SOME BOUNDS ON MULTIPARTY COMMUNICATION COMPLEXITY OF POINTER JUMPING

Carsten Damm, Stasys Jukna
and Jiří Sgall

Abstract. We introduce the model of *conservative one-way* multiparty complexity and prove lower and upper bounds on the complexity of *pointer jumping*.

The pointer jumping function takes as its input a directed layered graph with a starting node and k layers of n nodes, and a single edge from each node to one node from the next layer. The output is the node reached by following k edges from the starting node. In a conservative protocol the ith player can see only the node reached by following the first $i-1$ edges and the edges on the jth layer for each $j > i$. This is a restriction of the general model where the ith player sees edges of all layers except for the ith one. In a one-way protocol, each player communicates only once in a prescribed order: first Player 1 writes a message on the blackboard, then Player 2, etc., until the last player gives the answer. The cost is the total number of bits written on the blackboard.

Our main results are the following bounds on k-party conservative one-way communication complexity of pointer jumping with k layers:

(1) A lower bound of order $\Omega(n/k^2)$ for any $k = O(n^{1/3-\varepsilon})$.

(2) Matching upper and lower bounds of order $\Theta(n \log^{(k-1)} n)$ for $k \leq \log^* n$.

Key words. Multiparty communication complexity, one-way protocols, pointer jumping

Subject classifications. 68C25, 68Q99

1. Introduction

Multiparty games were introduced by Chandra *et al.* (1983) and have found applications as a means of proving lower bounds on the computational complexity of explicit Boolean functions. However, in spite of many results since then, no lower bounds for multiparty games with more than $\log n$ players have been proved. Attacking the barrier of $\log n$ is especially interesting, since by the results of Håstad & Goldmann (1991) such bounds are connected to lower bounds on ACC circuits.

In this paper we introduce a restricted model of conservative one-way protocols, and prove lower bounds for pointer jumping with up to about $n^{1/3}$ players. Due to the restriction to conservative protocols, our lower bounds do not imply new lower bounds in circuit complexity. However, they constitute a partial step in that direction and show the current barriers in lower bound techniques. We also give optimal conservative protocols for pointer jumping with a constant number of players. This shows that even the restricted model can perform nontrivial computations; in fact, our protocols are the best one-way protocols known.

The *pointer jumping function* is defined as follows. The input is a directed layered graph with a starting node and k layers of n nodes, where each node (except for the nodes on the last layer) has exactly one outgoing edge (a pointer) which points to a node from the next layer. The goal is to compute the node on the last layer which is reached by following k pointers from the starting node. Note that the input of the pointer jumping function has $\log n + (k-1)n \log n$ bits rather than kn bits.

For *multiparty games* with k players the input is divided into k pieces. In case of pointer jumping the pieces are the layers of pointers: the first piece is the single pointer from the source to the first layer of nodes, the second piece consists of the n pointers from the first to the second layer, and so on. Player i sees all pieces of the input except for the ith one. The players have unlimited computational power. They share a blackboard, viewed by all players, where they can exchange messages according to a protocol. The objective is to compute the function with as small amount of communication as possible. The cost of a protocol is the number of bits written on the blackboard for the worst case input.

In the general model the number of rounds and the order in which players speak is not limited. However, for many applications, including the bounds on ACC circuits, it would be sufficient to prove lower bounds on some restricted class of protocols. Of particular interest are *one-way (multiparty) protocols* (cf. Papadimitriou & Sipser (1984), Nisan & Wigderson (1993)), where the players speak in a prescribed order, each of them only once: first speaks Player 1, then Player 2, and so on, until Player k gives the answer.

The pointer jumping function is easy to compute in the general model (and, in fact, in any order of players but the one prescribed above): Player 2 writes on the blackboard the first part of the input, which has only $\log n$ bits, and Player 1 computes the answer. However, if Player 1 starts, as required in a one-way protocol, intuitively, without the knowledge of the first pointer he cannot communicate useful information using only a few bits. It is conjectured that the one-way communication complexity of pointer jumping is high, however, so far no nontrivial lower bounds for $k > 3$ were proven.

We say that a multiparty protocol for pointer jumping is *conservative* if Player i does not have complete information about the first $i-1$ layers of pointers, but only knows which node on the $(i-1)$th layer is reachable from the starting point; he also knows the later layers of pointers and the messages communicated so far. The notion of conservativeness can be extended to protocols for computing other functions, see Definition 3.3. The *conservative one-way communication complexity* of a function F is the smallest cost of a conservative one-way protocol computing F.

In this paper we prove the following bounds on the conservative one-way k-party complexity of pointer jumping with k layers.

1. We prove a lower bound of order $\Omega(n/k^2)$ for any $k = O((n/\log n)^{1/3})$. In particular, this means that no conservative protocol can use only $(\log n)^{O(1)}$ players and $n^{1-\varepsilon}$ bits of communication. This should be compared to the general one-way multiparty communication complexity where no lower bounds are known for more than $\log n$ players.

2. We prove that for $k \leq \log^* n - \omega(1)$, any conservative one-way protocol requires at least $n \log^{(k-1)} n(1 - o(1))$ bits of communication. ($\log^{(i)} n$ denotes the iterated logarithm and $\log^* n$ is the number of iterations until $\log^{(i)} n$ drops below 1.)

3. We give a matching construction, namely we construct a conservative one-way protocol which uses only $n \log^{(k-1)} n + O(n)$ bits of communication, for any $k \leq \log^* n$. In particular, for $\log^* n$ (or more) players the protocol uses only $O(n)$ bits. No better one-way protocols are known, even without the restriction to conservative protocols. (In fact, previously no better protocol than the trivial one using $n \log n$ bits was known.)

4. We prove that on a certain restricted input space conservative one-way protocols require more communication than general one-way protocols,

by a factor of at least $\Omega(\log n/\log\log n)$.

In Section 3 we introduce our model and notation. Our upper bounds are proved in Section 4 and lower bounds in Section 5. In Section 6 we demonstrate the gap between conservative and non-conservative protocols. Open problems are discussed in Section 7. The lower and upper bounds for small k were also reported in Damm & Jukna (1995). The preliminary version of this paper appeared in Damm, Jukna & Sgall (1996).

2. Related work

Our main motivation is the result of Håstad & Goldmann (1991), based on Yao (1990) (and following also easily from an improvement of Yao (1990) by Beigel & Tarui (1994)). They show that any function in ACC (i.e., computed by polynomial size, bounded depth and unbounded fan-in circuit with gates computing AND, OR, NOT, and MOD_m for a fixed m) can be computed by a one-way protocol with polylogarithmic number of players and only polylogarithmic cost. Thus improving our lower bounds to non-conservative communication complexity would lead to a proof that pointer jumping is not in ACC.

In fact, as was noted in Babai *et al.* (1995), the result of Håstad & Goldmann (1991) implies that it would be sufficient to prove the lower bounds for *simultaneous* protocols, instead of one-way protocols. In a simultaneous protocol each of the k players sends (independently from the others) one message to a referee, who sees none of the inputs. The referee then announces the result. Thus, in the simultaneous model no communication between the players is allowed, they act independently; the twist is that they share some inputs (if $k \geq 3$).

The model of multiparty communication turns out to be connected to many other computational models. Chandra *et al.* (1983), who introduced the model, used it to prove that majority requires superlinear length constant width branching programs. Babai *et al.* (1992) demonstrate that bounds for one-way communication complexity can be applied to Turing machine, branching program and formulae lower bounds. Nisan & Wigderson (1993) have shown, using a result of Valiant (1977), that a lower bound $\omega(n/\log\log n)$ on 3-party simultaneous protocols for a function f would imply that f cannot be computed by a circuit of linear size and logarithmic depth.

Unfortunately, so far we do not have sufficiently good bounds on the multiparty communication complexity of explicit functions to obtain interesting consequences in circuit complexity. The best lower bounds for the general, one-way, or even simultaneous multiparty complexity are $\Omega(n/2^k)$ lower bounds for the generalized inner product of Babai *et al.* (1992) and Chung & Tetali (1993), for the quadratic residue function of Babai *et al.* (1992), and for the matrix multiplication of Raz (1995). This means that so far we have no lower bounds at all for $k \geq \log n$. Generalized inner product is in ACC, and Grolmusz (1994) showed a matching upper bound $O(kn/2^k)$ for it. However, the other two functions are not believed to be in ACC and are good candidates for multiparty communication complexity lower bounds with many players.

Pointer jumping is also often considered in the context of lower bounds and communication complexity (e.g. Papadimitriou & Sipser (1984), Duriš *et al.* (1987), Grigni & Sipser (1991), Nisan & Wigderson (1993), Bollig *et al.* (1994), Pudlák *et al.* (to appear)). This is an important function, since it simulates many naturally occurring functions, e.g. shifting, addressing, multiplication of binary numbers, convolution, etc. It is easily seen that pointer jumping is LOGSPACE-complete (for $k = n$), which also makes it a good candidate for lower bounds.

Papadimitriou & Sipser (1984) were the first to investigate the communication complexity of pointer jumping. They consider the following 2-party k-round version: we have 2 players, the input consists of pointers from A to B (known to Player 2) and from B to A (known to Player 1), $|A| = |B| = n$. There is a fixed starting point $a_0 \in A$, and the output is given by following k pointers from this starting point. Player 1 (the "wrong" player) starts, and only k messages (rounds of communication) are sent. For this game, the first general lower bound of $\Omega(n/k^2)$ was proved by Duriš *et al.* (1987). Nisan & Wigderson (1993) improved this to a lower bound of order $\Omega(n)$ for deterministic protocols. For ε-error protocols they prove a lower bound of order $\Omega(n/k^2)$ and an upper bound of order $O((n/k)\log n)$.

In contrast to the 2-party case with limited number of rounds, very little is known about k-party one-way complexity of pointer jumping. Wigderson (1996) observed that similar argument as in Nisan & Wigderson (1993) implies an $\Omega(\sqrt{n})$ lower bound for pointer jumping in case of 3 players. For $k > 3$ no non-trivial bounds are known. Even in the simultaneous case, the best bound is $\Omega(n^{1/(k-1)})$ by Babai *et al.* (1995) and Pudlák *et al.* (to appear), which is obtained by an easy information-theoretic argument, and is much smaller than the more general bound on the generalized inner product Babai *et al.* (1992). These bounds are interesting here because they both work even for the restrictions of pointer jumping described next.

Recently, there was interesting progress in proving non-trivial upper bounds for special cases of pointer jumping.

The most general is an $O(n \log\log n)$ bound of Pudlák & Rödl (1993), Pudlák *et al.* (to appear) for $k = 3$ for the special case of pointer jumping when the mapping between the first and second layer of nodes is one-to-one. Intuitively this is the hardest case, but the protocol does not work for general inputs; also the proof uses colorings of random graphs and hence the protocol is non-constructive. We get the same bound for general pointer jumping in the case of $k = 3$. However their requirements on the communication are incomparable with our protocol. Their protocol is almost simultaneous, meaning that the first and second player communicate simultaneously to the third one who announces the result; on the other hand their protocol is not conservative.

Babai *et al.* (1995) and Pudlák *et al.* (to appear) consider the simultaneous communication complexity of functions that are special cases of pointer jumping where the first $k-1$ inputs are restricted to n special functions. The ith input, $i \leq k-1$ is fully determined by some parameter $s_i \in \{0, \dots, n-1\}$. The kth input are arbitrary strings $x \in \{0,1\}^n$ (or equivalently, on the last layer of the graph there are only two nodes, to give a boolean function, cf. Section 6). Nevertheless, even in such special cases finding good protocols is non-trivial. For the generalized addressing function defined by $GAF(s_1, \dots, s_{k-1}, x) = x_{s_1 \oplus \cdots \oplus s_{k-1}}$, where $\oplus$ is the bitwise sum modulo 2 (parity), simultaneous protocols given in Babai *et al.* (1995) use $o(n^{0.92})$ bits of communication for $k = 3$ and $O(\log^2 n)$ bits for logarithmic number of players. For the iterated shift function $shift(s_1, \dots, s_{k-1}, x) = x_{(s_1 + \cdots + s_{k-1}) \bmod n}$ simultaneous protocols from Pudlák *et al.* (to appear) use $O\Big(n(\log\log n / \log n)^k\Big)$ bits for any constant k, and $O(n^{6/7})$ bits for logarithmic number of players. For a polylogarithmic number of players the bound was recently improved by Ambainis (1996) to $O(n^\varepsilon)$ for any constant $\varepsilon > 0$. All these protocols are non-conservative.

For comprehensive information about communication complexity, see the upcoming book of Kushilevitz & Nisan (to appear).

3. Definitions and notation

Throughout the paper we suppose that the sets $A_0, \dots, A_k$ are fixed so that $|A_0| = 1$ and $|A_i| = n$ for $i = 1, \dots, k$. For $i = 1, \dots, k$, $\mathcal{F}_i$ is the set of all functions from A_{i-1} to A_i, $\mathcal{F}(i) = \mathcal{F}_i \times \ldots \times \mathcal{F}_k$, and $\mathcal{F} = \mathcal{F}(1)$. We denote the single element of A_0 by a_0. Given $\langle f_1, \ldots, f_k \rangle \in \mathcal{F}$, we define recursively $a_{i+1} = f_{i+1}(a_i)$.

DEFINITION 3.1. *The* pointer jumping *function* $\mathrm{Jump}^k : \mathcal{F} \to A_k$ *is defined by*

$$\mathrm{Jump}^k(f_1, \ldots, f_k) = f_k(\ldots(f_1(a_0))\ldots) = a_k \textit{ for } \langle f_1, \ldots, f_k \rangle \in \mathcal{F}.$$

We will use notation $\mathrm{Jump}^k(a_1, f_2, \ldots, f_k)$ as equivalent to $\mathrm{Jump}^k(f_1, \ldots, f_k)$ because a_1 and f_1 contain the same information, as there is only one starting point. Also remember that size of the first part f_1 of the input is $\lceil \log n \rceil$ bits and the size of the other parts of the input is $n\lceil \log n \rceil$ bits.

In a *k-party protocol* for a function $F(x_1, \ldots, x_k)$, there are k players, each with unlimited computational power. Player i sees all the inputs except for x_i. Players communicate by "writing on a blackboard" (broadcast). The game starts with the empty blackboard. For each string on the blackboard, the protocol either gives the value of the output (in the case the protocol is over), or specifies which player writes the next bit and what that bit should be as a function of the inputs this player knows (and the string on the board). The blackboard is never erased, players simply append their messages.

DEFINITION 3.2. *A* one-way protocol *is a k-party protocol in which each player writes only one message, in a predetermined order, first Player 1, then Player 2, ..., Player k.*

The string on the board still has to determine who speaks next, and hence in particular for any player and string on the board no message can be a prefix of another message possible in this context.

DEFINITION 3.3. *A* conservative protocol *is a k-party protocol in which the access of players to the input is limited so that Player i knows the function $F(x_1, \ldots, x_{i-1}, *, \ldots, *)$ (with $k-i+1$ unknowns) and the inputs $x_{i+1}, \ldots, x_k$.*

In this definition the "knowledge of the function" should be understood so that instead of knowing the values $x_1, \ldots, x_{i-1}$, the player only knows which function these values induce on the remaining inputs. For many natural functions, including pointer jumping, the number of induced functions is small (compared to the number of values of $\langle x_1, \ldots, x_{i-1}\rangle$), and hence this is a potentially severe restriction. The information of Player i together with x_i determines the output, similarly as in the general k-party setting.

For pointer jumping the knowledge of $\mathrm{Jump}(a_1, f_2, \ldots, f_{i-1}, *, \ldots, *)$ is equivalent to the knowledge of a_{i-1}. Thus a conservative one-way protocol for pointer jumping with k players is given by k mappings

$$\Phi_i : A_{i-1} \times \mathcal{F}(i+1) \times \{0,1\}^* \to \{0,1\}^*.$$

The players communicate according to the following straight-line program:

$$\begin{array}{l}
\text{Player 1 writes } z_1 = \Phi_1(a_0; f_2, \ldots, f_k) \\
\text{Player 2 writes } z_2 = \Phi_2(a_1; f_3, \ldots, f_k; z_1) \\
\text{Player 3 writes } z_3 = \Phi_3(a_2; f_4, \ldots, f_k; z_1, z_2) \\
\qquad\qquad\vdots \\
\text{Player } k \text{ writes } z_k = \Phi_k(a_{k-1}; z_1, z_2, \ldots, z_{k-1})
\end{array}$$

and z_k is the output of the protocol Φ.

In a conservative protocol for pointer jumping we can allow Player i to see $a_1, \ldots, a_{i-1}$, rather than only a_{i-1} without any significant change in cost: Player j can always communicate a_{j-1} in addition to other messages, increasing the complexity by only an additive term of $k \log n$.

All logarithms in the paper are to base 2. The iterated logarithm $\log^{(i)} n$ is defined by $\log^{(0)} n = n$, $\log^{(i+1)} n = \log\log^{(i)} n$. The largest i such that $\log^{(i)} n > 0$ is denoted $\log^* n$. The tower function $\mathrm{TOWER}(i, b)$ is defined by $\mathrm{TOWER}(1, b) = b$, $\mathrm{TOWER}(i+1, b) = 2^{\mathrm{TOWER}(i,b)}$.

4. The Upper Bound

The main idea of the protocol for Jump^k is that each player will "shrink the input space" considerably. Suppose that Player 1 communicates b bits of $f_2(a)$ for every $a \in A_1$. Player 2 sees a_1 and the message of Player 1, and hence he knows b bits of $a_2 = f_2(a_1)$ from the message of Player 1. This means that there are now only about $n/2^b$ possible values for a_2, and if Player 2 repeats the procedure, he can send 2^b bits for each value. We continue this way, each player communicating exponentially more bits for each value, until the $(k-1)$-st player communicates all $\log n$ bits for each possible value. A calculation shows that we need to start with $b = \log^{(k-1)} n$ bits. Note that in our protocol the message of Player i always depends only on a_{i-1}, f_{i+1}, and the previous communication.

THEOREM 4.1. *Let $k \le \log^* n$. Then there is a conservative one-way protocol for* Jump^k *which uses only $n \log^{(k-1)} n + O(n)$ bits of communication.*

PROOF. Let $b = \lceil \log^{(k-1)} n \rceil$, $b_1 = 0$, and $b_i = \mathrm{TOWER}(i-1, b) + i$ for $i > 1$. Due to our choice of b, we have $b \ge 2$, and $\log n \le b_k \le n + k$.

Player i, $i < k$, communicates b_{i+1} bits about each $f_{i+1}(a)$ consistent with previous messages. More precisely, for each $i \ge 1$, we partition A_i into 2^{b_i} blocks of size at most $\lceil n/2^{b_i} \rceil$. Let B_i be the block of A_i containing a_i. Player i, $i < k$, communicates for each $a \in B_i$ which block of A_{i+1} the value $f_{i+1}(a)$ belongs to. We have to verify the player has the necessary information, namely he knows which block is B_i. For Player 1 this is trivial, since $b_1 = 0$ and there is a single block. For $i > 1$, Player i knows a_{i-1}, by the definition of conservative protocols, and hence from the message of Player $i-1$ he knows which block of A_i the value $a_i = f_i(a_{i-1})$ belongs to, and this block is B_i.

The last player announces the answer, namely the single element of B_k. We know that there is only one element since $b_k \ge \log n$.

Now we compute the total amount of communication. Player 1 communicates $b_2 n = (b+2)n = n \log^{(k-1)} n + O(n)$ bits. For $1 < i < k$ we have $b_{i+1} = 2^{b_i - i} + i + 1$, and Player i communicates $b_{i+1}|B_i| \le 2^{b_i - i}\lceil n/2^{b_i} \rceil + (i+1)|B_i| \le n/2^i + 2^{b_i} + (i+1)|B_i|$ bits. Summing the terms of these bounds separately we get that the total communication of all players i, $1 < i < k$, is bounded by $O(n)$. Player k communicates only $\log n$ bits. Hence the total is $n \log^{(k-1)} n + O(n)$, as claimed. $\square$

COROLLARY 4.2. *Let $k \ge \log^* n - O(1)$. Then there is a conservative one-way protocol for* Jump^k *which uses only $O(n)$ bits of communication.*

The same ideas as in Theorem 4.1 can be used for the two-party model with limited number of rounds mentioned in Section 2. Nisan & Wigderson (1993) give ε-error randomized k-round protocols with communication complexity $O((n/k) \log n)$. Our techniques yield deterministic protocols with communication complexity $n \log^{(k-1)} n + O(n)$ for $k \le \log^* n$ (and hence $O(n)$ for $k \ge \log^* n - O(1)$); this is an improvement over Nisan & Wigderson (1993) as long as $k = o(\log n)$. We only need to make one small technical modification since the active player in round i does not know the previous point a_{i-1} from his input. We modify the protocol so that each Player i sends also a_{i-1} (in addition to the message according to our above protocols); then Player i knows a_{i-2} from the previous communication and can compute a_{i-1}. The extra cost is only $k \log n$.

5. The lower bound

We reduce conservative one-way protocols for Jump^k to protocols for Jump^{k-1}. We let Player 1 speak, and then fix one of his messages w, one of the points of the first layer a_1, and a subset of inputs consistent with the communication so far, leaving the players in a setup for Jump^{k-1} with still a large set of inputs on which it is supposed to work. For us a large set of inputs will mean that there is a large set of function tuples $g \in \mathcal{F}(2)$ such that many initial points a_1 are consistent with each g.

Given a finite set Ω (the universe), the *measure* of a subset X is $\mu_\Omega(X) = |X|/|\Omega|$. We usually omit the index Ω, as the universe is clear from the context — we typically use this notation for a set of tuples of functions $F \subseteq \mathcal{F}(i)$ or a set of functions $F \subseteq \mathcal{F}_i$.

Given a conservative one-way protocol for Jump^k, we say that an input $\langle a, g\rangle \in A_1 \times \mathcal{F}(2)$ is *good*, if the protocol answers correctly on that input. A set $G \subseteq \mathcal{F}(2)$ is (α, m)-*large* if $\mu(G) \geq \alpha$ and for every $g \in G$ there exist m values $a \in A_1$ such that the input $\langle a, g\rangle$ is good. A protocol is called (α, m)-*good* if there exists an (α, m)-large set.

A protocol working on all inputs is $(1, n)$-good. No protocol for $Jump^1$ is (α, m)-good if $\alpha > 0$ and $m > 1$, as the only player, Player 1, does not see f_1 and has to announce $a_1 = f_1(a_0)$.

The following quantity plays an important role in our reduction step. We prove the necessary bounds on it later.

DEFINITION 5.1. *Let $\gamma(m, M)$ denote the measure of the family of all functions $f : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that for at least m values of a, $f(a) < M$.*

The setting in which γ is used in the next lemma is the following: If for each a we have a set $T(a)$, $|T(a)| < M$, and we know that for each function f in our family there are at least m coordinates a such that $f(a) \in T(a)$, we can conclude that the measure of this family is at most $\gamma(m, M)$.

LEMMA 5.2. [Reduction Lemma] *Suppose there is an (α, m)-good protocol for* Jump^k *with total communication t. Then there exists an integer $x \le t$ such that for $\beta = \alpha/(n2^{x+1})$ and any integer M satisfying $\gamma(m, M) \le \beta$ there is a (β, M)-good protocol for* Jump^{k-1} *using only $t - x$ bits of communication.*

PROOF. Let $G \subseteq \mathcal{F}(2)$ be an (α, m)-large set of inputs on which the protocol works. First we let Player 1 speak. His communication depends only on $g \in G$, not on $a \in A_1$. By the pigeonhole principle there exists a string w which he communicates on at least $1/2^{|w|}$ fraction of the inputs in G. (This is true even in the case when the length of the message is not determined beforehand, since it has to be determined who writes the next bit on the blackboard, and hence no message of Player 1 is a prefix of another one.) Fix such a w. Let $x = |w|$ and let $\tilde{G}$ be the set of all tuples $g \in G$ on which Player 1 outputs w.

Given $h \in \mathcal{F}(3)$, let F_h be the set of all functions $f \in \mathcal{F}_2$ such that $\langle f, h\rangle \in \tilde{G}$. Define $H \subseteq \mathcal{F}(3)$ as the set of all tuples h for which $\mu(F_h) \ge \alpha/2^{x+1}$. By counting $\mu(H) \ge \alpha/2^{x+1}$. (As $\mu(\tilde{G}) \ge \alpha/2^x$, the measure of pairs $\langle f, h\rangle \in \tilde{G}$ with $\mu(F_h) < \alpha/2^{x+1}$ is at most $\alpha/2^{x+1}$, and the remaining pairs from $\tilde{G}$ have $h \in H$ and hence their measure is at most $\mu(H)$.)

For $h \in H$ and $a \in A_1$, define $T_h(a) = \{f(a) \mid f \in F_h \wedge \langle a, f, h\rangle \text{ is good}\}$, i.e., the set of all images of a under some function f consistent with a, h and the protocol so far. Since G is (α, m)-large, for any $f \in F_h$ there are at least m values of a such that $\langle a, f, h\rangle$ is good, and for each such a, $f(a) \in T_h(a)$. Suppose that for some $h \in H$, $|T_h(a)| < M$ for all a. Then the condition in the definition of γ is satisfied by the family F_h (more precisely, by an isomorphic family, cf. the remark after Definition 5.1), and hence $\mu(F_h) \le \gamma(m, M)$, which contradicts the assumptions of the lemma, since $\mu(F_h) \ge \alpha/2^{x+1} > \beta$. Hence for every $h \in H$ there exists $a \in A_1$ such that $|T_h(a)| \ge M$.

Let $H_a = \{h \in H \mid |T_h(a)| \ge M\}$. From the last paragraph it follows that there exist $a_1 \in A_1$ such that $\mu(H_{a_1}) \ge \mu(H)/n \ge \alpha/(n2^{x+1}) = \beta$. Fix such an a_1.

Now consider the protocol Players 2 to k use on the inputs with a_1 as chosen in the previous paragraph, after Player 1 communicated w. We claim that this is a well-defined (β, M)-good conservative one-way protocol for Jump^{k-1}.

More precisely, suppose that the original protocol Φ for $\mathrm{Jump}^k(a_1, f_2, \ldots, f_k)$ works in the following way:

$$\begin{aligned}
&\text{Player 1 writes } z_1 = \Phi_1(a_0; f_2, \ldots, f_k),\\
&\text{Player 2 writes } z_2 = \Phi_2(a_1; f_3, \ldots, f_k; z_1),\\
&\text{Player 3 writes } z_3 = \Phi_3(a_2; f_4, \ldots, f_k; z_1, z_2),\\
&\qquad\vdots
\end{aligned}$$

Player k writes the output $z_k = \Phi_k(a_{k-1}; z_1, z_2, \ldots, z_{k-1})$.

The new protocol Ψ computes $\mathrm{Jump}^{k-1}(a_2, f_3, \ldots, f_k)$. For convenience we number its players from $2'$ to k', to avoid renumbering of all the inputs. They communicate as follows:

$$\begin{aligned}
&\text{Player } 2' \text{ writes } z'_2 = \Psi_2(a_1; f_3, \ldots, f_k) = \Phi_2(a_1; f_3, \ldots, f_k; w)\\
&\text{Player } 3' \text{ writes } z'_3 = \Psi_3(a_2; f_4, \ldots, f_k; z'_2) = \Phi_3(a_2; f_4, \ldots, f_k; w, z'_2),\\
&\qquad\vdots\\
&\text{Player } k' \text{ writes the output } z'_k = \Psi_k(a_{k-1}; z'_2, \ldots, z'_{k-1}) =\\
&\qquad\qquad\qquad\qquad\qquad\quad = \Phi_k(a_{k-1}; w, z'_2, \ldots, z'_{k-1}).
\end{aligned}$$

By inspection, this is a well-defined conservative protocol, as Players $2'$ to k' have access to all the information they need to compute Ψ_i. To prove that Ψ is (β, M)-good, it is sufficient to verify that H_{a_1} is (β, M)-large for Ψ. Since $\mu(H_{a_1}) \geq \beta$ and for each $h \in H_{a_1}$, $|T_h(a_1)| \geq M$, it is sufficient to verify that Ψ answers correctly on each input $\langle a_2, h\rangle$ where $a_2 \in T_h(a_1)$. Let $f \in F_h$ be such that $a_2 = f(a_1)$ and $\langle a_1, f, h\rangle$ is good; such f exists since $a_2 \in T_h(a_1)$. It follows that Φ is correct on $\langle a_1, f, h\rangle$ and Player 1 (of Φ) communicates w on this input. Let $\langle z'_2, \ldots, z'_k\rangle$ be the messages under Ψ on the input $\langle a_2, h\rangle$ and let $\langle w, z_2, \ldots, z_k\rangle$ be the messages in Φ on the input $\langle a_1, f, h\rangle$. Now we observe that $z'_i = z_i$, and hence Ψ outputs the same answer as Φ. This answer is correct, as $\mathrm{Jump}^{k-1}(a_2, h) = \mathrm{Jump}^k(a_1, f, h)$ and Φ is correct on $\langle a_1, f, h\rangle$. $\square$

The way in which we will use Lemma 5.2 is the following. Given a protocol for $Jump^k$ with total communication t, we construct inductively protocols for $Jump^{k-i+1}$ that are (α_i, m_i)-good for appropriate values of α_i and m_i, both decreasing with increasing i. We start with $\alpha_1 = 1$ and $m_1 = n$, i.e., with a protocol working on all inputs. Our goal is to end up with $\alpha_k > 0$ and $m_k > 1$, which leads to a contradiction as then there exists no (α_k, m_k)-good protocol for Jump^1. Thus we want to keep α_i and m_i as large as possible. We always set $\alpha_{i+1} = \alpha_i/(n2^{x_i+1})$, where x_i is the integer x from the ith application of Lemma 5.2. We can bound α_i in terms of t, since $x_1 + \cdots + x_k \leq t$ and $\alpha_i > 1/2^{t+k+k\log n}$. The rest of the proofs consists only of setting m_i sufficiently large but still satisfying the condition $\gamma(m_i, m_{i+1}) \leq \alpha_{i+1}$.

The first theorem shows that we can iterate the Reduction Lemma about $n^{1/3}$ times. In particular, any protocol with polylogarithmic number of players needs almost linear communication, more than $n^{1-\varepsilon}$ for any $\varepsilon > 0$. To prove this we estimate $\gamma(m, M)$ using Chernoff bounds.

LEMMA 5.3. *There exists an absolute constant $c > 0$ such that if $M \leq m - c\sqrt{sn}$ then $\gamma(m, M) \leq 2^{-s}$.*

PROOF. Pick a function $\mathbf{f} : \{1, \ldots, n\} \to \{1, \ldots, n\}$ uniformly at random. Let $\mathbf{X}_a$ be the indicator random variable of the event that $\mathbf{f}(a) < M$, and let $\mathbf{S} = \sum_{a=1}^{n} \mathbf{X}_a$. We have $\mathrm{Prob}[\mathbf{X}_a = 1] < M/n$ and $\mathrm{E}[\mathbf{S}] < M$. The events $\mathbf{X}_a$ are independent, and thus we can use Chernoff bounds. We get

$$\gamma(m, M) = \mathrm{Prob}[\mathbf{S} \geq m] = \mathrm{Prob}[\mathbf{S} \geq M + c\sqrt{sn}] \leq e^{-c^2 s/2} \leq 2^{-s}$$

for a sufficiently large c. $\square$

THEOREM 5.4. *For $k = o((n/\log n)^{1/3})$, any conservative one-way protocol for* Jump^k *uses at least $\Omega(n/k^2)$ bits of communication.*

PROOF. Suppose that a conservative one-way protocol for Jump^k uses $t = o(n/k^2)$ bits of communication. We iterate the Reduction Lemma $k - 1$ times as described above. Set $s = t + k + k \log n$, $\alpha_1 = 1$, and $\alpha_{i+1} = \alpha_i/(n2^{x_i+1})$, where x_i are the integers from Lemma 5.2. Since $x_1 + \cdots + x_{k-1} \leq t$, we have $\alpha_i > 2^{-s}$ for any $i = 1, \ldots, k$. Now it is sufficient to choose a sequence of integers $m_1 = n$, m_2, $\ldots$, m_k, so that $\gamma(m_i, m_{i+1}) \leq 2^{-s}$ and $m_k > 1$. We set $m_1 = n$ and $m_{i+1} = m_i - c\sqrt{sn}$, where c is the constant from Lemma 5.3. The desired bound $\gamma(m_i, m_{i+1}) \leq 2^{-s}$ holds due to Lemma 5.3. Our assumptions on k and t imply that $s = o(n/k^2)$. Thus $m_k \geq n - o(n) > 1$, which means that the original protocol could not be correct. $\square$

For small k, the lower bound $\Omega\left(n \log^{(k-1)} n\right)$ was proved in Damm & Jukna (1995). This bound can be also obtained from the Reduction Lemma using a better estimate on γ because here we use relatively small values of m.

LEMMA 5.5. *If $n/M \geq 2^{n(s+1)/m}$ then $\gamma(m, M) \leq 2^{-ns}$.*

PROOF. We count the functions directly, first choosing m values that are mapped to numbers up to M:

$$\gamma(m, M) \leq \frac{\binom{n}{m} M^m n^{n-m}}{n^n} \leq 2^n \left(\frac{M}{n}\right)^m \leq 2^n 2^{-n(s+1)} = 2^{-ns}.$$

$\square$

THEOREM 5.6. *For any $k \leq \log^* n - \omega(1)$, every conservative one-way protocol for* Jump^k *uses at least $(n \log^{(k-1)} n)(1 - o(1))$ bits of communication.*

PROOF. Suppose that we have a conservative one-way protocol with $t = (1-\varepsilon)n\log^{(k-1)} n$ bits of communication for some $\varepsilon > 0$. We put $s = (t + k + k\log n)/n$, hence $s < c\log^{(k-1)} n$ for some $c < 1$. We choose $m_i = n/r_i$, where r_i is defined recursively by $r_1 = 1$, $r_{i+1} = 2^{r_i(s+1)}$. By Lemma 5.5, this guarantees that $\gamma(m_i, m_{i+1}) \leq 2^{-ns} \leq \alpha_{i+1}$. A calculation (using $s \geq t/n \geq \omega(1)$) shows that $r_i \leq \mathrm{TOWER}(i, s(1 + o(1)))$. Thus $r_k < n$ and the protocol is not correct. □

Combining the methods used in both theorems we can get a bound that is slightly better than the bound of Theorem 5.4 for k close to $\log^* n$, namely $\Omega(n/(k - \log^* n)^2)$ — in particular every protocol with $\log^* n + O(1)$ players uses $\Omega(n)$ bits of communication. To do this, we choose a suitable constant c and first iterate the Reduction Lemma $k + c - \log^* n$ times as in Theorem 5.4 until $m = n/2$ and then continue as in Theorem 5.6 for the remaining $\log^* n - c - 1$ iterations.

Remark: Impagliazzo (1995) considered another restriction, namely that each communicated bit depends on the previous communication and only on a single layer of pointers chosen among the layers seen by the communicating player; these may be different layers for different bits even if communicated by the same player. Let us call such protocols selective protocols. He noted that any one-way selective protocol needs at least $\Omega(n)$ bits of communication, regardless of the number of players.

Our conservative protocols presented in Section 4 can be made selective by the small modification that each player first communicates a_{i-1}, known to him from the previous layer and a_{i-2} (communicated before). Thus our upper bounds are valid also for selective protocols for $k \leq n/\log n$.

In general, the notion of selective protocols is technically not comparable to conservative protocols: in conservative protocols the players have unrestricted access to all the following levels of input simultaneously, but the access to the previous levels is more restricted than in selective protocols. However, proving lower bounds for selective protocols seems to be significantly simpler since the set of inputs consistent with some partial communication can be maintained to be a Cartesian product of the sets for individual levels. To illustrate this, we sketch the lower bound of $\Omega(n)$ for selective protocols below. We note that for small k it is possible to improve this proof to yield the same bounds as we get for conservative protocols; the necessary combinatorics is essentially the same as in that lower bound.

Let an α-good selective protocol be a selective protocol which works for some set of inputs $G = A \times G_2 \times \cdots \times G_k \subseteq \mathcal{F}$ such that $G_i \subseteq \mathcal{F}_i$, $|A| \geq n/2$ and $\mu(G_i) \geq \alpha$ for $i \geq 2$. We claim that if there exists an α-good protocol for Jump^k with total communication t then there exists an integer x and an $\alpha/2^x$-good selective protocol for Jump^{k-1} with total communication $t - x$, as long as $\alpha/2^t \geq 2^{-n/2}$. The last condition is always true if we start with a 1-good selective protocol with total communication at most $n/2$. No selective protocol is α-good for Jump^1 and $\alpha > 0$, hence the claim proves the $\Omega(n)$ lower bound. To prove the claim, we let Player 1 to speak bit by bit, always choosing the more popular answer; this always reduces size of one G_i, $i \geq 2$, by at most one half. After $x \leq t$ bits Player 1 finishes, and at this point we have $\mu(G_i) \geq \alpha/2^x$ for $i \geq 2$. Now we fix $a \in A$ such that it can be mapped to at least $n/2$ distinct elements of A_2 by functions in G_2; this is possible since now $\mu(G_i) \geq \alpha/2^x \geq \alpha/2^t \geq 2^{-n/2}$ for $i \geq 2$. This gives us a product set on which the rest of the protocol works as an $\alpha/2^x$-good selective protocol. It is important here that since we work with products, we can choose one function in G_2 for each point in A_2, independent of the choice of the input from $G_3 \times \cdots \times G_k$.

6. A gap between conservative and non-conservative protocols

In this section we give a version of pointer jumping which can be computed somewhat cheaper by a non-conservative protocol than by a conservative one.

To exhibit the gap, we modify the three-party pointer jumping function Jump^3 in two ways. First, we use the boolean version of pointer jumping, which is also used in many applications, e.g. Babai *et al.* (1992), Bollig *et al.* (1994). To obtain boolean output, we restrict the number of nodes in the last layer to two. Clearly, this is equivalent to a version of pointer jumping where the last part of input is a string of n bits and the output is its a_{k-1}th bit. In particular for 3 players we define

$$\mathrm{jump}(a, f, x) = x_{f(a)}$$

where $a \in \{1, \ldots, n\}$, $f : \{1, \ldots, n\} \to \{1, \ldots, n\}$, and $x \in \{0, 1\}^n$. Usually the distinction between the boolean and general versions of pointer jumping is not important, as the communication complexity differs at most by a factor of $\log n$. However, our gap is small and hence the distinction is important for us.

Second, we also demand that f is a permutation. Intuitively this is the hardest case of pointer jumping, however, the protocol we use to exhibit the gap does not work for the general case.

THEOREM 6.1. *Let us consider the function* $\mathrm{jump}(a, f, x)$ *defined above with the restriction that* f *is one-to-one. Its general one-way communication complexity is* $O(n \log\log n/\log n)$*, while its conservative one-way communication complexity is* $\Omega(n)$*.*

PROOF. A one-way protocol with required complexity is given in Pudlák & Rödl (1993), Pudlák *et al.* (to appear). Hence we only need to prove that any conservative protocol needs a linear number of bits. The proof is similar to our previous lower bounds, and we only sketch it.

Suppose we have a conservative protocol in which the first two players communicate less than cn bits each, for some small $c > 0$. First, choose a message w sent by Player 1 (seeing f and x) on at least $1/2^{cn-1}$ fraction of inputs. Let X be the set of $x \in \{0,1\}^n$ such on at least a $1/2^{cn}$ fraction of the one-to-one functions $f : \{1, \ldots, n\} \to \{1, \ldots, n\}$ Player 1 outputs w when reading the input $\langle f, x\rangle$. By counting, X contains a $1/2^{cn}$ fraction of strings x. For $a \in \{1, \ldots, n\}$ and $x \in X$, let $T_x(a) = \{f(a) \mid \text{Player 1 communicates } w \text{ on } \langle f, x\rangle\}$. For each $x \in X$ there exists some $a \in \{1, \ldots, n\}$ for which $|T_x(a)| \geq n(1-d)$ where $d \leq 2c/\log n$. This way we can associate with each $x \in X$ a pair (a, T) where $a \in \{1, \ldots, n\}$, $T \subseteq T_x(a)$ and $|T| = n(1-d)$. Choose now a maximal subset $Y \subseteq X$ of strings $x \in X$, all of which lead to the same pair (a, T). Choose further a maximal subset $Z \subseteq Y$ such that for each $x \in Z$ Player 2 outputs the same message if he sees $\langle a, x\rangle$ after Player 1 communicates w. Then $|Z| \geq |X|/\left(n\binom{n}{|T|}2^{cn}\right) \geq 2^{n(1/2-2c)}/n > 2^{dn}$ for sufficiently small c. Thus there exist $x, x' \in Z$ which differ on at least one coordinate $i \in T$. Since $i \in T \subseteq T_x(a) \cap T_{x'}(a)$, there exist functions f and f' such that $f(a) = f'(a) = i$ and the first two players do not distinguish the inputs $\langle a, f, x\rangle$ and $\langle a, f', x'\rangle$. However, $x_i \neq x'_i$, and on one of the inputs the last player's answer must be wrong because in a conservative protocol he can see only the point i and the messages of the first two players. □

It would be interesting to prove any larger gap or any gap on pointer jumping without restrictions. However, even the upper bound we used for this result is highly nontrivial.

7. Conclusion and open problems

We have proved non-trivial bounds for the conservative one-way communication complexity of the pointer jumping function. We feel that for our understanding of communication complexity it is important to formulate restricted models of communication complexity and prove lower bounds for them, similarly as it is important to prove lower bounds for restricted classes of circuits for circuit complexity.

The main open problem remains to find nontrivial lower bounds for general one-way, or even simultaneous, communication. The pointer jumping function seems to be a good candidate for a function not in ACC. It would be therefore important to understand if this function can be computed by simultaneous or one-way protocols with $k = \text{polylog}(n)$ players using only $\text{polylog}(n)$ bits. However, even the following simpler problems are open.

Open Problem 1: Prove any nontrivial ($\omega(\log n)$) lower bound on simultaneous communication complexity for $k > \log n$ players.

Also for small number of players we know very little.

Open Problem 2: For some $\varepsilon > 0$, prove a lower bound of $\Omega(n^{1/2+\varepsilon})$ on simultaneous protocols for pointer jumping with 3 players. Prove a lower bound of $\Omega(n^{\varepsilon})$ for 4 players.

The best protocol we know uses $\Theta(n)$ bits of communication and $\log^* n$ players. We know of no protocol which uses less than n bits, even for more players and in the general one-way model. In our protocols, Player i uses only the knowledge of f_{i+1} (in addition to a_{i-1} and previous messages). It is easy to prove that such protocols need at least n bits, and hence better protocols would have to use significant new ideas.

Open Problem 3: Find a one-way protocol for pointer jumping with $o(n)$ bits of communication and arbitrary number of players.

Acknowledgment

We are thankful to Noam Nisan, Ran Raz, and Avi Wigderson for useful comments on earlier versions of this paper. We would also like to thank the anonymous referees for helpful remarks concerning the presentation.

The second author was supported in part by DFG grant Me 1077/10-1.

The work of the third author was partially supported by grant A119107 of AV ČR. Part of this work was done at Institute of Computer Science, Hebrew University, Jerusalem, Israel; partially supported by a Golda Meir Postgraduate Fellowship.

References

A. Ambainis, Upper bounds on multiparty communication complexity of shifts. In *Proc. 13th Ann. Symp. on Theor. Aspects of Comput. Sci.* Springer Lecture Notes in Comput. Sci., **1046** (1996), 631–642.

L. Babai, N. Nisan, and M. Szegedy, Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. System Sci.* **45** (1992), 204–232.

L. Babai, G. Kimmel, and S. V. Lokam, Simultaneous messages vs. communication. In *Proc. 12th Ann. Symp. on Theor. Aspects of Comput. Sci.* Springer Lecture Notes in Comput. Sci., **900** (1995), 361–372.

R. Beigel and J. Tarui, On ACC. *Comput complexity* (1994), 350–366.

B. Bollig, M. Sauerhoff, D. Sieling, and I. Wegener, On the power of different types of restricted branching programs. *Electronic Colloqium on Computational Complexity*, Report Nr. TR94-026) 1994. To appear in: *Theoretical Comput. Sci.*

A. K. Chandra, M. L. Furst, and R. J. Lipton, Multi-party protocols. In *Proc. 15th Ann. ACM Symp. Theory of Computing*, 1983, 94–99.

F.R.K. Chung and P. Tetali, Communication complexity and quasi-randomness. *SIAM J. Disc. Math.* **6**(1) (1993), 110–123.

C. Damm and S. Jukna, On multiparty games for pointer jumping. Tech. Rep. Forschungsbericht Mathematik/Informatik Nr. 95-09, Universität Trier, 1995.

C. Damm, S. Jukna and J. Sgall, Some bounds on multiparty communication complexity of pointer jumping. In *Proc. 13th Ann. Symp. on Theor. Aspects of Comput. Sci.*, Springer Lecture Notes in Comput. Sci., **1046** (1996), 643–654.

P. Ďuriš, Z. Galil, and G. Schnitger, Lower bounds on communication complexity. *Inform. and Comput.* **73** (1987), 1–22.

M. Grigni and M. Sipser, Monotone separation of logspace from NC^1. In *Proc. 6th Structure in Complexity Theory*. IEEE, 1991, 294–298.

V. Grolmusz, The BNS Lower Bound for Multi-party Protocols is Nearly Optimal. *Inform. and Comput.* **112** (1994), 51–54.

J. Håstad and M. Goldmann, On the power of small-depth threshold circuits. *Comput complexity* **1** (1991), 113–129.

R. Impagliazzo. Personal communication, 1995.

E. Kushilevitz and N. Nisan, *Communication Complexity.* Cambridge University Press, to appear.

N. Nisan and A. Wigderson, Rounds in communication complexity revisited. *SIAM J. Comput.* **22**(1) (1993).

C. Papadimitriou and M. Sipser, Communication complexity. *J. Comput. System Sci.* **28** (1984), 260–269.

P. Pudlák and V. Rödl, Modified ranks of tensors and the size of circuits. In *Proc 24th Ann. ACM Symp. Theory of Computing*, 1993, 523–531.

P. Pudlák, V. Rödl, and J. Sgall, Boolean circuits, tensor ranks and communication complexity. *SIAM J. Comput.* (to appear).

R. Raz, The BNS-Chung criterion for multi-party communication complexity. Manuscript, 1995.

L. G. Valiant, Graph-theoretic arguments in low level complexity. In *Proc. 6th Symp. Math. Found. of Comput. Sci.*, Springer Lecture Notes in Comput. Science (1977), 162–176.

A. Wigderson. Personal communication, 1996.

A. C.-C. Yao, On ACC and threshold circuits. In *Proc. 31st Ann. IEEE Symp. Found. Comput. Sci.*, 1990, 619–627.

Manuscript received ...

Carsten Damm
Fachbereich IV — Informatik
Universität Trier
D-54286 Trier
damm@uni-trier.de

Stasys Jukna
Institute of Mathematics
Akademijos 4
2600 Vilnius
Lithuania

Jiří Sgall
Mathematical Institute, AV ČR
Žitná 25
115 67 Praha 1
Czech Republic
sgallj@beba.cesnet.cz

Current address of S. Jukna:
Fachbereich IV — Informatik
Universität Trier
D-54286 Trier
jukna@uni-trier.de