Lower bounds using Kolmogorov complexity

Sophie Laplante

LRI, Université Paris-Sud XI, 91405 Orsay CEDEX, FRANCE
laplante@lri.fr

Abstract. In this paper, we survey a few recent applications of Kolmogorov complexity to lower bounds in several models of computation. We consider KI complexity of Boolean functions, which gives the complexity of finding a bit where inputs differ, for pairs of inputs that map to different function values. This measure and variants thereof were shown to imply lower bounds for quantum and randomized decision tree complexity (or query complexity) [LM04]. We give a similar result for deterministic decision trees as well. It was later shown in [LLS05] that KI complexity gives lower bounds for circuit depth. We review those results here, emphasizing simple proofs using Kolmogorov complexity, instead of strongest possible lower bounds.

We also present a Kolmogorov complexity alternative to Yao's min-max principle [LL04]. As an example, this is applied to randomized one-way communication complexity.

Keywords: Lower bounds, Kolmogorov complexity, circuit complexity, query complexity, communication complexity.

1 Introduction

Kolmogorov complexity has been used in a variety of settings to prove lower bounds and other complexity results. However, until recently, the methods have been *ad hoc*, tailored to a particular problem and a particular computational model. In the past few years, techniques have been developed that apply to any Boolean function, and to a wide variety of computational models, so that a single analysis yields lower bounds in multiple models. In this paper, we review these results and present them in a unified setting, called KI complexity. We also present a Kolmogorov-based alternative to Yao's min-max principle, and apply it to one-way randomized communication complexity.

2 Preliminaries

Kolmogorov complexity is the main tool that is used to prove lower bounds in this paper, and we recall the main notions here. We also present the models of computation used in the paper.

2.1 Kolmogorov complexity

Kolmogorov complexity captures well the information theoretic component of many lower bound arguments. We review a few of its main properties in this section.

Definition 1. *Let M be a Turing machine. Let x and y be finite strings.*

1. *The* Kolmogorov complexity of x given y with respect to M *is denoted* $\mathsf{C}_M(x|y)$*, and defined as follows:*

$$\mathsf{C}_M(x|y) = \min(|P| \text{ such that } M(P,y) = x).$$

2. *A set of strings is* prefix-free *if no string is a prefix of another in the set.*
3. *A Turing machine M' is prefix-free if the set of programs is prefix-free, that is, the set $\{P : \exists x\, M'(P,x) \neq \epsilon\}$, where ϵ is the empty string, is prefix-free.*
4. *The* prefix-free Kolmogorov complexity of x given y with respect to a prefix-free Turing Machine M' *is denoted* $\mathsf{K}_{M'}(x|y)$*, and defined as follows:*

$$\mathsf{K}_{M'}(x|y) = \min(|P| \text{ such that } M'(P,y) = x),$$

In the rest of the paper M is a fixed prefix-free universal Turing machine, and we will write K instead of $\mathsf{K}_{M'}$. When y is the empty string, we write $\mathsf{K}(x)$ instead of $\mathsf{K}(x|y)$. To simplify notation we omit additive terms in the upper bounds.

Incompressibility Perhaps the most important property of Kolmogorov complexity that we use for lower bounds is the existence of incompressible strings, that is, strings whose shortest description is maximal.

Proposition 1. *[Incompressibility] For any finite set $A \subseteq \{0,1\}^*$, and any string σ, there exists $x \in A$ such that $\mathsf{K}(x|\sigma) \geq \log(\#A)$.*

The proposition is proved by comparing the number of succinct programs ($2^l - 1$ have length strictly less than l), with the number of strings ($\#A$) that these programs are purported to describe, and conclude by applying the pigeonhole principle.

This should be compared with the corresponding upper bound.

Proposition 2. *For any finite set $A \subseteq \{0,1\}^*$, $\exists \sigma$, $\forall x \in A$, $\mathsf{K}(x|\sigma) \leq \log(\#A)$.*

To describe x, it suffices to give an index into some pre-determined enumeration of the set A, which can be encoded in σ.

We will also need Kraft's inequality.

Proposition 3 (Kraft's inequality). *Let S be any prefix-free set of finite strings. Then $\sum_{x \in S} 2^{-|x|} \leq 1$.*

We shall also use the following bound on conditional Kolmogorov complexity.

Proposition 4. *There is a constant $c \geq 0$ such that for any three strings x, y, z,*

$$\mathsf{K}(z|x) \geq \mathsf{K}(x,y) - \mathsf{K}(x) - \mathsf{K}(y|z,x) + \mathsf{K}(z|x,y,\mathsf{K}(x,y)) - c.$$

The proof uses symmetry of information in an essential way.

2.2 Decision trees and query complexity

A decision tree is a rooted binary tree, where each internal node is labeled with an integer i referencing an input variable, one of the outgoing edges of an internal node is labeled 0 and the other is labeled 1, and each leaf is labeled with an output value. The tree is evaluated on an input $x = x_1 \cdots x_n$, starting at the root, by evaluating x_i if the node is labeled i and following the corresponding edge, and so on, until a leaf is reached, and outputing the value at the leaf. A decision tree T computes f if the output on x equals $f(x)$, for all x. The decision tree complexity of f, written $\mathsf{DT}(f)$, is the depth of the shallowest decision tree that computes f.

We also consider quantum and randomized analogues of decision trees. In these models, the complexity measure is the number of queries to the input, but unlike the classical case, queries can be made in superposition, in the quantum case, or according to some distribution, in the randomized case. Access to the input is achieved by way of a query operator O_x, which behaves like a classical query on classical states, but in the quantum case, it is defined as a unitary matrix O_x that satisfies $O_x|i, z, w\rangle = |i, z \oplus x_i, w\rangle$, for every i, z, w, where i represents a query, z is a register to hold the answer to the query, and w is the remainder of the workspace of the algorithm. Randomized queries can be defined similarly, except the matrix is stochastic. The *query complexity* of an algorithm is the number of calls to O_x. Details of the model can be found for example in [LM04], but they are not necessary for this paper.

We say that the algorithm A *ε-computes* a function $f : \{0,1\}^n \to \{0,1\}$, if the observation of the last bits of the work register equals $f(x)$ with probability at least $1-\varepsilon$, for every $x \in S$. Then $\mathsf{QQC}(f)$ (resp., $\mathsf{RQC}(f)$) is the minimum query complexity of quantum (resp., randomized) query algorithms that ε_0-compute f, where ε_0 is a fixed positive constant no greater than $\frac{1}{3}$.

2.3 Communication complexity

Communication complexity is a model of computation widely used to prove lower bounds in various models of computation. Here we will appeal to this model for lower bounds for circuit depth. We also consider one-way communication complexity in Section 4.

Let X, Y, Z be finite sets, and $R \subseteq X \times Y \times Z$. In the communication game for R, Alice is given some $x \in X$, Bob is given some $y \in Y$ and their goal is to find some $z \in Z$ such that $(x, y, z) \in R$, if such a z exists. A communication protocol determines what message each player sends in each round, and by convention, Bob produces an output at the end of the protocol. The cost of a protocol is the total number of bits exchanged in the worst case, and the communication complexity of R, written $\mathsf{D}(R)$, is the minumum cost of a protocol computing R.

There are many variants of communication complexity, and we will also consider one-way communication complexity of boolean functions. In a one-way communication protocol, two players, A and B wish to compute the value of a two-argument function $f : X \times Y \to Z$. Player A receives an input $x \in X$, and

sends a message m to Player B. Player B receives an input $y \in Y$, as well as A's message m and should output the value of the function $f(x, y)$. The protocol is successful if B's output equals $f(x, y)$, for all x, y.

In the randomized model, a protocol is δ-correct if for all inputs x, y, the error probability on x, y is at most δ. The probability is taken over the random choices made by the players. $\mathsf{R}_\delta(R)$ is the minumum cost of a protocol computing R in this way.

In the distributional model, we consider deterministic protocols, together with a distribution of the inputs μ, and an error threshold δ. A distributional protocol is δ-correct if the probability taken over μ that the output differs from the function is at most δ. The *distributional communication complexity for* μ, $\mathsf{D}_{\delta,\mu}(f)$, is the maximum number of bits exchanged for the best δ-correct protocol for f when the input is chosen according to μ. The distributional complexity $\mathsf{D}_\delta(f)$ of f is the maximum, over all probability distributions μ on the inputs, of $\mathsf{D}_{\delta,\mu}(f)$.

2.4 Circuits and formulae

A Boolean formula over the standard basis $\{\vee, \wedge, \neg\}$ is a binary tree where each internal node is labeled with $\vee$ or $\wedge$, and each leaf is labeled with a literal, that is, a Boolean variable or its negation. The size of a formula is its number of leaves.

Definition 2. *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. The formula size of f, denoted $\mathsf{L}(f)$, is the size of the smallest formula which computes f. The formula depth of f, denoted $\mathsf{d}(f)$ is the minimum depth of a formula computing f.*

It is clear that $\mathsf{L}(f) \leq 2^{\mathsf{d}(f)}$. Spira has also shown that $\mathsf{d}(f) \leq O(\log \mathsf{L}(f))$ [Spi71].

Karchmer and Wigderson [KW88] give an elegant characterization of formula size and depth in terms of communication complexity.

Definition 3. *For any Boolean function f, the relation $R_f = \{(x, y, i) : f(x) = 0, f(y) = 1, x_i \neq y_i\}$.*

Theorem 1 (Karchmer-Wigderson). *For any Boolean function f, $\mathsf{d}(f) = \mathsf{D}(R_f)$.*

3 KI complexity, its variants, and applications

In order to prove a lower bound for a Boolean function f, consider two inputs that are mapped by f to different values. Then these two inputs must differ in some position and if the computation is correct, it must implicitly or explicitly have found one of these positions where the inputs differ. This is the principle which we will show how to exploit in this section, to obtain lower bounds in various models of computation.

3.1 Decision trees and KI complexity

Proposition 5. *Let f be a Boolean function, x, y be inputs such that $f(x) \neq f(y)$. Then* $\mathsf{DT}(f) \geq \min_{\alpha\in\{0,1\}^*} \max_{\substack{x,y \\ f(x)\neq f(y)}} \min_{i:x_i\neq y_i}\{\max\{2^{K(i|x,\alpha)}, 2^{K(i|y,\alpha)}\}\}$

Proof. Let T be a decision tree for f. If $f(x) \neq f(y)$, then the computation paths on x and y must diverge at some level of the decision tree. Let i be the variable queried at this level. Since the computation paths diverge at this point, $x_i \neq y_i$. So $\mathsf{K}(i|x,T) \leq log(depth(T))$ since it suffices to give an index into the depth of the tree, and similarly, $\mathsf{K}(i|y,T) \leq log(depth(T))$. Therefore, $\exists\alpha = T, \forall x, y : f(x) \neq f(y), \exists i, \mathsf{DT}(f) \geq \max\{2^{\mathsf{K}(i|x,T)}, 2^{\mathsf{K}(i|y,T)}\}$, which concludes the proof. □

Similar results hold for various models of computation, but with somewhat different combinations of the terms $K(i|x)$ and $K(i|y)$, for $f(x) \neq f(y)$ and $x_i \neq y_i$. We introduce a general definition that captures the known lower bounds in a common framework.

Definition 4. *Let $f : \{0,1\}^n \to \{0,1\}$. Let $\Lambda : \mathbb{R}^* \to \mathbb{R}$ (Λ takes an arbitrary number of real inputs, such as* max *or Σ, which we will take over all terms parameterized by i where $x_i \neq y_i$) and $\star : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ (where we sometimes use infix notation, e.g. $A \star B$). Define*

$$\mathsf{KI}^{\Lambda,\star}(f) = \min_{\alpha\in\{0,1\}^*} \max_{\substack{x,y \\ f(x)\neq f(y)}} \frac{1}{\Lambda_{i:x_i\neq y_i} 2^{-\mathsf{K}(i|x,\alpha)} \star 2^{-\mathsf{K}(i|y,\alpha)}}.$$

Reformulating Proposition 5 in terms of KI, we have

Proposition 6. $\mathsf{DT}(f) \geq \mathsf{KI}^{\max,\min}(f)$.

3.2 Randomized and quantum query complexity lower bounds

Proposition 6 can be extended to randomized and quantum query complexity. The intuition is the same, but one has to analyze the the contribution of making a "useful" query much more carefully, since in these models, a query can be made with some probability or some amplitude.

Theorem 2. *[LM04] Let $f : \{0,1\}^n \to \{0,1\}$.*

1. $\mathsf{QQC}(f) \geq \Omega(KI^{\Sigma,geom}(f))$ *where Σ denotes sum over i such that $x_i \neq y_i$ and geom is the geometric average: $geom(A,B) = \sqrt{A \cdot B}$.*
2. $\mathsf{RQC}(f) \geq \Omega(KI^{\Sigma,\min}(f))$.

The theorem is proved by analyzing the overall contribution of each query towards disinguishing pairs of inputs with different values. Roughly speaking, the sum appears as a result of considering progress over all input pairs x, y such that $f(x) \neq f(y)$. The $\star$ operation is not so easily explained but the difference can be attributed to the fact that in the quantum case we operate under the ℓ_2 norm whereas in the randomized case, the ℓ_1 norm is used.

It turns out that this lower bound on query complexity implies all so-called adversary techniques for proving lower bounds in quantum query complexity, including the quantum and randomized weighted methods [Amb03,Aar04] and the spectral method [BSS03].

To give an idea of why this is the case we give an proof of Ambainis' unweighted adversary method, which is given in terms of the combinatorial structure of the graph that represents pairs (edges) x, y such that $f(x) \neq f(y)$. This graph is thought of as containing the pairs of instances that are hard to distinguish. Furthermore, the pairs x, y that differ on some index i are those that a query to i can be helpful to distinguish x from y. Comparing the graph R with the subgraph R_i where the ith query is useful allows us to establish lower bounds on the number of queries required to distinguish all the pairs in R.

Theorem 3. *[Amb02,Aar04,LM04] Let $R \subseteq X \times Y$, be a relation on pairs of instances, where $X = f^{-1}(0)$ and $Y = f^{-1}(1)$, and let R_i be the restriction of R to pairs x, y for which $x_i \neq y_i$. Viewing the relation R as a bipartite graph, then if*

- *m is a lower bound on the degree of all $x \in X$,*
- *m' is a lower bound on the degree of all $y \in Y$,*
- *for any fixed $i, 1 \leq i \leq n$, the degree of any $x \in X$ in R_i is at most l,*
- *for any fixed $i, 1 \leq i \leq n$, the degree of any $y \in Y$ in R_i is at most l',*

then $\mathsf{QQC}(f) = \Omega\left(\sqrt{\frac{mm'}{ll'}}\right)$ *and* $\mathsf{RQC}(f) = \Omega\left(\max\{\frac{m}{l}, \frac{m'}{l'}\}\right)$.

Proof. We make the following observations.

1. $|R| \geq \max\{m|X|, m'|Y|\}$, so $\exists x, y\ \mathsf{K}(x, y) \geq \max\left(\log(m|X|), \log(m'|Y|)\right)$.
2. $\forall x \in X, \mathsf{K}(x) \leq \log(|X|)$ and $\mathsf{K}(y) \leq \log(|Y|)$, for all $y \in Y$.
3. $\forall x, y, i$ with $(x, y) \in R_i, \mathsf{K}(y|i, x) \leq \log(l)$ and similarly, $\mathsf{K}(x|i, y) \leq \log(l')$.

For any i with $x_i \neq y_i$, by Proposition 4,

$$\begin{aligned}\mathsf{K}(i|x) &\geq \mathsf{K}(x, y) - \mathsf{K}(x) - \mathsf{K}(y|i, x) + \mathsf{K}(i|x, y, \mathsf{K}(x, y)) \\ &\geq \log(m|X|) - \log(|X|) - \log(l) + \mathsf{K}(i|x, y, \mathsf{K}(x, y)) \\ &= \log(\tfrac{m}{l}) + \mathsf{K}(i|x, y, \mathsf{K}(x, y))\end{aligned}$$

The same proof works to show that $\mathsf{K}(i|y) \geq \log(\frac{m'}{l'}) + \mathsf{K}(i|x, y, \mathsf{K}(x, y))$. We can conclude by Theorem 2 and Kraft's inequality. □

3.3 Circuit depth and formula size

Another model where KI can be used to obtain lower bounds is boolean formulas.

We give a simple proof that KI gives a lower bound on circuit depth.

Theorem 4. *For any Boolean function f, $\mathsf{d}(f) \geq \mathsf{KI}^{\max,\cdot}(f)$.*

Proof. Let P be a protocol for R_f. Fix x, y with different values under f, and let T_A be a transcript of the messages sent from A to B, on input x, y. Similarly, let T_B be a transcript of the messages sent from B to A. Let i be the output of the protocol, therefore $x_i \neq y_i$. To print i given x, simulate P using x and T_B. To print i given y, simulate P using y and T_A. This shows that $\forall x, y : f(x) \neq f(y), \exists i : x_i \neq y_i, K(i|x, \alpha) + K(i|y, \alpha) \leq |T_A| + |T_B| \leq \mathsf{D}(R_f)$, where α is a description of A's and B's algorithms. The theorem then follows from Theorem 1. □

3.4 A few examples

We give a few elementary examples to demonstrate how the technique can be applied to specific functions. To apply the adversary method, we have to give a relation R of hard instances; however, when applying KI, it suffices to exhibit a single hard pair of inputs.

Example 1: OR The OR function is 0 on the all-0 input and 1 everywhere else. Consider inputs x, y of length n, where x is the all-0 string, and y is 0 everywhere except in bit i, where i is chosen so that $\mathsf{K}(i) \geq \log(n)$. (More exactly, for any α we choose i such that $\mathsf{K}(i|\alpha) \geq \log(n)$.) Such an i exists by incompressibility (Proposition 1). Therefore, by Theorems 2 and 4, and Proposition 6,

1. $\mathsf{DT}(OR) \geq \Omega(n)$,
2. $\mathsf{RQC}(OR) \geq \Omega(n)$,
3. $\mathsf{QQC}(OR) \geq \Omega(\sqrt{n})$,
4. $\mathsf{d}(OR) \geq \Omega(\log n)$.

Example 2: PARITY The parity function is defined as $\oplus(x) = \Sigma_i x_i \ (mod\ 2)$. Consider inputs x, y chosen as follows. Take x, i so that $\mathsf{K}(x, i) \geq n + \log(n)$, and let $y = x^i$ (x with the ith bit flipped). It is easy to show that $\mathsf{K}(i|x) \geq \log(n)$ and $\mathsf{K}(i|y) \geq \log(n)$.

1. $\mathsf{DT}(\oplus) \geq \Omega(n)$,
2. $\mathsf{RQC}(\oplus) \geq \Omega(n)$,
3. $\mathsf{QQC}(\oplus) \geq \Omega(n)$,
4. $\mathsf{d}(\oplus) \geq \Omega(\log n)$.

Several examples relating to graph properties are also given in [LM04].

4 Kolmogorov alternative to the min-max principle

Usually, lower bounds for randomized complexity are proven by first applying Yao's min-max principle, and proving a lower bounds in the distributional model where the algorithms are deterministic and the inputs are chosen at random according to some distribution. We propose an alternative to (or perhaps only a reformlation of) Yao's min-max principle, which makes use of Kolmogorov complexity. (To be precise, we only give an analogue of the "easy direction"

that is generally used for lower bounds.) We illustrate how it can be applied, by proving a very general statement about one-way communication complexity. In this case, the proof is somewhat simpler than the previous proof of Bar-Yossef, Jayram, Kumar and Sivakumar [BYJKS02] that used information theory.

4.1 Yao in the style of Kolmogorov

Yao's min-max principle consists in replacing randomness in the algorithm, with randomness in the inputs. Our approach is to replace randomness in the algorithm by a Kolmogorov random string, resulting in a deterministic algorithm. It remains to see that the errors made on this random string are not too many. This is what is proven in the following lemma. The lemma is stated for private coin communication complexity but a similar statement can be made for other models of computation.

We assume, without loss of generality, that the players use a random string r_A, r_B taken uniformly at random from finite sets R_A, R_B, and that this is the same distribution regardless of the players' inputs x, y.

Lemma 1. *Let $f : X \times Y \to Z$. Fix any δ-correct randomized communication complexity protocol P for f, and consider any subset of inputs $S \subseteq X \times Y$. Fix $(r_A^*, r_B^*) \in R_A \times R_B$ such that $C(r_A*, r_B^*|P, S) \geq \log(|R_A|) + \log(|R_B|)$. Then when the protocol is run using r_A^*, r_B^* as random choices, the output is incorrect on at most $2\delta|S|$ inputs in $|S|$.*

Proof. For any r_A, r_B, let $\tilde{S}$ represent the inputs on which the outcome of the protocol is incorrect, that is, $\tilde{S}_{r_A,r_B} = \{\tilde{x}, \tilde{y} \in S : P(\tilde{x}, \tilde{y}, r_A, r_B) \neq f(\tilde{x}, \tilde{y})\}$. Also define the set of "much-worse-than-average" random choices for inputs in S to be $\tilde{R} = \{r_A, r_B : |\tilde{S}_{r_A,r_B}| > 2\delta|S|\}$.

Because at most half the inputs can have more than double the average number of errors, $|\tilde{R}| \leq \frac{|R_A||R_B|}{2}$, therefore by incompressibility, $r_A^*, r_B^* \notin \tilde{R}$. (Otherwise, describe r_A^*, r_B^* by giving an index into the set $\tilde{R}$. using $\log(|R|) < \log(|R_A|) + \log(|R_B|)$ bits, a contradiction.) Therefore $|\tilde{S}_{r_A^*,r_B^*}| \leq 2\delta|S|$.

□

4.2 Shatter coefficients lower bound

To give an example of how this method is applied, we give a proof of a general theorem on one-way communication complexity.

First we define VC dimension and its generalization, shatter coefficients. Let F be a set of strings of length n, and I be a set of indices, $I \subseteq [n]$, $I = i_1, \cdots, i_{|I|}$. For any string $x = x_0, \cdots x_{n-1}$ of length n, $x|_I$ denotes the string $x_{i_1} \cdots x_{i_{|I|}}$. Likewise, $F|_I = \{x|_I : x \in F\}$ A set of strings F is *shattered* by a set of indices I if $F|_I$ is the set of all possible strings of length $|I|$. The *VC dimension* of F, denoted $VC(F)$, is the size of the largest I that shatters F.

The lth shatter coefficient of F (for any $l > VC(F)$), denoted $SC(F, l)$ is the maximum, over all I of size l, of $|F|_I|$. Let $F' \subseteq F$ be a subset of F for which

$F'|_I$ takes on this maximal number of distinct values. We say that $F' \times I$ is a witness for $SC(F, l)$.

We give a new proof of a well-known result about one-way communication complexity. Recall that in this model, Alice sends one message to Bob and Bob produces the output. We use the superscript $A \to B$ to specify this model.

Theorem 5 ([KNR99,BYJKS02]). *For every function $f : X \times Y \to \{0,1\}$, every $l \geq VC(f)$, and every $\delta > 0$, $\mathsf{R}_\delta^{A\to B}(f) \geq \log(SC(f|_X, l)) - lH_2(2\delta)$, where $H_2(p) = -p\log(p) - (1-p)\log(1-p)$.*

Proof. Let $row_f(x, Y') = f(x, y_1) \cdots f(x, y_{|Y'|})$ be the string of consecutive values of f when x is fixed, where $Y' = \{y_1, \ldots y_{|Y'|}\}$. We denote by $f|_{X,Y}$ the set of strings $\{row_f(x, Y) : x \in X\}$. Let $S' = X' \times Y'$ be a witness for $SC(F, l)$ where $F = f|_{X,Y}$. Fix $x^* \in X', r_A^* \in R_A, r_B^* \in R_B$ with $C(x^*, r_A^*, r_B^*|P, S') \geq \log(|X'|) + \log(|R_A|) + \log(|R_B|)$ and let $S = \{x^*\} \times Y'$. Notice that $|S| = l$. By Lemma 1, when the protocol is run using r_A^*, r_B^* as random choices, the output is incorrect on at most $2\delta|S|$ inputs in $|S|$. To correct these errors we can just describe their location. This requires $\log(\binom{l}{2\delta l}) \approx l \cdot H_2(2\delta)$ additional bits.

All $\{row_f(x, Y') : x \in X'\}$ are unique, so x^* is uniquely determined within X' by $row_f(x, Y')$. This allows us to conclude that

$$\begin{aligned} \log(SC(f|_X, l)) &\leq C(x^*|P, r_A^*, r_B^*) \\ &\leq C(row_f(x^*, Y')|P, r_A^*, r_B^*) \\ &\leq \mathsf{R}_\delta^{A\to B}(f) + lH_2(2\delta). \end{aligned}$$

□

5 Concluding remarks

We have presented two different frameworks based on Kolmogorov complexity in which many lower bound techniques can be expressed. One might naturally ask what other models of computation these techniques can be applied to. One consequence of studying the KI lower bounds is that it brings to light the shared limitations of these techniques (see for example [LLS05]. Hopefully, understanding these limitations better will be a first step towards breaking the current lower bound barriers.

In the case of the min-max proofs using Kolmogorov complexity, it turns out in many cases that after rewriting the proofs in terms of Kolmogorov complexity, one can the remove Kolmogorov complexity entirely. An important role of Kolmogorov complexity is that the intuition it provides to help highlight the essential parts of the argument.

Acknowledgments

Many thanks are due to Marc Kaplan, Troy Lee, Frédéric Magniez, and Mario Szegedy, for many enlightening discussions on these results. Special thanks to Troy Lee for permission to include the unpublished results of Section 4.

References

[Aar04] S. Aaronson. Lower bounds for local search by quantum arguments lower bounds for local search by quantum arguments. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 465–474, 2004.

[Amb02] A. Ambainis. Quantum lower bounds by quantum arguments. *Journal of Computer and System Sciences*, 64:750–767, 2002.

[Amb03] A. Ambainis. Polynomial degree vs. quantum query complexity. In *Proceedings of 44th IEEE Symposium on Foundations of Computer Science*, pages 230–239, 2003.

[BSS03] H. Barnum, M. Saks, and M. Szegedy. Quantum decision trees and semidefinite programming. In *Proceedings of the 18th IEEE Conference on Computational Complexity*, pages 179–193, 2003.

[BYJKS02] Z. Bar-Yossef, T. S. Jayram, R. Kumar, and D. Sivakumar. Information theory methods in communication complexity. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity*, pages 93–102, 2002.

[KNR99] I. Kremer, N. Nisan, and D. Ron. On randomized one-round communication complexity. *Computational Complexity*, 8(1):21–49, 1999.

[KW88] M. Karchmer and A. Wigderson. Monotone connectivity circuits require super-logarithmic depth. In *Proceedings of the 20th STOC*, pages 539–550, 1988.

[LL04] S. Laplante and T. Lee. A few short lower bounds in one-way communication complexity. Unpublished manuscript, July 2004.

[LLS05] S. Laplante, T. Lee, and M. Szegedy. The quantum adversary method and classical formula size lower bounds. In *Proceedings of the Twentieth Annual IEEE Conference on Computational Complexity*, pages 76–90, 2005.

[LM04] S. Laplante and F. Magniez. Lower bounds for randomized and quantum query complexity using kolmogorov arguments. In *Proceedings of the Nineteenth Annual IEEE Conference on Computational Complexity*, pages 294–304, 2004.

[Spi71] P. Spira. On time-hardware complexity tradeoffs for Boolean functions. In *Proceedings of the 4th Hawaii Symposium on System Sciences*, pages 525–527. Western Periodicals Company, North Hollywood, 1971.