New Bounds for the Garden-Hose Model*

Hartmut Klauck[1] and Supartha Podder[2]

1 **CQT and Nanyang Technological University, Singapore**
`hklauck@gmail.com`

2 **CQT Singapore**
`supartha@gmail.com`

Abstract

We show new results about the garden-hose model. Our main results include improved lower bounds based on non-deterministic communication complexity (leading to the previously unknown $\Theta(n)$ bounds for Inner Product mod 2 and Disjointness), as well as an $O(n \cdot \log^3 n)$ upper bound for the Distributed Majority function (previously conjectured to have quadratic complexity). We show an efficient simulation of formulae made of AND, OR, XOR gates in the garden-hose model, which implies that lower bounds on the garden-hose complexity $GH(f)$ of the order $\Omega(n^{2+\epsilon})$ will be hard to obtain for explicit functions. Furthermore we study a time-bounded variant of the model, in which even modest savings in time can lead to exponential lower bounds on the size of garden-hose protocols.

1998 ACM Subject Classification F.1.1 Models of Computation, F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Space Complexity, Communication Complexity, Garden-Hose Model

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2014.481

1 Introduction

1.1 Background: The Model

Recently, Buhrman et al. [4] proposed a new measure of complexity for finite Boolean functions, called *garden-hose complexity*. This measure can be viewed as a type of distributed space complexity, and while its motivation is mainly in applications to position based quantum cryptography, the playful definition of the model is quite appealing in itself. Garden-hose complexity can be viewed as a natural measure of space, when two players with private inputs compute a Boolean function cooperatively. Space-bounded communication complexity has been investigated before [2, 7, 9] (usually for problems with many outputs), and recently Brody et al. [3] have studied a related model of space bounded communication complexity for Boolean functions (see also [17]). In this context the garden-hose model can be viewed as a memoryless model of communication that is also reversible.

To describe the garden-hose model let us consider two neighbors, Alice and Bob. They own adjacent gardens which happen to have s empty water pipes crossing their common boundary. These pipes are the only means of communication available to the two. Their goal is to compute a Boolean function on a pair of private inputs, using water and the pipes across their gardens as a means of communication.[1]

* This work is funded by the Singapore Ministry of Education (partly through the Academic Research Fund Tier 3 MOE2012-T3-1-009) and by the Singapore National Research Foundation.

[1] It should be mentioned that even though Alice and Bob choose to not communicate in any other way, their intentions are not hostile and neither will deviate from a previously agreed upon protocol.

A garden-hose protocol works as follows: There are s shared pipes. Alice takes some pieces of hose and connects pairs of the open ends of the s pipes. She may keep some of the ends open. Bob acts in the same way for his end of the pipes. The connections Alice and Bob place depend on their local inputs x, y, and we stress that every end of a pipe is only connected to at most one other end of a pipe (meaning no Y-shaped pieces of hose may be used to split or combine the flow of water). Finally, Alice connects a water tap to one of those open ends on her side and starts the water. Based on the connections of Alice and Bob, water flows back and forth through the pipes and finally ends up spilling on one side.

If the water spills on Alice's side we define the output to be 0. Otherwise, the water spills on Bob's side and the output value is 1. It is easy to see that due to the way the connections are made the water must eventually spill on one of the two sides, since cycles are not possible.

Note that the pipes can be viewed as a communication channel that can transmit $\log s$ bits, and that the garden-hose protocol is memoryless, i.e., regardless of the previous history, water from pipe i always flows to pipe j if those two pipes are connected. Furthermore computation is reversible, i.e., one can follow the path taken by the water backwards (e.g. by sucking the water back).

Buhrman et al. [4] have shown that it is possible to compute every function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ by playing a garden-hose game. A garden-hose protocol consists of the scheme by which Alice chooses her connections depending on her private input $x \in \{0,1\}^n$ and how Bob chooses his connections depending on his private input $y \in \{0,1\}^n$. Alice also chooses the pipe that is connected to the tap. The protocol computes a function f, if for all inputs with $f(x,y) = 0$ the water spills on Alice's side, and for all inputs with $f(x,y) = 1$ the water spills on Bob's side.

The size of a garden-hose protocol is the number s of pipes used. The garden-hose complexity $GH(f)$ of a function $f(x,y)$ is the minimum number of pipes needed in any garden-hose game that computes the value of f for all x and y such that $f(x,y)$ is defined.

The garden-hose model is originally motivated by an application to quantum position-verification schemes [4]. In this setting the position of a prover is verified via communications between the prover and several verifiers. An attack on such a scheme is performed by several provers, none of which are in the claimed position. [4] proposes a protocol for position-verification that depends on a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, and a certain attack on this scheme requires the attackers to share as many entangled qubits as the garden-hose complexity of f. Hence all f with low garden-hose complexity are not suitable for this task, and it becomes desirable to find explicit functions with large garden-hose complexity.

Buhrman et al. [4] prove a number of results about the garden-hose model:

- Deterministic one-way communication complexity can be used to show lower bounds of up to $\Omega(n/\log n)$ for many functions.
- For the Equality problem they refer to a bound of $GH(\mathit{Equality}) = \Theta(n)$ shown by Pietrzak (the proof implicitly uses the fooling set technique from communication complexity [10] [personal communication]).
- They argue that super-polynomial lower bounds for the garden-hose complexity of a function f imply that the function cannot be computed in Logspace, making such bounds hard to prove for 'explicit' functions.
- They define randomized and quantum variants of the model and show that randomness can be removed at the expense of multiplying size by a factor of $O(n)$ (for quantum larger gaps are known).
- Via a counting argument it is easy to see that most Boolean functions need size $GH(f) = 2^{\Omega(n)}$.

Very recently Chiu et al. [5] have improved the upper bound for the Equality function to $1.359n$ from the previously known $2n$ bound [4].

1.2 Our Results

We study garden-hose complexity and establish several new connections with well studied models like communication complexity, permutation branching programs, and formula size.

We start by showing that non-deterministic communication complexity gives lower bounds on the garden-hose complexity of any function f. This improves the lower bounds of $\Omega(\frac{n}{\log n})$ for several important functions like Inner Product, Disjointness to $\Omega(n)$.

We observe that any 2-way deterministic communication protocol can be converted to a garden-hose protocol so that the complexity $GH(f)$ is upper bounded by the *size* of the protocol tree of the communication protocol.

We then turn to comparing the model to another nonuniform notion of space complexity, namely branching programs. We show how to convert any *permutation branching program* to a garden-hose protocol with only a constant factor loss in size.

The most important application of this simulation is that it allows us to find a garden-hose protocol for the distributed Majority function, $DMAJ(x, y) = 1$ iff $\sum_{i=1}^{n}(x_i \cdot y_i) \geq \frac{n}{2}$, that has size $O(n \cdot \log^3 n)$, disproving the conjecture in [4] that this function has complexity $\Omega(n^2)$.

Using the garden-hose protocols for Majority, Parity, AND, OR, we show upper bounds on the composition of functions with these.

We then show how to convert any Boolean formula with AND, OR, XOR gates to a garden-hose protocol with a small loss in size. In particular, any formula consisting of arbitrary fan-in 2 gates only can be simulated by a garden-hose protocol with a constant factor loss in size. This result strengthens the previous observation that explicit super-polynomial lower bounds for $GH(f)$ will be hard to show: even bounds of $\Omega(n^{2+\epsilon})$ would improve on the long-standing best lower bounds on formula size due to Nečiporuk from 1966 [12]. We can also simulate formulae including a limited number of Majority gates of arbitrary fan-in, so one might be worried that even super-linear lower bounds could be difficult to prove. We argue, however, that for formulae using arbitrary symmetric gates we can still get near-quadratic lower bounds using a Nečiporuk-type method. Nevertheless we have to leave super-linear lower bounds on the garden-hose complexity as an open problem.

Next we define a notion of *time* in garden-hose protocols and prove that for any function f, if we restrict the number of times water can flow through pipes to some value k, we have $GH_k(f) = \Omega(2^{D_k(f)/k})$, where GH_k denotes the time-bounded garden-hose complexity, and D_k the k-round deterministic communication complexity. This result leads to strong lower bounds for the time bounded complexity of e.g. Equality, and to a time-hierarchy based on the pointer jumping problem.

Finally, we further investigate the power of randomness in the garden-hose model by considering private coin randomness ([4] consider only public coin randomness).

1.3 Organization

Most proofs are contained only in the full version of the paper, which is available on the arXiv.

2 Preliminaries

2.1 Definition of the Model

We now describe the garden-hose model in graph terminology. In a garden-hose protocol with s pipes there is a set V of s vertices plus one extra vertex, the *tap* t.

Given their inputs x, y Alice and Bob want to compute $f(x, y)$. Depending on x Alice connects some of the vertices in $V \cup \{t\}$ in pairs by adding edges $E_A(x)$ that form a matching among the vertices in $V \cup \{t\}$. Similarly Bob connects some of the vertices in V in pairs by adding edges $E_B(y)$ that form a matching in V.

Notice that after they have added the additional edges, a path starting from vertex t is formed in the graph $G = (V \cup \{t\}, E_A(x) \cup E_B(y))$. Since no vertex has degree larger than 2, this path is unique and ends at some vertex. We define the output of the game to be the parity of the length of the path starting at t. For instance, if the tap is not connected the path has length 0, and the output is 0. If the tap is connected to another vertex, and that vertex is the end of the path, then the path has length 1 and the output is 1 etc.

A garden-hose protocol for $f : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ is a mapping from $x \in \mathcal{X}$ to matchings among $V \cup \{t\}$ together with a mapping from $y \in \mathcal{Y}$ to matchings among V. The protocol computes $f(x, y)$ if for all x, y the path has even length iff $f(x, y) = 0$. The garden-hose complexity of f is the smallest s such that a garden-hose protocol of size s exists that computes f.

We note that one can form a matrix G_s that has rows labeled by all of Alice's matchings, and columns labeled by Bob's matchings, and contains the parity of the path lengths. A function f has garden-hose complexity s iff its communication matrix is a sub-matrix of G_s. G_s is called the *garden-hose matrix* for size s.

2.2 Communication Complexity, Formulae, Branching Programs

▶ **Definition 1.** Let $f : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}$. In a communication complexity protocol two players Alice and Bob receive inputs x and y from $\{0, 1\}^n$. In the protocol players exchange messages in order to compute $f(x, y)$. Such a protocol is represented by a protocol tree, in which vertices, alternating by layer, belong to Alice or to Bob, edges are labeled with messages, and leaves either accept or reject. See [10] for more details. The communication matrix is the matrix containing $f(x, y)$ in row x and column y.

We say a protocol P correctly computes the function $f(x, y)$ if for all x, y the output of the protocol $P(x, y)$ is equal to $f(x, y)$. The communication complexity of a protocol is the maximum number of bits exchanged for all x, y.

The deterministic communication complexity $D(f)$ of a function f is the complexity of an optimal protocol that computes f.

▶ **Definition 2.** The non-deterministic communication complexity $N(f)$ of a Boolean function f is the length of the communication in an optimal two-player protocol in which Alice and Bob can make non-deterministic guesses, and there are three possible outputs `accept`, `reject`, `undecided`. For each x, y with $f(x, y) = 1$ there is a guess that will make the players accept but there is no guess that will make the players reject, and vice versa for inputs with $f(x, y) = 0$.

Note that the above is the two-sided version of non-deterministic communication complexity. It is well known [10] that $N(f) \leq D(f) \leq O(N^2(f))$, and that these inequalities are tight.

▶ **Definition 3.** In a public coin randomized protocol for f the players have access to a public source of random bits. For all inputs x, y it is required that the protocol gives the correct output with probability $1 - \epsilon$ for some $\epsilon < 1/2$. The public coin randomized communication complexity of f, $R_\epsilon^{pub}(f)$ is the complexity of the optimal public coin randomized protocol. Private coin protocols are defined analogously (players now have access only to private random bits), and their complexity is denoted by $R_\epsilon(f)$.

▶ **Definition 4.** The deterministic communication complexity of protocols with at most k messages exchanged, starting with Alice, is denoted by $D_k(f)$.

▶ **Definition 5.** In a simultaneous message passing protocol, both Alice and Bob send messages m_A, m_B to a referee. The referee, based on m_A, m_B, computes the output. The simultaneous communication complexity of a function f, $R^{||}(f)$, is the cost of the best simultaneous protocol that computes the function f using private randomness and error 1/3.

Next we define Boolean formulae.

▶ **Definition 6.** A Boolean formula is a Boolean circuit whose every node has fan-out 1 (except the output gate). A Boolean formula of depth d is then a tree of depth d. The nodes are labeled by gate functions from a family of allowed gate functions, e.g. the class of the 16 possible functions of the form $f : \{0,1\} \times \{0,1\} \to \{0,1\}$ in case the fan-in is restricted to 2. Another interesting class of gate functions is the class of all symmetric functions (of arbitrary fan-in). The *formula size* of a function f (relative to a class of gate functions) is the smallest number of leaves in a formula computing f.

Finally, we define branching programs. Our definition of permutation branching programs is extended in a slightly non-standard way.

▶ **Definition 7.** A branching program is a directed acyclic graph with one source node and two sink nodes (labeled with `accept` and `reject`). The source node has in-degree 0. The sink nodes have out-degree 0. All non-sink nodes are labeled by variables $x_i \in \{x_1, \cdots, x_n\}$ and have out-degree 2. The computation on an input x starts from the source node and depending on the value of x_i on a node either moves along the left outgoing edge or the right outgoing edge of that node. An input $x \in \{0,1\}^n$ is accepted iff the path defined by x in the branching program leads to the sink node labeled by `accept`. The length of the branching program is the maximum length of any path, and the size is the number of nodes.

A layered branching program of length l is a branching program where all non-sink nodes (except the source) are partitioned into l layers. All the nodes in the same layer query the same variable x_i, and all outgoing edges of the nodes in a layer go to the nodes in the next layer or directly to a sink. The width of a layered branching program is defined to be the maximum number of nodes in any layer of the program. We consider the starting node to be in layer 0 and the sink nodes to be in layer l.

A permutation branching program is a layered branching program, where each layer has the same number k of nodes, and if x_i is queried in layer i, then the edges labeled with 0 between layers i and $i+1$ form an injective mapping from $\{1, \dots, k\}$ to $\{1, \dots, k\} \cup \{\texttt{accept}, \texttt{reject}\}$ (and so do the the edges labeled with 0). Thus, for permutation branching programs if we fix the value of x_i, each node on level $i+1$ has in-degree at most 1.

We call a permutation branching program *strict* if there are no edges to `accept`/`reject` from internal layers. This is the original definition of permutation branching programs. Programs that are not strict are also referred to as *loose* for emphasis.

We denote by $PBP(f)$ the minimal size of a permutation branching program that computes f.

We note that simple functions like AND, OR can easily be computed by linear size loose permutation branching programs of width 2, something that is not possible for strict permutation branching programs [1].

3 Garden-Hose Protocols and Communication Complexity

3.1 Lower Bound via Non-deterministic Communication

In this section we show that non-deterministic communication complexity can be used to lower bound $GH(f)$. This bound is often better than the bound $GH(f) \geq \Omega(D_1(f)/\log(D_1(f)))$ shown in [4], which cannot be larger than $n/\log n$.

▶ **Theorem 8.** $GH(f) \geq N(f) - 1$.

The main idea is that a nondeterministic protocol that simulates the garden-hose game can choose the *set* of pipes that are used on a path used on inputs x, y instead of the path itself, reducing the complexity of the protocol. The set that is guessed may be a superset of the actually used pipes, introducing ambiguity. Nevertheless we can make sure that the additionally guessed pipes form cycles and are thus irrelevant.

As an application consider the function $IP(x, y) = \sum_{i=1}^{n}(x_i \cdot y_i)\ mod\ 2$. It is well known that $N(IP) \geq n + 1$ [10], hence we get that $GH(IP) \geq n$. The same bound holds for Disjointness. These bounds improve on the previous $\Omega(n/\log n)$ bounds for these functions [4]. Furthermore note that the fooling set technique gives only bounds of size $\log^2 n$ for the complexity of IP (see [10]), so the technique previously used to get a linear lower bound for Equality fails for IP.

3.2 $GH(f)$ At Most The Size of a Protocol Tree for f

Buhrman et al. [4] show that any one way communication complexity protocol with complexity $D_1(f)$ can be converted to a garden-hose protocol with $2^{D_1(f)} + 1$ pipes. One-way communication complexity can be much larger than two-way communication [16].

▶ **Theorem 9.** *For any function f, the garden-hose complexity $GH(f)$ is upper bounded by the number of edges in a protocol tree for f.*

The construction is better than the previous one in [4] for problems for which one-way communication is far from the many-round communication complexity.

4 Relating Permutation Branching Programs and the Garden-Hose Model

▶ **Definition 10.** In a garden hose protocol a spilling-pipe on a player's side is a pipe such that water spills out of that pipe on the player's side during the computation for some input x, y.

We say a protocol has multiple spilling-pipes if there is more than one spilling-pipe on Alice's side or on Bob's side.

We now show a technical lemma that helps us compose garden-hose protocols without blowing up the size too much.

▶ **Lemma 11.** *A garden-hose protocol P for f with multiple spilling pipes can be converted to another garden-hose protocol P' for f that has only one spilling pipe on Alice's side and one spilling pipe on Bob's side. The size of P' is at most 3 times the size of P plus 1.*

Next we are going to show that it is possible to convert a (loose) permutation branching program into a garden-hose protocol with only a constant factor increase in size. We are stating a more general fact, namely that the inputs to the branching program we simulate can be functions (with small garden-hose complexity) instead of just variables. This allows us to use composition.

▶ **Lemma 12.** *$GH(g(f_1, f_2, \dots, f_k)) = O(s \cdot \max(C_i)) + O(1)$, where $PBP(g) = s$ and $GH(f_i) = C_i$ and $f_i : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. The f_i do not necessarily have the same inputs x, y.*

A first corollary is the following fact already shown in [4]. Nonuniform Logspace is equal to the class of all languages recognizable by polynomial size families of branching programs. Since reversible Logspace equals deterministic Logspace [11], and a reversible Logspace machine (on a fixed input length) can be transformed into a polynomial size permutation branching program, we get the following.

▶ **Corollary 13.** *Logspace $\subseteq GH(poly(n))$. This holds for any partition of the variables among Alice and Bob.*

5 The Distributed Majority Function

In this section we investigate the complexity of the Distributed Majority function.

▶ **Definition 14.** Distributed Majority: $DMAJ(x, y) = 1$ iff $\sum_i^n (x_i \cdot y_i) \geq \frac{n}{2}$, where $x, y \in \{0,1\}^n$.

Buhrman et al. [4] have conjectured that the complexity of this function is quadratic, which is what is suggested by the naïve garden-hose protocol for the problem. The naïve protocol implicitly keeps one counter for i and one for the sum, leading to quadratic size. Here we describe a construction of a permutation branching program of size $O(n \cdot \log^3 n)$ for Majority, which can then be used to construct a garden-hose protocol for the Distributed Majority function. Note that the Majority function itself can be computed in the garden-hose model using $O(n)$ pipes (for any way to distribute inputs to Alice and Bob), since Alice can just communicate $\sum_i x_i$ to Bob. The advantage of using a permutation branching program to compute Majority is that by Lemma 12 we can then find a garden-hose protocol for the composition of Majority and the Boolean AND, which is the Distributed Majority function. Our construction of a permutation branching program adapts a branching program construction by Sinha and Thathachar [19].

▶ **Lemma 15.** *$PBP(MAJ) = O(n \cdot \log^3 n)$.*

We can now state our result about the composition of functions $f_1, \dots, f_k$ with small garden-hose complexity via a Majority function.

▶ **Lemma 16.** *For $(f_1, f_2, \dots, f_k)$, where each function f_i has garden-hose complexity $GH(f_i)$, $GH(MAJ(f_1, \dots, f_k)) = O(\sum GH(f_i)) \cdot \log^3 k)$.*

The lemma immediately follows from combining Lemma 15 with Lemma 12. Considering $f_i = x_i \wedge y_i$ we get

▶ **Corollary 17.** *The garden-hose complexity of distributed Majority is $O(n \log^3 n)$.*

6 Composition and Connection to Formula Size

We wish to relate $GH(f)$ to the formula size of f. To do so we examine composition of garden-hose protocols by popular gate functions.

▶ **Theorem 18.** *For $(f_1, f_2, \ldots, f_k)$, where each function f_i has garden-hose complexity $GH(f_i)$*

- $GH(\bigvee f_i) = O(\sum GH(f_i))$.
- $GH(\bigwedge f_i) = O(\sum GH(f_i))$.
- $GH(\oplus f_i) = O(\sum GH(f_i))$.
- $GH(MAJ(f_i)) = O(\sum GH(f_i) \cdot \log^3 k)$.

This result follows from Lemma 16 and Lemma 12 combined with the trivial loose permutation branching programs for AND, OR, XOR.

We now turn to the simulation of Boolean formulae by garden-hose protocols. We use the simulation of formulae over the set of all fan-in 2 function by branching programs due to Giel [6].

▶ **Theorem 19.** *Let F be a formula for a Boolean function g on k inputs made of gates $\{\wedge, \vee, \oplus\}$ of arbitrary fan-in. If F has size s and $GH(f_i) \le c$ for all i, then for all constants $\epsilon > 0$ we have $GH(g(f_1, f_2, \ldots, f_k)) \le O(s^{1+\epsilon} \cdot c)$.*

Proof. Giel [6] shows the following simulation result:

▶ **Fact 1.** *Let $\epsilon > 0$ be any constant. Assume there is a formula with arbitrary fan-in 2 gates and size s for a Boolean function f. Then there is a layered branching program of size $O(s^{1+\epsilon})$ and width $O(1)$ that also computes f.*

By inspection of the proof it becomes clear that the constructed branching program is in fact a strict permutation branching program (an exponential increase in the width would yield this property in any case). The theorem follows by applying Lemma 12. ◀

▶ **Corollary 20.** *When the f_i's are single variables $GH(g) \le O(s^{1+\epsilon})$ for all constants $\epsilon > 0$. Thus any lower bound on the garden-hose complexity of a function g yields a slightly smaller lower bound on formula-size (all gates of fan-in 2 allowed).*

The best lower bound of $\Omega(n^2/\log n)$ known for the size of formulae over the basis of all fan-in 2 gate function is due to Nečiporuk [12]. The Nečiporuk lower bound method (based on counting subfunctions) can also be used to give the best general branching program lower bound of $\Omega(n^2/\log^2 n)$ (see [20]).

Due to the above any lower bound larger than $\Omega(n^{2+\epsilon})$ for the garden-hose model would immediately give lower bounds of almost the same magnitude for formula size and permutation branching program size. Proving super-quadratic lower bounds in these models is a long-standing open problem.

Due to the fact that we have small permutation branching programs for Majority, we can even simulate a more general class of formulae involving a limited number of Majority gates.

▶ **Theorem 21.** *Let F be a formula for a Boolean function g on n inputs made of gates $\{\wedge, \vee, \oplus\}$ of arbitrary fan-in. Additionally, on any path from the root to the leaves there may be up to $O(1)$ Majority gates. If F has size s, then for all constants $\epsilon > 0$ we have $GH(g) \le O(s^{1+\epsilon})$.*

Proof. Proceeding in reverse topological order we can replace all sub-formulae below a Majority gate by garden-hose protocols with Theorem 19, increasing the size of the sub-formula. Then we can apply Lemma 16 to replace the sub-formula including the Majority gate by a garden-hose protocol. If the size of the formula below the Majority gate is $\tilde{s}$, then the garden-hose size is $O(\tilde{s}^{1+\epsilon'})$, where the poly-logarithmic factor of Lemma 16 is hidden in the polynomial increase. Since every path from root to leaf has at most $c = O(1)$ Majority gates, and we may choose the ϵ' in Theorem 19 to be smaller than ϵ/c, we get our result. ◀

6.1 The Nečiporuk Bound with Arbitrary Symmetric Gates

Since garden-hose protocols can even simulate formulae containing some arbitrary fan-in Majority gates, the question arises whether one can hope for super-linear lower bounds at all. Maybe it is hard to show super-linear lower bounds for formulae having Majority gates? Note that very small formulae for the Majority function itself are not known (the currently best construction yields formulae of size $O(n^{3.03})$ [18]), hence we cannot argue that Majority gates do not add power to the model. In this subsection we sketch the simple observation that the Nečiporuk method [12] can be used to give good lower bounds for formulae made of *arbitrary symmetric gates of any fan-in.* Hence there is no obstacle to near-quadratic lower bounds from the formula size connection we have shown. We stress that nevertheless we do not have any super-linear lower bounds for the garden-hose model.

We employ the communication complexity notation for the Nečiporuk bound from [8].

▶ **Theorem 22.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function and $B_1, \ldots, B_k$ a partition of the input bits of f. Denote by $D_j(f)$ the deterministic one-way communication complexity of f, when Alice receives all inputs except those in B_j, and Bob the inputs in B_j. Then the size (number of leaves) of any formula consisting of arbitrary symmetric Boolean gates is at least $\sum D_j(f)/\log n$.*

The theorem is as good as the usual Nečiporuk bound except for the log-factor, and can hence be used to show lower bounds of up to $\Omega(n^2/\log^2 n)$ on the formula size of explicit functions like IndirectStorageAccess [20].

7 Time Bounded Garden-Hose Protocols

We now define a notion of time in garden-hose complexity.

▶ **Definition 23.** Given a garden-hose protocol P for computing function f, and an input x, y we refer to the pipes that carry water in P on x, y as the wet pipes. Let T_P denote the maximum number of wet pipes for any input (x, y) in P.

The number of wet pipes on input x, y is equal to the length of the path the water takes and thus corresponds to the time the computation takes. Thus it makes sense to investigate protocols which have bounded time T_P. Furthermore, the question is whether it is possible to simultaneously optimize T_P and the number of pipes used.

▶ **Definition 24.** We define $GH_k(f)$ to be the complexity of an optimal garden-hose protocol P for computing f where for any input (x, y) we have that T_P is bounded by k.

As an example consider the Equality function (test whether $x = y$). The straightforward protocol that compares bit after bit has cost $3n$ but needs time $2n$ in the worst case. On the other hand one can easily obtain a protocol with time 2, that has cost $O(2^n)$: use 2^n pipes to communicate x to Bob. We have the following general lower bound.

▶ **Theorem 25.** *For all Boolean functions f we have $GH_k(f) = \Omega(2^{D_k(f)/k})$, where $D_k(f)$ is the deterministic communication complexity of f with at most k rounds (Alice starting).*

Proof. We rewrite the claim as $D_k(f) = O(k \cdot \log GH_k(f))$.

Let P' be the garden-hose protocol for f that achieves complexity $GH_k(f)$ for f. The deterministic k-round communication protocol for f simulates P' by simply following the flow of the water. In each round Alice or Bob (alternatingly) send the name of the pipe used at that time by P'. ◀

Thus for Equality we have for instance that $GH_{\sqrt{n}}(\mathit{Equality}) = \Omega(2^{\sqrt{n}})$. There is an almost matching upper bound of $GH_{\sqrt{n}}(\mathit{Equality}) = O(2^{\sqrt{n}} \cdot \sqrt{n})$ by using $\sqrt{n}$ blocks of $2^{\sqrt{n}}$ pipes to communicate blocks of $\sqrt{n}$ bits each.

We can easily deduce a time-cost tradeoff from the above: For Equality the product of time and cost is at least $\Omega(n^2/\log n)$, because for time $T < o(n/\log n)$ we get a super-linear bound on the size, whereas for larger T we can use that the size is always at least n.

7.1 A Time-Size Hierarchy

The Pointer Jumping Function is well-studied in communication complexity. We describe a slight restriction of the problem in which the inputs are permutations of $\{1, \dots, n\}$.

▶ **Definition 26.** Let U and V be two disjoint sets of vertices such that $|U| = |V| = n$.

Let $F_A = \{f_A | f_A : U \to V$ and f_A is bijective$\}$ and $F_B = \{f_B | f_B : V \to U$ and f_B is bijective$\}$. For a pair of functions $f_A \in F_A$ and $f_B \in F_B$ define $f(v) = \begin{cases} f_A(v) & \text{if } v \in U \\ f_B(v) & \text{if } v \in V. \end{cases}$

Then $f_0(v) = v$ and $f_k(v) = f(f_{k-1}(v))$.

Finally, the pointer jumping function $PJ_k : F_A \times F_B \to \{0, 1\}$ is defined to be the XOR of all bits in the binary name of $f_k(v_0)$, where v_0 is a fixed vertex in U.

Round-communication hierarchies for PJ_k or related functions are investigated in [15]. Here we observe that PJ_k gives a time-size hierarchy in the garden-hose model. For simplicity we only consider the case where Alice starts.

▶ **Theorem 27.** **1.** *PJ_k can be computed by a garden-hose protocol with time k and size kn.*
2. *Any garden-hose protocol for PJ_k that uses time at most $k-1$ has size $2^{\Omega(n/k)}$ for all $k \le n/(100 \log n)$.*

We note that slightly weaker lower bounds hold for the randomized setting.

8 Randomized Garden-Hose Protocols

We now bring randomness into the picture and investigate its power in the garden-hose model. Buhrman et al [4] have already considered protocols with public randomness. In this section we are mainly interested in the power of private randomness.

▶ **Definition 28.** Let $RGH^{pub}(f)$ denote the minimum complexity of a garden-hose protocol for computing f, where the players have access to public randomness, and the output is correct with probability 2/3 (over the randomness). Similarly, we can define $RGH^{pri}(f)$, the cost of garden-hose protocols with access to private randomness.

By standard fingerprinting ideas [10] we can observe the following.

▶ **Claim 1.** *$RGH^{pub}(\mathit{Equality}) = O(1)$.*

▶ **Claim 2.** *$RGH^{pri}(Equality) = O(n)$, and this is achieved by a constant time protocol.*

Proof. The second claim follows from Newman's theorem [13] showing that any public coin protocol with communication cost c can be converted into a private coin protocol with communication cost $c + \log n + O(1)$ bits on inputs of length n together with the standard public coin protocol for Equality, and the protocol tree simulation of Theorem 9. ◀

Of course we already know that even the deterministic complexity of Equality is $O(n)$, hence the only thing achieved by the above protocol is the reduction in time complexity. Note that due to our result of the previous section computing Equality deterministically in constant time needs exponentially many pipes.

Buhrman et al. [4] have shown how to de-randomize a public coin protocols at the cost of increasing size by a factor of $O(n)$, so the factor n in the separation between public coin and deterministic protocols above is the best that can be achieved. This raises the question whether private coin protocols can ever be more efficient in size than the optimal deterministic protocol. We now show that there are no very efficient private coin protocols for Equality.

▶ **Claim 3.** *$RGH^{pri}(Equality) = \Omega(\sqrt{n}/\log n)$*

Proof. To prove this we first note that $RGH^{pri}(f) = \Omega(R^{||}(f)/\log R^{||}(f))$, where $R^{||}(f)$ is the cost of randomized private coin simultaneous message protocols for f (Alice and Bob can send their connections to the referee). Hence, $RGH^{pri}(f) = \Omega(R^{||pri}(f)/\log R^{||pri}(f))$, but Newman and Szegedy [14] show that $RGH^{pri}(Equality) = \Omega(\sqrt{n})$. ◀

9 Open Problems

- We show that getting lower bounds on $GH(f)$ larger than $\Omega(n^{2+\epsilon})$ will be hard. But we know of no obstacles to proving super-linear lower bounds.
- Possible candidates for quadratic lower bounds could be the Disjointness function with set size n and universe size n^2, and the IndirectStorageAccess function.
- Consider the garden-hose matrix G_s as a communication matrix. How many distinct rows does G_s have? What is the deterministic communication complexity of G_s? The best upper bound is $O(s \log s)$, and the lower bound is $\Omega(s)$. An improved lower bound would give a problem, for which $D(f)$ is larger than $GH(f)$.
- We have proved $RGH^{pri}(Equality) = \Omega(\sqrt{n}/\log n)$. Is it true that $RGH^{pri}(Equality) = \Theta(n)$? Is there any problem where $RGH^{pri}(f)$ is smaller than $GH(f)$?
- It would be interesting to investigate the relation between the garden-hose model and memoryless communication complexity, i.e., a model in which Alice and Bob must send messages depending on their input and the message just received only. The garden-hose model is memoryless, but also reversible.

Acknowledgement. We thank an anonymous referee for pointing out a mistake in an earlier version of this paper.

References

1 D. A. Barrington. Width-3 permutation branching programs, 1985. Technical report, MIT/LCS/TM-293.

2 P. Beame, M. Tompa, and P. Yan. Communication-space tradeoffs for unrestricted protocols. *SIAM Journal on Computing*, 23(3):652–661, 1994. Earlier version in FOCS'90.

3 Joshua Brody, Shiteng Chen, Periklis A. Papakonstantinou, Hao Song, and Xiaoming Sun. Space-bounded communication complexity. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 159–172, 2013.

4 Harry Buhrman, Serge Fehr, Christian Schaffner, and Florian Speelman. The garden-hose model. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 145–158. ACM, 2013.

5 Well Y. Chiu, Mario Szegedy, Chengu Wang, and Yixin Xu. The garden hose complexity for the equality function. *arXiv:1312.7222*, 2013.

6 O. Giel. Branching program size is almost linear in formula size. *Journal of Computer and System Sciences*, 63(2):222–235, 2001.

7 H. Klauck. Quantum and classical communication-space tradeoffs from rectangle bounds. In *Proceedings of FSTTCS*, 2004.

8 H. Klauck. One-Way Communication Complexity and the Nečiporuk Lower Bound on Formula Size. *SIAM J. Comput.*, 37(2):552–583, 2007.

9 H. Klauck, R. Špalek, and R. de Wolf. Quantum and classical strong direct product theorems and optimal time-space tradeoffs. *SIAM Journal on Computing*, 36(5):1472–1493, 2007. Earlier version in FOCS'04. quant-ph/0402123.

10 Eyal Kushilevitz and Noam Nisan. *Communication Complexity.* Cambridge University Press, 1997.

11 K. J. Lange, P. McKenzie, and A. Tapp. Reversible space equals deterministic space. *Journal of Computer and System Sciences*, 2(60):354–367, 2000.

12 E. I. Nečiporuk. A boolean function. *Soviet Mathematics Doklady*, 7(S 999), 1966.

13 I. Newman. Private vs. common random bits in communication complexity. *Information Processing Letters*, 39(2):67–71, 1991.

14 Ilan Newman and Mario Szegedy. Public vs. private coin flips in one round communication games (extended abstract). In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC'96, pages 561–570, 1996.

15 Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SIAM J. Comput.*, 22(1):211–219, February 1993.

16 C. H. Papadimitriou and M. Sipser. Communication complexity. *Journal of Computer and System Sciences*, 28(2):260–269, 1984. Earlier version in STOC'82.

17 P. Papakonstantinou, D. Scheder, and H. Song. Overlays and limited memory communication mode(l)s. In *Proc. of the 29th Conference on Computational Complexity*, 2014.

18 I. S. Sergeev. Upper bounds for the formula size of symmetric boolean functions. *Russian Mathematics, Iz. VUZ*, 58(5):30–42, 2014.

19 Rakesh Kumar Sinha and Jayram S Thathachar. Efficient oblivious branching programs for threshold and mod functions. *Journal of Computer and System Sciences*, 55(3):373–384, 1997.

20 I. Wegener. *The Complexity of Boolean Functions.* Wiley-Teubner Series in Computer Science, 1987.