An algebraic approach to communication complexity

Jean-François Raymond Pascal Tesson

Denis Thérien*

School of Computer Science,
McGill University
3480 rue University, Montréal (PQ)
H3A 2A7, Canada
{raymond,ptesso,denis}@cs.mcgill.ca

February 1, 2005

Abstract

Let M be a finite monoid: define $C^{(k)}(M)$ to be the maximum number of bits that need to be exchanged in the k-party communication game to decide membership in any language recognized by M. We prove the following:

a) If M is a group then, for any k, $C^{(k)}(M) = O(1)$ if M is nilpotent of class $k-1$ and $C^{(k)}(M) = \Theta(n)$ otherwise.

b) If M is aperiodic, then $C^{(2)}(M) = O(1)$ if M is commutative, $C^{(2)}(M) = \Theta(\log n)$ if M belongs to the variety **DA** but is not commutative and $C^{(2)}(M) = \Theta(n)$ otherwise.

We also show that when M is in **DA**, $C^{(k)}(M) = O(1)$ for some k and conjecture that this algebraic condition is also necessary.

1 Introduction

In [20], Yao introduced a communication game where two players try to collaboratively evaluate a function $f(x)$. Each receives half the input bits, and their goal is to compute f by exchanging a minimal amount of information. This model was motivated by questions in distributed computing and VLSI. It has been well studied [11, 12] and variants of the original game have found deep applications in complexity [10, 16, 17].

*Research supported by FCAR and NSERC grants. We would like to thank Peter Bro Miltersen of the University of Aarhus, Denmark for his ideas and comments, that helped us improve some of our lower bounds.

One interesting extension was proposed by Chandra, Furst and Lipton in [5]. They introduced multiparty games where k-players try to compute a k-argument function. Each player has part of the input written on his forehead so that this information is available to all but himself. This model is stronger than the 2-party game, hence lower bounds for it are difficult to prove [1, 5]. Results that have been obtained have many applications in the study of bounded width branching programs [5], pseudorandomness [1], and circuit complexity [7, 8, 9].

In this paper, we will discuss the communication complexity of finite monoids. Since the 50's, algebra has provided powerful tools to analyze and classify languages that can be recognized by finite-state machines (See [6, 14] for comprehensive surveys). More recently, using the formalism of programs over monoids, deep connections have been uncovered between algebraic properties of finite monoids and several natural classes of boolean circuits [2, 4, 13]. There are reasons to suspect the existence of interesting links between monoids and communication complexity. In [17], Szegedy provided an algebraic characterization of languages for which membership can be decided in constant communication by two players: they are exactly the sets recognized by programs over finite commutative monoids.

It thus seems worthwhile to investigate further how the algebraic structure of a finite algebraic system influences the communication complexity of the languages that can be recognized, using morphisms or programs. Our results indicate several interesting connections. We first show that the complexity classes that are defined in the communication framework induce in a natural way "varieties" of monoids (the notion of variety being the relevant one to classify monoids in terms of their computing power). We then study what varieties arise in this way, first if we restrict to groups, then to aperiodic structures. In the group case, we find that for any fixed k, languages recognized by the group G can be decided in constant communication in the k-player game if and only if the group is nilpotent of class $k-1$, and require linear communication otherwise. In the aperiodic case, we show that, for the two-player game, languages recognized by the monoid M will be decided in $O(1)$ communication if M is commutative, in $\Theta(\log n)$ if M belongs to the variety $\mathbf{DA}$ but is not commutative, and in $\Theta(n)$ otherwise. Furthermore, when M is in $\mathbf{DA}$, there exists some k such that corresponding languages can be done in $O(1)$ communication in the k-party game and we conjecture that this condition is also necessary. This points to a natural conjecture on how to extend Szegedy's theorem from 2 players to $O(1)$ players.

Our paper is organized as follows: in section 2, we summarize the needed background on communication complexity and algebra. In section 3, we show that complexity classes defined in the communication complexity framework give rise to varieties of monoids. In section 4 and 5, we discuss our results about groups and aperiodic structures respectively. In the conclusion, we will discuss our conjecture to extend Szegedy's theorem.

2 Background and definitions

2.1 Communication complexity

We give here some formal definitions together with basic results in communication complexity. For a more detailed introduction, we refer the reader to chapters 1, 2 and 6 of [11].

2-party model

In the 2-party game, two players P_1 and P_2, each having unlimited computational power try to jointly compute a function $f : X^n \to Y$, where X and Y are finite alphabets. The set A of input variables is partitioned in two, $A = A_1 \dot{\cup} A_2$, so that each player sees only variables assigned to him. They exchange bits according to a fixed protocol and when the communication stops, both players should know the value of f on the given input.

The cost of a protocol is the maximum number, taken over all possible inputs, of bits exchanged between P_1 and P_2. The communication complexity of f with respect to the partition (A_1, A_2), denoted $C^{(2)}_{(A_1,A_2)}$ is the cost of the cheapest protocol computing f given this partition. The symetric communication complexity (or just the communication complexity) of f, denoted $C^{(2)}(f)$ is the maximum, taken over all possible partitions of the input variables, of f's communication complexity. That is: $C^{(2)}(f) = \max_{A=A_1 \dot{\cup} A_2} C^{(2)}_{(A_1,A_2)}(f)$

Among the many lower bound techniques known for 2-party communication complexity, we will use the so-called fooling-set method. A fooling set for the function $f(x_1, x_2)$ is a set F of input pairs (x_1, x_2) such that for all $(x_1, x_2) \in F$, $f(x_1, x_2) = v$ and for any two distinct (x_1, x_2), $(x'_1, x'_2) \in F$, either $f(x'_1, x_2) \neq v$ or $f(x_1, x'_2) \neq v$.

Lemma 2.1 *If f has a size $|F|$ fooling set, then $C^{(2)}(f) \geq \log |F|$*

Multiparty model

The two-party model can be generalized to k-party in the following way. We have k players $P_1, \ldots, P_k$. The input variables are partitioned in k sets: $A = A_1 \dot{\cup} \ldots \dot{\cup} A_k$, and player P_i knows all the variables except those in A_i. There is a considerable overlap of information since each variable is known to (k-1) players.

The players now communicate by writing bits on a blackboard seen by all players.The protocol should specify which player writes the next bit. This should be a function of the communication thus far, and not of the input. What a party writes depends on the information on the board and on the parts of the input it sees. Protocol cost, communication complexity with respect to a given partition and symetric communication complexity are defined in the obvious way. We denote the k-party communication complexity of f as $C^{(k)}(f)$. As usual, we will be interested in studying the complexity of functions $f : X^* \to Y$ and consequently consider $C^{(k)}(f)$ as a function from $\mathbf{N}$ to $\mathbf{N}$. We define $CC^{(k)}(g(n))$ as the class of functions f such that $C^{(k)}(f) = O(g(n))$.

We state two easy facts about k-party complexity.

Fact 2.2 *If $k < k'$ then $C^{(k)}(f) \geq C^{(k')}(f)$*

Fact 2.3 $C^{(k)}(f) \leq \frac{n}{k} + 1$

2.2 Algebra

Monoids and morphisms

We refer the reader to [6] or [14] for more details on the material sketched here. A monoid is a set together with a binary associative operation and a distinguished identity element. All monoids considered are finite, except for X^*, the free monoid generated by the set X.

Given two monoids M and N, we say that M divides N, and write $M \prec N$, if M is a morphic image of a submonoid of N. The natural unit to classify finite monoids is given by the notion of pseudo-variety (or variety for short). A class $\mathbf{V}$ of finite monoids forms a variety iff it is closed under division and finite direct product.

We say that a language $L \subseteq X^*$ is recognized by the monoid M if there is a morphism $\phi : X^* \to M$ such that $L = \phi^{-1}(F)$ for some subset $F \subseteq M$. It is well known that L is regular iff it can be recognized by some finite monoid M.

Given a regular language $L \subseteq X^*$, define for any word x the set $Y_x = \{(u, v) : uxv \in L\}$. We get a congruence of finite index on X^* by declaring $x \sim_L y$ iff $Y_x = Y_y$. The monoid $X^*/\sim_L$ is called the syntactic monoid of L, and is denoted $M(L)$. This monoid $M(L)$ recognizes L and for any N also recognizing L, $M(L) \prec N$.

Monoid M is a group iff it satisfies the equation $m^q = 1$ for some $q \geq 1$: it is aperiodic iff it satisfies $m^t = m^{t+1}$ for some $t \geq 0$. Groups and aperiodic monoids form varieties. Given a group G, form the sequence $G_0 = G$ and $G_i = [G_{i-1}, G]$ where $[H, G]$ is defined as the subgroup of G generated by all commutators $[h, g] = h^{-1}g^{-1}hg$, $h \in H$ $g \in G$. We say that G is nilpotent of class k if $G_k = \{1\}$: for any $k \geq 0$, nilpotent groups of class k form a variety. We will also mention the variety of commutative monoids, i.e. those satisfying the equation $st = ts$, and the variety $\mathbf{DA}$ which is defined as the class of monoids satisfying $(stu)^n t (stu)^n = (stu)^n$, for some $n \geq 0$.

Monoids and programs

The concept of recognition of a language by a morphism can be extended using programs over monoids. Formally, given an input alphabet A, an n-input program of length l over a monoid M is a sequence of instructions

$$(i_1, f_1), (i_2, f_2) \ldots, (i_l, f_l)$$

with $i_j \in \{1, \ldots, n\}$ and $f_j : A \to M$, together with an accepting subset $F \subseteq M$. The program accepts the input $x_1 x_2 \ldots x_n$ iff the sequence $f_1(x_{i_1}) \ldots f_l(x_{i_l})$ evaluates to an element of F.

We say that $L \subseteq A^*$ is P-recognized by M if there is a sequence $(\varphi_1, \varphi_2, \ldots)$ where φ_n is an n-input program over M, of length $l(n)$ with $l(n)$ polynomial in n, such that the program φ_n accepts $L \cap A^n$.

We say that the monoid M P-divides monoid N and write $M \prec_{\wp} N$ if any language recognized by a morphism over M can be recognized by a polynomial length program over N. If $M \prec N$ then $M \prec_{\wp} N$ since morphisms are just special cases of programs. We say that a class $\mathbf{V}$ of monoids is a P-variety if it is closed under finite direct product and P-division.

We state (without proof) the following :

Fact 2.4 *Let S, T be finite monoids, with $S \prec_{\wp} T$. Suppose T is commutative, then S is commutative.*

3 Communication complexity of finite monoids

In this section we define the communication complexity of a monoid and relate communication complexity classes to monoid varieties. For this section, we fix k, the number of players, to some arbitrary finite value.

3.1 Complexity of monoids

We define the communication complexity of a monoid M as the maximum complexity of any language recognized by M.

The operation on a monoid defines a canonical surjective morphism $eval_M : M^* \to M$

$$eval_M(m_1 m_2 \ldots m_n) = m$$

where $m = m_1 \cdot m_2 \cdot \ldots \cdot m_n$ is the product in M of this sequence of monoid elements.

We say that a regular language $L \subseteq A^*$ has an identity letter if there exists $e \in A$ such that $\phi(e) = 1_M$, where ϕ is the recognizing morphism.

Theorem 3.1 *Let $L \subseteq A^*$ be a regular language and let $M = M(L)$ be its syntactic monoid. Let c_L and c_{eval_M} be the k-party communication complexity of L and $eval_M$ respectively. Then $c_L = O(c_{eval_M})$.*

Moreover, if L has an identity letter then $c_{eval_M} = O(c_L)$ and so $c_L = \Theta(c_{eval_M})$.

Proof: First, we give a $O(c_{eval_M})$-bit k-party protocol for L.

Let $\phi : A^* \to M$ be the recognizing morphism. Given a length n input in A^*, each player privately applies ϕ to each letter of the input. They now have a word of length n in M^*. Using the c_{eval_M}-bit protocol they can evaluate this product. Finally, each player privately checks whether or not the result is in F, where $\phi^{-1}(F) = L$.

For the other direction, we know that each monoid element is an equivalence class of the syntactic congruence of L. Since M is finite, there are only $|M|$ equivalence classes of $\sim_L$. Moreover, each such class corresponds to some unique Y_x, and two different Y's must disagree on some pair (u, v). We can thus choose minimal length pairs $(u_1, v_1), (u_2, v_2), \ldots, (u_t, v_t)$, $t \leq |M|^2$, such that $x \sim_L y$ iff Y_x and Y_y contain exactly the same pairs (u_i, v_i).

Using this fact, we can now build a $O(c_L)$ protocol for $eval_M$.

The players have agreed beforehand on a choice of pairs (u_i, v_i) and, for each $m \in M$ on a word $w \in A^*$ of length q such that $\phi(w) = m$. (Note that this can be done because of the identity letter)

The players are given an input in M^*, say $m_1, \ldots, m_n$. Privately, the players translate this input to a word x of length qn in A^*. Note that

$$eval(m_1, \ldots, m_n) = m \equiv [x]_{\sim_L}$$

Now for each pair (u_i, v_i) the players can use the c_L-bit protocol to determine if (u_i, v_i) is in Y_x. Finally, each player knows Y_x and there is a unique $m \in M$ corresponding to it.

Also, since $c_L = O(n)$, we have $c_L(qn) = O(c_L(t))$. Hence $c_{eval_M} = O(c_L)$. $\square$

This theorem shows that $C^{(k)}(M) = \Theta(C^{(k)}(eval_M))$. This is very convenient as $eval_M$ provides an explicit representation for $C^{(k)}(M)$. In fact, any language L having an identity letter and such that $M(L) = M$ will have the same property.

Lemma 3.2 *Let M, N be finite monoids. Let c_M, c_N and $c_{M\times N}$ be the communication complexity of M, N and $M \times N$ respectively. Then* $\max\{c_M, c_N\} \leq c_{M\times N} \leq c_M + c_N$

Proof: First we can obviously build a protocol of cost $c_{M\times N}$ for $eval_M$, using the identity in N for the N-coordinate. This shows the first inequality.

To show $c_{M\times N} \leq c_M + c_N$, we build a cost $(c_M + c_N)$ protocol for $eval_{M\times N}$ by evaluating the product coordinate-wise using the c_M and c_N-bit protocols. $\square$

Lemma 3.3 *Let M, N be finite monoids such that $M \prec N$. Then $c_M \leq c_N$*

Proof: There is a submonoid $T \subseteq N$ and a surjective morphism $\phi : T \to M$. Given a word in M^*, the k players want to evaluate the product $m_1 m_2 \ldots m_l$ in M. Since ϕ is a surjective morphism, we have

$$m_1 m_2 \ldots m_l = \phi[\phi^{-1}(m_1) \ldots \phi^{-1}(m_l)]$$

Note that ϕ^{-1} is not necessarily defined uniquely, but any choice will do since ϕ is a morphism.

So, on any input in M^*, the players privately translate the input to a word in T^* using a previously agreed upon scheme. (i.e. for every $m \in M$ they have chosen some $t \in T$ such that $\phi(t) = m$)

Now, using the c_N-bit protocol, they can evaluate in T the product

$$\phi^{-1}(m_1)\ldots\phi^{-1}(m_l) = t_0$$

and privately compute $\phi(t_0)$. Note that t_0 is in T, since T is a submonoid and is therefore closed under the operation of N. $\Box$

Lemma 3.4 *Let $L \subseteq A^*$ be such that L can be recognized by a $O(l(n))$-length program over a finite monoid M. Let c_L and c_M be the communication complexities of L and M respectively, then $c_L(n) = O(c_M(l(n)))$. In particular, if $l(n) = n$ then $c_L = O(c_M)$.*

Proof: Given a length n input in A^*, the players can use that program and then evaluate a string of length $O(l(n))$ in M^*, which they can do using $c_M(O(l(n)) = O(c_M(l(n)))$ bits of communication. $\Box$

Corollary 3.5 *Suppose $M \prec_{\wp} N$ and $c_N = O(\log^r(n))$, for some r; then $c_M = O(c_N)$.*

As a direct consequence of these lemmas, we get the following theorem:

Theorem 3.6 *The monoids in $CC^{(k)}(f(n))$, that is monoids whose k-party communication complexity are $O(f(n))$, form a monoid variety. Moreover they form a P-variety if $f(n)$ is polylogarithmic.*

We are interested here in the varieties defined by the following communication complexity classes:

$$CCC^{(k)} = CC^{(k)}(O(1)) \text{ and } CCC = \bigcup_{k\geq 2} CCC^{(k)}$$

3.2 Szegedy's result

In [17], Szegedy proved an algebraic characterization for the class $CCC^{(2)}$. His main theorem is:

Theorem 3.7 (Szegedy) *A language L has bounded 2-party communication complexity iff L can be recognized by an M-program over a commutative monoid M.*

Combining this with theorem 3.6 and fact 2.4, we get the following corollary.

Corollary 3.8 *A finite monoid N has constant communication complexity iff N is commutative.*

Our goal now is to investigate the varieties defined by communication complexity classes. We study separately the cases of groups and aperiodic monoids.

4 The case of groups

In this section, we characterize exactly groups having constant communication complexity in the k-party game, for every $k \geq 2$. In order to do this, we need the combinatorial description of the languages recognized by nilpotent groups of class k that was presented in [18].

We say that $u = u_1 u_2 \dots u_m$ is a subword of x if x can be factorized as $x = A^* u_1 A^* u_2 A^* \dots u_m A^*$. Let $u \in A^*$, $q \geq 1$. Define $|x|_u$ as the number of occurrences of u as a subword of x. Also, define $[u, i, q] = \{x; |x|_u \equiv i \text{ (mod q)}\}$

Denote by $\mathbf{G_{nil,m}}$ the variety of finite nilpotent groups of class $\leq m$.

It is known that $L \subseteq A^*$ is recognized by $G \in \mathbf{G_{nil,m}}$ iff L is in the boolean algebra generated by $\{[u, i, q]; |u| \leq m, q \geq 1\}$. Equivalently, $\mathbf{G_{nil,m}}$ is generated by the syntactic monoids of the languages $[u, i, q]$ where $|u| \leq m$.

Of course in an $(m+1)$-party game, every subword of size m is seen entirely by at least one player, so clearly, if $G \in \mathbf{G_{nil,m}}$ then the $(m+1)$-party communication complexity of G is $O(1)$.

Groups in $\mathbf{G_{nil,m}}$ are actually the only groups in $CCC^{(m+1)}$. To show this, we will use lemma 3.4 together with the powerful lower bound obtained in [1] and generalized by Grolmusz in [7]:

Theorem 4.1 (Grolmusz) *Let $GIP_{k,p} : \{0,1\}^{n \times k} \to \{0,1\}$ be the function defined as $GIP_{k,p}(A) := 1$ iff the number of all 1 rows of A is congruent to 0 (mod p).*

$C^{(k)}(GIP_{k,p}) = \Omega(n/4^k)$ for any prime p.

Let $L_{k,p}$ be the regular language defined by $x \in L$ iff $|x|_{a_1} \equiv 0 \pmod{p}$, $|x|_{a_1 a_2} \equiv 0 \pmod{p}$, $\dots$ $|x|_{a_1 a_2 \dots a_k} \equiv 0 \pmod{p}$.

Let $G_{k,p}$ be the syntactic monoid of $L_{k,p}$. $G_{k,p}$ is a class k nilpotent group.

Lemma 4.2 *For any $t \leq k$, there is a t-input program $B(t)$ over $G_{k,p}$ which outputs an element of order p if all t input bits are on and e otherwise. The length of $B(t)$ is less than 2^{t+1}.*

Proof: For any $i \leq k$, the commutator $[[\dots[a_1, a_2], a_3], \dots, a_i]$ has order p in $G_{k,p}$. Indeed, an easy induction shows that this word contains no subwords (mod p) of the form $a_1 a_2 \dots a_j$ for $j < i$ and contains (mod p) exactly one subword $a_1 a_2 \dots a_i$.

Now, let $C(i)$ be the single instruction outputting a_i if the i^{th} bit is on and e otherwise. We define inductively

$B(1) = C(1)$

$B(i+1) = B(i)C(i+1)B(i)^{-1}C(i+1)^{-1}$

If some bit is off, $B(t)$ will output e. Otherwise it produces the commutator $[[\dots[a_1, a_2], a_3], \dots, a_t]$. which is an element of order p. $\square$

Theorem 4.3 *Let G be a nilpotent group of class k. $C^{(k)}(G) = \Omega(n)$.*

Proof: It is sufficient to show that $C^{(k)}(G_{k,p}) = \Omega(n)$ for any k, p.

By concatenating n copies of the $B(k)$'s defined in the previous lemma, we can get a length $2^{k+1}.n$ program over $G_{k,p}$ recognizing $GIP_{k,p}$. Indeed, $GIP_{k,p}$ counts (mod p) the results of n k-wise AND's.

By lemma 3.4, this shows that $C^{(k)}(G_{k,p}) = \Omega(n)$ for any k, p. $\square$

Theorem 4.4 *Let G be a non-nilpotent group. $C^{(k)}(G) = \Omega(n)$ for any k.*

Proof: Since G is not nilpotent, there is a normal subgroup $H \triangleleft G$ such that $[G, H] = H$, and $|H| \neq 1$.

It was shown in [3] (Theorem 5) that for any $h \in H$, there is a k-input program of length at most $(4 \cdot |H|)^k$ which outputs h if all input bits are on and outputs e otherwise.

Let $h \in H$ be an element of order p, where p divides $|H|$. It is now clear how one can construct a program of length at most $n \cdot (4|H|)^k$ which recognizes $GIP_{k,p}$. Finally, lemma 3.4 yields the result. $\square$

Hence, we have proved the following theorem :

Theorem 4.5 *Let G be a finite group. G is in $CCC^{(k)}$ if and only if G is nilpotent of class $< k$. Otherwise $C^{(k)}(G) = \Omega(n)$.*

5 The case of aperiodic monoids

We now investigate the communication complexity of aperiodic monoids. In the 2-party game, we provide a complete characterization for all aperiodic monoids. We then present some results for the situation where $O(1)$ players are involved.

The case with 2 players

By corollary 3.8, we know that any monoid, hence any aperiodic monoid, has constant communication complexity iff it is commutative. We now proceed to show that every non-commutative aperiodic monoid has communication complexity of either $\Theta(\log n)$ or $\Theta(n)$, depending if the monoid belongs to the variety **DA** or not. Recall that the variety **DA** is defined as the class of monoids that satisfy $(stu)^n t (stu)^n = (stu)^n$ for some n. It is known that $L \subseteq A^*$ is recognized by M in **DA** iff L is a disjoint union of languages of the form $A_0^* a_1 A_1^* \ldots a_k A_k^*$ where $a_i \in A$, $A_i \subseteq A$ and the concatenation is unambiguous [15], which means that whenever we have w in L, the factorization $w = w_0 a_1 w_1 \ldots w_{k-1} a_k w_k$ where $w_i \in A_i^*$, is unique.

Lemma 5.1 *Let $L = A_0^* a_1 A_1^* a_2 \ldots a_k A_k^*$ be an unambiguous concatenation.*
$C^{(2)}(L) = O(\log n)$.

Proof:[Sketch] We will show by induction on k, the number of bookmarks, that an $O(\log n)$ protocol exists for any such language.

For $k = 1$, notice that the only non-trivial unambiguous concatenations $A_0^* a_1 A_1^*$ are $c^* a A^*$ and $A^* a c^*$, where $A = \{a, b, c\}$. The $O(\log n)$ bound is simple in this case: each player gives the position of the first a it sees. Now, that they know where the first a is, they check if every letter preceding it is a c.

For the induction step, notice that if $A_0^* a_1 A_1^* a_2 \dots a_k A_k^*$ is an unambiguous concatenation, then so is $A_{i-1}^* a_i A_i^* a_{i+1} \dots a_j A_j^*$, for any $1 \leq i < j \leq k$.

We say that the players have identified the bookmark a_i at position p if they know that if the input w is in L, then the bookmark a_i is at position p.

By taking advantage of the unambiguity, it is possible to show that the players will always be able to identify one of the bookmarks, using only $O(\log n)$ communication. Thus they can reduce the problem for k bookmarks to two problems of less than k bookmarks.□

Hence, monoids in **DA** have logarithmic 2-party communication complexity.

We state the following fact whose proof will be given in the full paper:

Fact 5.2 *Let M be a finite non-commutative aperiodic monoid. Then M is divided by one of the following aperiodic monoids:*

1. *BA_2, the syntactic monoid of $(c^* a c^* b c^*)^*$*
2. *U, the syntactic monoid of $((b + c)^* a (b + c)^* b (b + c)^*)^*$*
3. *The syntactic monoid of $A^* a A^* b A^*$*
4. *The syntactic monoid of either $c^* a A^*$ or $A^* a c^*$*

*Moreover, if $M \notin$ **DA**, M is divided by BA_2 or U.*

The next two lemmas show that the upper bound proved for monoids in **DA** is tight for non-commutative monoids in this variety.

Lemma 5.3 *Let $L = A^* a A^* b A^*$, we have $C^{(2)}(L) = \Theta(\log n)$.*

Proof: Consider the partition where player 1 gets odd-indexed bits and player 2 gets even-indexed bits. The following is a size $\frac{n}{2}$ fooling set for L: $F = \{b^{2i} a^{n-2i}\}$ None of these words are in L, but for any two words of this set, one of the words in the rectangle they define is of the form $b^j (ab)^l a^{(n-j-2l)}$, which is in L. □

Lemma 5.4 *The unambiguous concatenation $K = c^* a A^*$ on the alphabet $A = \{a, b, c\}$ requires $\Theta(\log n)$ communication for two players.*

Proof: For the partition where player 1 gets only odd-indexed bits, one can provide a size n fooling set of similar nature: $F = \{c^i b a c^{n-2-i}\}$. □

Note that the languages $L = (c^* a c^* b c^*)^*$ and $K = ((b+c)^* a (b+c)^* b (b+c)^*)^*$ used in fact 5.2 to define the monoids BA_2 and U respectively have similarities: L requires that a's and b's alternate while K only requires that there be a b between any two a's. So it is easy to see that BA_2 divides $U \times U$.

Both monoids have linear 2-party communication complexity. Because of the preceeding remark, it is sufficient to establish this lower bound for BA_2.

First, we need a result from [12]

Fact 5.5 *Let $DISJ(x, y)$ be the Disjointness function, defined as*

$$DISJ(x, y) = 1 \text{ iff } x_i = 0 \vee y_i = 0, \text{ for all } 1 \leq i \leq n$$

We have $C^{(2)}(DISJ) = n$.

Lemma 5.6 $C^{(2)}(BA_2) = \Omega(n)$.

Proof: Disjointness can be recognized by the following linear length program over BA_2. For every pair x_i, y_i, we build a four letter string w. Let $w_1 = a$ if $x_i = 1$ $w_2 = a$ if $y_i = 1$ $w_3 = b$ if $x_i = 1$ $w_4 = b$ if $y_i = 1$. Any letter of w is c otherwise. One can easily check that the concatenation of the w's corresponding to each pair x_i, y_i is a program over BA_2 recognizing $DISJ$. $\square$

Summing up the bounds proved in lemmas 5.1, 5.3, 5.4 and 5.6, we get :

Theorem 5.7 *Let M be a finite aperiodic monoid,*

1. $C^{(2)}(M) = \Theta(1)$ *iff M is commutative.*
2. $C^{(2)}(M) = \Theta(\log n)$ *iff M is in* **DA** *and non-commutative.*
3. $C^{(2)}(M) = \Theta(n)$ *iff M is not in* **DA**.

The case with $O(1)$ players

By the results of the previous section, we know that any monoid in **DA** has $O(\log n)$ complexity for $O(1)$ players. However, we can do better and for any monoid $M \in$ **DA**, there is a k such that $C^{(k)}(M) = O(1)$ as the next theorem shows.

Theorem 5.8 *Let $L = A_0^* a_1 A_1^* \dots A_{k-1}^* a_k A_k^*$ be an unambiguous concatenation, then $L \in CCC^{(k+2)}$.*

Proof: We exhibit a constant cost protocol for L.

Each player in turn verifies if the input he sees is in L. If so he outputs a 1 followed by the identities of the players who see each letter he identified as one of the bookmarks in his factorization. Otherwise, he outputs 0.

The protocol accepts the input if two players outputted a 1 and subsequently identified each other. We claim that this protocol is valid.

First, if the input w is in L, then w can be uniquely factorized as $w = w_0 a_1 w_1 \dots w_{k-1} a_k w_k$. Two players, at least, see the bookmarks $a_1, \dots, a_k$. The letters they ignore are taken from the $w_i's$ so they both receive an input which is in L. Moreover, since the concatenation is unambiguous, they must choose these same bookmarks when they factorize their input. Therefore, the protocol will accept.

Conversely, if the protocol accepts, then two players, say P1 and P2, received inputs w^1, w^2 such that $w^1, w^2 \in L$.

Let w^0 be the word obtained from w by erasing letters unknown to either P1 or P2. Since the protocol accepted, w^0 contains the bookmarks of w^1. It can therefore be factorized using these bookmarks. Similarly, w^0 can be factorized using w^2's bookmarks. But by the uniqueness of the factorization, it follows that w^1 and w^2 have the same bookmarks.

Finally, since w^1 and w^2 are in L, and since P1 and P2 collectively know the whole input, w is in L.

The protocol is simultaneous and has cost at most $(k+2) \cdot (1 + \log(k+2))$ $\square$

It is not known whether one can, in general, reduce the number of players and keep the communication complexity constant.

Not all aperiodic monoids are in the variety defined by CCC:

Theorem 5.9 *For any fixed k, $C^{(k)}(U) = \Omega(\log n)$*

Proof: It is shown in [19] that U is universal, that is any boolean function can be recognized by a program of length $O(c^n)$ over U, corresponding to the function's conjunctive normal form.

For any k, we know there exists a language L such that $C^{(k)}(L) = \Omega(n)$. So lemma 3.4 completes the proof. $\square$

6 Conclusion

In this paper, we have established a number of bridges between communication complexity and algebraic structure of finite monoids. A first question that remains, concerns the communication complexity of the language $(c^*ac^*bc^*)^*$ in the case where $O(1)$ players are involved : we conjecture that its communication complexity is $\Theta(n)$. But the real challenge would be to extend Szegedy's result to constant number of players: we are thus looking for a theorem of the form "language L can be decided in constant communication by $O(1)$ players iff L can be recognized by a program over a monoid that belongs to the variety $\mathbf{V}$". We conjecture that $\mathbf{V}$ should be $\mathbf{DA} \vee \mathbf{G_{nil}}$, the smallest variety that contains $\mathbf{DA}$ and all nilpotent groups. This conjecture is consistent with all results that we have obtained. There is also some combinatorial intuition to support that claim : this variety is the largest one for which all languages can be characterized by subwords (order of appearance and counting with respect to a finite index congruence). A detailed discussion of this idea will be presented in the full paper.

References

[1] L. Babai, N. Nisan, and M. Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. Syst. Sci.*, 45(2):204–232, Oct. 1992.

[2] D. A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *J. Comput. Syst. Sci.*, 38(1):150–164, Feb. 1989.

[3] D. A. M. Barrington, H. Straubing, and D. Thérien. Non-uniform automata over groups. *Information and Computation*, 89(2):109–132, Dec. 1990.

[4] D. A. M. Barrington and D. Thérien. Finite monoids and the fine structure of NC^1. *Journal of the ACM*, 35(4):941–952, Oct. 1988.

[5] A. K. Chandra, M. L. Furst, and R. J. Lipton. Multi-party protocols. In *Proc. 15th ACM STOC*, pages 94–99, 1983.

[6] S. Eilenberg. *Automata, Languages and Machines*, volume B. Academic Press, 1976.

[7] V. Grolmusz. Separating the communication complexities of MOD m and MOD p circuits. In *Proc. 33rd IEEE FOCS*, pages 278–287, 1992.

[8] V. Grolmusz. A weight-size trade-off for circuits and MOD m gates. In *Proc. 26th ACM STOC*, pages 68–74, 1994.

[9] J. Håstad and M. Goldmann. On the power of small-depth threshold circuits. In *Proc. 31st IEEE FOCS*, volume II, pages 610–618, 1990.

[10] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-logarithmic depth. In *Proc. 20th ACM STOC*, pages 539–550, 1988.

[11] E. Kushilevitz and N. Nisan. *Communication Complexity.* Cambridge University Press, 1997.

[12] L. Lovász. Communication complexity: a survey. Technical Report CS-TR-204-89, Princeton University, 1989.

[13] P. McKenzie, P. Péladeau, and D. Thérien. NC^1: The automata theoretic viewpoint. *Computational Complexity*, 1:330–359, 1991.

[14] J.-E. Pin. *Varieties of formal languages.* North Oxford Academic Publishers Ltd, London, 1986.

[15] J.-É. Pin, H. Straubing, and D. Thérien. Locally trivial categories and unambiguous concatenation. *J. Pure Applied Algebra*, 52:297–311, 1988.

[16] R. Raz and P. McKenzie. Separation of the monotone NC hierarchy. In *Proc. 38thth IEEE FOCS*, 1997.

[17] M. Szegedy. Functions with bounded symmetric communication complexity, programs over commutative monoids, and ACC. *J. Comput. Syst. Sci.*, 47(3):405–423, 1993.

[18] D. Thérien. Subword counting and nilpotent groups. In L. Cummings, editor, *Combinatorics on Words: Progress and Perspectives*, pages 195–208. Academic Press, 1983.

[19] D. Thérien. Programs over aperiodic monoids. *Theoretical Computer Science*, 64(3):271–280, 29 May 1989.

[20] A. C. Yao. Some complexity questions related to distributive computing. In *Proc. 11^{th} ACM STOC*, pages 209–213, 1979.