$M_k(f_2)_{ij} = 1 \Leftrightarrow (i = k \text{ and } j \geq k) \text{ or } (i \geq k \text{ and } j = k),$$

for all $i, j \in \{0, \dots, n-1\}$ and $k = 0, \dots, n-1$.
It is easy to see that

$$\operatorname{rank}(M_k(f_2)) = 2, \text{ for all } k = 0, \dots, n-2 \text{ and } \operatorname{rank}(M_{n-1}(f_2)) = 1,$$

this implies by Proposition 2, that

$$D(f_2) \geq \left\lceil \log_2 \sum_{k=0}^{n-1} \operatorname{rank}(M_k(f_2)) \right\rceil = \lceil \log_2(2n-1) \rceil .$$

It is easy to check that $\lceil \log_2(2n-1) \rceil = \lceil \log_2 n \rceil + 1$, for $n > 1$.
Hence $D(f_2) \geq \lceil \log_2 n \rceil + 1$.

In the next section we consider the average communication complexity. We give an upper bound on this measure for the minimum function.

3. The average communication complexity

For every protocol $\mathcal{P}$ of the function f, we can compute the average communication complexity. This measure, denoted as $\bar{D}_{\mathcal{P}}(f)$, is defined as follows

$$\bar{D}_{\mathcal{P}}(f) = \frac{1}{|X||Y|} \sum_{(x,y)\in X\times Y} l_{\mathcal{P}}(x,y),$$

where $l_{\mathcal{P}}(x,y)$ is the number of bits transmitted in the protocol $\mathcal{P}$ for the input (x,y). The average communication complexity of the function f, denoted as $\bar{D}(f)$, is defined as

$$\bar{D}(f) = \min\{\bar{D}_{\mathcal{P}}(f) : \mathcal{P} \text{ is a protocol of the function } f\}.$$

It is easy to see, that the average communication complexity is less than or equal to the deterministic communication complexity. This measure is often more difficult to compute than the deterministic communication complexity, because we have to know all the protocols of the function and the distribution of values of $l_{\mathcal{P}}(x,y)$.

The following consideration will be done under the assumption and the notation from the previous section.

First, we compute the average communication complexity of the protocol described in the previous section. In this order, observe that the number of $(r,s) \in X \times X$ such that $r \leq s$ is equal to $\frac{1}{2}n(n+1)$. The number of bits transmitted for such inputs is equal to $M+k$. The number of pairs $(r,s) \in X \times X$ such that $r > s$ equals $\frac{1}{2}n(n-1)$. Let $x, y \in \{0,1\}^M$ be the binary expansion of r, s, respectively. Note that, if $N(x_1) > N(y_1)$ then the number of bits transmitted by the players equals $M+v+1 \leq M+k+1$. If $N(x_1 \dots x_{i-1}) = N(y_1 \dots y_{i-1})$ and $N(x_i) > N(y_i)$, for $i \in \{2, \dots, k\}$ then the number of bits transmitted is equal to $M+k+1$.

Hence,

$$\begin{aligned}
\bar{D}_{\mathcal{P}}(f) &\leq \frac{1}{n^2}\left(\frac{1}{2}n(n+1)(M+k) + \frac{1}{2}n(n-1)(M+k+1)\right) \\
&= \frac{1}{2n}(2nM + 2nk + n - 1) = M + k + \frac{n-1}{2n} \\
&\leq \lceil \log_2 n \rceil + \left\lceil \sqrt{2\lceil \log_2 n \rceil} \right\rceil + \frac{1}{2} - \Omega\left(\frac{1}{n}\right).
\end{aligned}$$

In order to obtain the better upper bound on the average communication complexity of the minimum function let us consider the simply modification of the original protocol. Alice and Bob hold some elements $r, s \in X$, respectively. Let $x, y \in \{0,1\}^M$ denote its binary expansion. The players divide their messages into blocks in the same way as above. Let suppose that $i \in \{1, \ldots, k\}$ and the protocol is not finished yet. At the i-th step of the protocol Alice sends to Bob the block x_i. Bob compares his i-th block with that received from Alice and
if $N(x_i) = N(y_i)$ then Bob sends "0" and the protocol goes to the next step;
if $N(x_i) < N(y_i)$ then Bob sends "10". In this case the value of the function equals Alice's input so she transmits to Bob the blocks $x_{i+1} \ldots x_k$ and the protocol stops;
if $N(x_i) > N(y_i)$ then Bob sends "11" and in this case the value of the function is equal to Bob's input so he transmits to Alice the blocks $y_i \ldots y_k$ and the protocol stops.
The presented protocol can be summarized as follows:

Algorithm ALICE(r)
Input: The number $r \in X$.
Output: The number $f(r, s)$.

```
1: Alice divides the binary expansion of her input r into k blocks. She
   stores these blocks in a table A[1..k].
2: i ← 1
3: while i ≤ k do
4:     SendBlock(A[i])
5:     AnswerFromBob ← GetBit()
6:     if AnswerFromBob=0 then
7:         i ← i + 1
8:     else
9:         AnswerFromBob ← GetBit()
10:        if AnswerFromBob=0 then
11:            for j ← i + 1 to k do
12:                SendBlock(A[j])
13:            end for
14:            return r
15:        else
```

```
16:                 for j ← i to k do
17:                     B[j] ← GetBlock() {Alice saves in a table B[1..k]
                        the blocks sent by Bob}
18:                 end for
19:                 return N(A[1]...A[i − 1]B[i]...B[k])
20:             end if
21:         end if
22: end while
23: return r.
```

Algorithm Bob(s)
Input: The number $s \in X$.
Output: The number $f(r, s)$.

```
1: Bob divides the binary expansion of his input s into k blocks. He
   stores these blocks in a table B[1..k].
2: i ← 1
3: while i ≤ k do
4:     A[i] ← GetBlock() {Bob saves in a table A[1..k] the blocks
       sent by Alice}
5:     if N(A[i]) = N(B[i]) then
6:         SendBit(0)
7:         i ← i + 1
8:     else
9:         SendBit(1)
10:        if N(A[i]) < N(B[i]) then
11:            SendBit(0)
12:            for j ← i + 1 to k do
13:                A[j] ← GetBlock()
14:            end for
15:            return N(A[1]...A[k])
16:        else
17:            SendBit(1)
18:            for j ← i to k do
19:                SendBlock(B[j])
20:            end for
21:            return s
22:        end if
```

23: **end if**
24: **end while**
25: **return** $N(A[1]\dots A[k])$.

It remains to compute the average cost of the presented protocol. Note, that if $r = s$ then the protocol consists of k steps and its cost is equal to $M + k$. The number of inputs $(r, s) \in X \times X$ such that $r = s$ is equal to n. If $r > s$ then there exists $l \in \{1, \dots, k\}$ such that

$$N(x_i) = N(y_i),\ i < l \text{ and } N(x_l) > N(y_l).$$

In this case the protocol consists of l steps. In the last step both players know that the value of the function is equal to Bob's input, so in the last step he takes over the active role in the protocol and sends to Alice his blocks beginning from the l-th block. In this way the cost of the protocol is equal to $M + \text{length}(y_l) + l + 1$. If $l = 1$ then the cost of the protocol equals $M + v + 2 \leq M + k + 2$, otherwise it is equal to $M + k + 2$. The number of pairs $(r, s) \in X \times X$ such that $r > s$ is equal to $\frac{1}{2}n(n-1)$.
If $r < s$ then the protocol consists of l steps, where

$$l = \min\{j : 1 \leq j \leq k \text{ and } N(x_j) < N(y_j)\}.$$

In the last step both players know, that the value of the function equals Alice's input, so she sends to Bob the rest of her message. In this case the cost of the protocol equals $M + l + 1$.
In order to simplify the notation we define the numbers

$$M_0 = 0, \quad M_l = \sum_{i=1}^{l} \text{length}(x_i), \quad l = 1, \dots, k.$$

Further let

$$a_l := |\{(x, y) \in \{0,1\}^M : N(x), N(y) \in X \ \wedge\ (\forall_{i<l} N(x_i) = N(y_i)) \ \wedge\ N(x_l) < N(y_l)\}|,$$

for $l = 1, \dots, k$.
Observe, that if n is a power of two then a_l is equal to the number of pairs (r, s) such that they are equal on the first M_{l-1} bits, on the next $M_l - M_{l-1}$ bits the block x_i is lower than the block y_i and on the rest $M - M_l$ bits they are arbitrary.

Hence,

$$a_l \leq 2^{M_{l-1}} \cdot \frac{1}{2} 2^{M_l - M_{l-1}} (2^{M_l - M_{l-1}} - 1) \cdot 2^{2(M-M_l)}$$
$$= 2^{2M-1}(2^{-M_{l-1}} - 2^{-M_l}), \tag{1}$$

for $l = 1, \ldots, k$.
The average communication complexity of this modified protocol $\mathcal{P}'$ is equal to

$$\bar{D}_{\mathcal{P}'}(f) = \frac{1}{n^2}\left(n(M+k) + \frac{1}{2}n(n-1)(M+k+2) + \sum_{l=1}^{k} a_l(M+l+1)\right)$$

$$= M + \frac{1}{n^2}\left(nk + \frac{1}{2}n(n-1)(k+2) + \sum_{l=1}^{k} a_l(l+1)\right).$$

Note, that

$$\sum_{l=1}^{k} a_l = \frac{1}{2}n(n-1) \tag{2}$$

and from (1) we have

$$\sum_{l=1}^{k} a_l \cdot l = \sum_{i=1}^{k}\sum_{l=i}^{k} a_l \leq 2^{2M-1} \sum_{i=1}^{k}\sum_{l=i}^{k}(2^{-M_{l-1}} - 2^{-M_l})$$

$$= 2^{2M-1}\sum_{i=1}^{k}(2^{-M_{i-1}} - 2^{-M}) = 2^{2M-1}\sum_{i=1}^{k} 2^{-M_{i-1}} - k \cdot 2^{M-1}.$$

Observe that $M_1 \geq 1$ and if $M \geq 5$ then $M_2 \geq 4$, hence

$$\sum_{i=1}^{k} 2^{-M_{i-1}} = 2^{-M_0} + 2^{-M_1} + \ldots + 2^{-M_{k-1}}$$

$$\leq 1 + \frac{1}{2} + \frac{1}{2^4} + \frac{1}{2^5} + \ldots = 1 + \frac{1}{2} + \frac{1}{2^3} = \frac{13}{8}.$$

It is easy to see that this inequality holds also for $M < 5$. Considering that

$$\sum_{l=1}^{k} a_l \cdot l \leq 13 \cdot 2^{2M-4} - k \cdot 2^{M-1}. \tag{3}$$

Since $M = \lceil \log_2 n \rceil$, the number n can be bounded as follows

$$2^{M-1} < n \leq 2^M. \tag{4}$$

Using (2), (3) and (4) we can estimate the average communication complexity

$$\begin{aligned}
\bar{D}_{\mathcal{P}'}(f) &\leq M + \frac{1}{n^2}\left(nk + \frac{n(n-1)}{2}(k+2) + \frac{n(n-1)}{2} + 13 \cdot 2^{2M-4} - k2^{M-1}\right) \\
&= M + \frac{n+1}{2n}k + \frac{3(n-1)}{2n} + \frac{13 \cdot 2^{2M-4}}{n^2} - \frac{k \cdot 2^M}{2n^2} \\
&< M + \frac{n+1}{2n}k + \frac{3(n-1)}{2n} + \frac{13 \cdot 2^{2M-4}}{2^{2M-2}} - \frac{kn}{2n^2} \\
&= M + \frac{1}{2}k + \frac{19}{4} - \frac{3}{2n} \\
&\leq \lceil \log_2 n \rceil + \frac{1}{2}\left\lceil \sqrt{2\lceil \log_2 n \rceil} \right\rceil + \frac{19}{4} - \Omega\left(\frac{1}{n}\right).
\end{aligned}$$

This is an upper bound on the average communication complexity of the minimum function.

Remarks

We can consider also the simpler function $LW\colon X \times X \to \{0,1\}$ defined as follows $LW(x,y) = 1$ if and only if $x < y$. It is easy to prove, that $D(LW) = \lceil \log_2 n \rceil + 1$ (cf. [4], Chapter 1, Example 1.22). Note, that the deterministic communication complexity of the function LW is equal to the lower bound on the deterministic communication complexity of the minimum function, but we see that the estimating the value of the minimum function is harder than the estimating of the value of the lower function. In this way it seems that the deterministic communication complexity of the minimum function will be near the upper bound.

References

[1] A. C. Yao, *Some complexity questions related to distributive computing*, **Proc. 11th Ann. ACM Symp. Theory of Computing** (1979), pp. 209–213.

[2] R. Ahlswede, N. Cai, U.Tamm, *Communication complexity in lattices*, **Appl. Math. Lett.** 6, No. 6 (1993), pp. 53–58.

[3] K. Mehlhorn, E. Schmidt, *Las Vegas is better than determinism in VLSI and distributed computing*, **Proc. 14th Ann. ACM Symp. on Theory of Computing** (1982), pp. 330–337.

[4] E. Kushilevitz, N. Nisan, **Communication complexity**, Cambridge University Press, Cambridge (1997).

Institute of Mathematics, University of Silesia,
40-007 Katowice, ul. Bankowa 14, Poland
e-mail: olszewska@ux2.math.us.edu.pl