The Complexity of Complexity

Eric Allender*

November 14, 2016

Abstract

Given a string, what is its complexity? We survey what is known about the computational complexity of this problem, and describe several open questions.

1 Introduction

There are many different ways to define the "complexity" of a string. Indeed, if you look up "complexity" in the index of Downey and Hirschfeldt [23] you'll find that the list of entries for this single item extends over more than two full pages. And this list barely mentions any of the *computable* notions of complexity that constitute much of the focus of this article.

All of these notions of "complexity" seem to share some common characteristics. First: determining complexity is hard; even among those notions that happen to be computable, there does not seem to be any widely-studied notion of complexity where the complexity of a string is *efficiently* computable. Secondly: there is a paucity of techniques that have been developed, in order to *show* that it is a computationally hard problem to determine the complexity of a string. Indeed, since complexity theory almost always resorts to *reducibility* in order to show that certain problems are hard, we are led to the problem of studying the class of problems that are reducible to the task of computing the "complexity" of a string, for different notions of "complexity", and under different types of reducibility.

The goal of this article is to give an overview of what is known about this topic (including some new developments that have not yet been published), and to suggest several directions for future work.

2 Preliminaries and Ancient History

One advantage of writing a high-level survey, is that the author is permitted the luxury of providing only a high-level overview of various important definitions, leaving the details to be found in the cited references. So let us indulge in this luxury.

*Department of Computer Science, Rutgers University, Piscataway, NJ

Kolmogorov complexity (in all of its many variations) provides the best tools for quantifying how much information a string x contains. In this survey, the only two versions of Kolmogorov complexity that will be considered are the plain complexity $C(x)$ and the prefix-free complexity $K(x)$. But the reader will be reminded at key moments that the measures C and K actually represent an infinite collection of measures C_U and K_U, indexed by the choice of "universal" Turing machine that is used in defining Kolmogorov complexity. For more formal definitions and background, see [23].

We will be especially concerned with the set of *random* strings, and with the *overgraph* of these complexity functions:

Definition 2.1. *Let U be a universal Turing machine.*

- $R_{K_U} = \{x : K_U(x) \geq |x|\}$. *This is the set of K-random strings.*
- $R_{C_U} = \{x : C_U(x) \geq |x|\}$. *This is the set of C-random strings.*
- $O_{K_U} = \{(x,k) : K_U(x) \leq k\}$. *This is the overgraph for K.*
- $O_{C_U} = \{(x,k) : C_U(x) \leq k\}$. *This is the overgraph for C.*

As usual, we suppress the "U" when the choice of universal machine is not important. We will be considering several other measures μ of the complexity of strings. For any such measure μ, we will refer to $R_\mu = \{x : \mu(x) \geq |x|\}$, and similarly O_μ is $\{(x,k) : \mu(x) \leq k\}$.

A binary string x of length N can also be viewed as a representation of a function $f_x : [N] \to \{0,1\}$. When $N = 2^n$ is a power of two, then it is customary to view f_x as the truth table of an n-ary Boolean function, in which case computational complexity theory provides a different suite of definitions to describe the complexity of x, such as circuit size $\mathsf{CSize}(x)$, branching program size $\mathsf{BPSize}(x)$, and formula size $\mathsf{FSize}(x)$.[1] For more background and definitions on circuits, formulas, and branching programs, see [50].

Definition 2.2. *The* Minimum Circuit Size Problem MCSP *is the overgraph of the* CSize *function:* $\mathsf{MCSP} = \{(x,k) : \mathsf{CSize}(x) \leq k\}$.

The study of MCSP dates back to the 1960's ([49], see also the discussion in [15]). More recently, Kabanets and Cai focused attention on MCSP [32] in connection with its relation to the "natural proofs" framework of Razborov and Rudich [43]. In Section 3 we will review what is known about MCSP as a candidate "NP-intermediate" problem: a problem in $\mathsf{NP} - \mathsf{P}$ that is not NP-complete.

Although it was recognized in the 1960's in the Soviet mathematical community that MCSP had somewhat the same flavor as a time-bounded notion of Kolmogorov complexity [49], a formal connection was not established until the 1990's, by way of a modification of a time-bounded version of Kolmogorov complexity that had been introduced earlier by Levin.

[1] These measures can be extended to *all* strings by appending bits to obtain a string whose length is a power of 2 [8].

The "standard" way to define time-bounded Kolmogorov complexity is to define a measure such as $C^t_U(x) = \min\{|d| : U(d) = x \text{ in at most } t(|x|) \text{ steps}\}$. In contrast, Levin's definition [36] of the the measure Kt combines both the running time and the description length into a single number: $\mathrm{Kt}(x) = \min\{|d| + \log t : U(d) = x \text{ in at most } t \text{ steps}\}$. Levin's motivation in formulating this definition came from its utility in defining optimal search strategies for deterministic simulations of nondeterministic computations.

The connection between CSize and time-bounded Kolmogorov complexity provided by [8] is the result of replacing "$\log t$" by "t" in the definition of Kt, to obtain a new measure KT. In order to accommodate sublinear running-times as part of this definition (so that $\mathrm{KT}(x)$ can be much less than $|x|$), the technical description of what it means for $U(d) = x$ is changed. Now U is given random access to the description d, and – given d and i – U determines the i-th bit of x. (For formal definitions, see [8].) This time-bounded measure KT is of interest because:

- $\mathrm{KT}(x)$ is polynomially-related to $\mathsf{CSize}(x)$ [8]. (In contrast, no such relation is known for other polynomial-time-bounded variants of Kolmogorov complexity, such as the "standard" version C^{n^2} mentioned above.)

- By providing the universal machine U with an oracle, variants of other notions of Kolmogorov complexity are obtained. For instance, with an oracle A that is complete for deterministic exponential time, $KT^A(x)$ is essentially the same as $\mathrm{Kt}(x)$. Similarly, if A is the halting problem (or any problem complete for the c.e. sets under linear-time reductions), then $\mathrm{KT}^A(x)$ and $C(x)$ are linearly-related [8].

- The connection between MCSP and KT carries over also to the relativized setting. One can define $\mathsf{CSize}^A(x)$ in terms of circuits with "oracle gates", and thus obtain the language MCSP^A. $\mathrm{KT}^A(x)$ is polynomially-related to $\mathsf{CSize}^A(x)$. This has been of interest primarily in the case of $A = \mathsf{QBF}$, the standard complete set for PSPACE; see Section 3.

Other resource-bounded Kolmogorov complexity measures in a similar vein were presented in [15]: KF is polynomially-related to FSize, and KB is polynomially-related to BPSize. Note in this regard the following sequence of inequalities:

$$C(x) \leq K(x) \leq \mathrm{Kt}(x) \leq \mathrm{KT}(x) \leq \mathrm{KB}(x) \leq \mathrm{KF}(x)$$

(where, as usual, all inequalities are modulo an additive $O(1)$ term). In the other direction, $K(x) \leq C(x) + O(\log C(x))$, whereas there is no computable function f such that $\mathrm{Kt}(x) \leq f(K(x))$. It is conjectured that the other measures can differ exponentially from their "neighbors", although it is an open question even whether $\mathrm{KF}(x) \leq \mathrm{Kt}(x)^2$. (In fact, this question is equivalent to a question about the nonuniform circuit complexity of the deterministic exponential time class EXP [15].) Related measures have been introduced in [8, 2, 15] in order to connect (deterministic and nondeterministic) space complexity, nondeterministic time complexity, and "distinguishing" complexity.

We will also need to mention one other family of "complexity" measures. Consider a $2^n \times 2^n$ matrix M with rows indexed by n-bit strings i that are viewed as possible inputs to a player

in a 2-person game, named Alice. The columns are similarly indexed by strings j that are the possible inputs for another player, named Bob. The (deterministic) communication complexity of M, denoted $\mathsf{DCC}(M)$, is the minimal number of bits that Alice and Bob must communicate with each other in order for them to know $M(i,j)$. Thus if x is any string of length 2^{2n}, we will define $\mathsf{DCC}(x)$ to be equal to $\mathsf{DCC}(M)$, where we interpret x as a $2^n \times 2^n$ matrix M. There is also a nondeterministic variant of communication complexity, which we will denote NCC. The only fact that we will need regarding NCC is that, for all x, $\mathsf{NCC}(x) \leq \mathsf{DCC}(x)$. For more about communication complexity, see [34].

The reader will encounter various complexity classes in these pages, beyond the familiar P, NP, and PSPACE. There are three versions of probabilistic polynomial time ($\mathsf{ZPP} = \mathsf{RP} \cap \mathsf{coRP} \subseteq \mathsf{RP} \subseteq \mathsf{BPP}$), and the exponential-time analogs of P and NP (EXP and NEXP), as well as small subclasses of P defined in terms of classes of circuits ($\mathsf{AC}^0 \subseteq \mathsf{TC}^0$). We refer the reader to [17, 50] for more background on these topics.

3 The Minimum Circuit Size Problem

There are few "natural" problems that have as strong a claim to NP-Intermediate status, as MCSP. To be sure, there are various problems that are widely used in cryptographic applications, such as factoring and the discrete logarithm problem, that are widely suspected (or hoped) not to have polynomial-time algorithms – but if it turns out they are easy, then they don't bring a large class of other problems with them into P. In contrast, if MCSP is easy, then there are large ramifications.

3.1 Reductions to MCSP

Kabanets and Cai showed that if MCSP is in $\mathsf{P/poly}$, then every potential cryptographically-secure one-way function is easy to invert on a substantial portion of its range [32]. Subsequently, it was shown that every language in the complexity class SZK lies in $\mathsf{BPP}^{\mathsf{MCSP}}$ [9], thereby providing strong evidence, based on the structure of complexity classes, that MCSP lies outside of P.[2] "SZK" stands for "Statistical Zero Knowledge", and it contains the graph isomorphism problem, as well as a great many problems whose presumed intractability is essential for the security of well-studied cryptosystems. For this survey, the reader will not need to know much about zero-knowledge interactive proofs, which are used in order to define SZK. It will suffice to know that every language in SZK is contained in $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$, and thus this class is in some sense "close" to $\mathsf{NP} \cap \mathsf{coNP}$. It will also be important to know that SZK is best defined in terms of "promise problems", and that some of these promise problems are in fact *complete* for SZK.

A *promise problem* consists of two disjoint sets Y and N, called the YES-instances and the NO-instances, respectively. A *solution* to the promise problem (Y, N) is any language that

[2] In spite of this "strong evidence", the only *unconditional* lower bound known for R_{KT} and MCSP is that neither is in AC^0 [8]. It is not even known whether these problems lie in $\mathsf{AC}^0[2]$ (that is, AC^0 augmented with parity gates).

contains Y and is disjoint from N. When we say that MCSP is hard for SZK under BPP reductions, we mean that, for every promise problem in SZK, there is a probabilistic oracle Turing machine M that accepts with very high probability on all of the YES-instances and rejects with very high probability on all of the NO-instances, but M might have acceptance probability close to 1/2 on other instances.

In order to establish that MCSP is hard for SZK, [9] presents a BPP reduction from the standard complete problem for SZK, called SD, for "*Statistical Difference*". The input to SD consists of a pair of circuits (C, D) defining probability distributions. The YES instances consist of pairs (C, D) such that these probability distributions are very close, and the NO instances consist of pairs for which the probability distributions are quite far apart.

There are a few things to mention about this reduction:

- The same proof establishes that the set of KT-random strings R_{KT} is also hard for SZK, as well as the overgraph of the KT function O_{KT}. In fact, until very recently, *every* efficient reduction to MCSP^A or O_{KT^A} for any oracle A carried over to the more restrictive problem R_{KT^A}. (Note that, for every measure μ, $\overline{R_\mu} \leq^{\mathrm{p}}_{\mathrm{m}} O_\mu$.) This is because, until very recently, all such reductions have proceeded by using derandomization techniques, using R_{KT^A} as a statistical test to foil pseudorandom generators.

- Motivated by the fact that SZK is a class of promise problems, one can define a promise problem related to MCSP, such as the problem Gap-MCSP with YES instances consisting of $\{x : \mathrm{KT}(x) \leq \sqrt{|x|}\}$ and NO instances consisting of $\{x : \mathrm{KT}(x) \geq |x|/2\}$. The proof in [9] establishes that every promise problem in SZK is in $\mathsf{Promise\text{-}BPP}^A$ for every set A that is a solution to Gap-MCSP.

For certain problems in SZK, more restrictive reductions to MCSP are known. It is shown in [8, 45] that factoring and Discrete Log are in $\mathsf{ZPP}^{\mathsf{MCSP}}$, and Graph Isomorphism lies in $\mathsf{RP}^{\mathsf{MCSP}}$ [9]. As with the reductions mentioned above, these also carry over to O_{KT} and R_{KT}, and they are proved using the same suite of techniques.

Thus it is of interest that a fundamentally different type of reduction is given in [13], showing that the Graph Automorphism problem is in $\mathsf{ZPP}^{O_{\mathrm{KT}}}$. It is still not known whether Graph Automorphism is in $\mathsf{ZPP}^{\mathsf{MCSP}}$ or $\mathsf{ZPP}^{R_{\mathrm{KT}}}$.

Open Question 3.1. *Is Graph Automorphism in* $\mathsf{ZPP}^{\mathsf{MCSP}}$ *and/or in* $\mathsf{ZPP}^{R_{\mathrm{KT}}}$*? Until the appearance of [13], the problems* O_{KT} *and* R_{KT} *had been viewed as convenient proxies for* MCSP*, such that a theorem that was proved for one of the problems would hold for all of them. Similarly, the different versions of* R_{KT_U} *(for different universal Turing machines* U*) had been viewed as being more-or-less equivalent, and different versions of* MCSP *(say, with "size" defined in terms of number of gates instead of number of wires, or with a slightly different set of allowable gates, etc) were viewed as being more-or-less the same set. We do not know of any theorem that can be proved for one version of* R_{KT_U} *and not for another, and we do not know of any theorem that holds for one version of* MCSP *and not for another – but we also do not know of any efficient reduction between these different versions. [13] provides the first example of a reduction that is known to hold for* O_{KT} *but does not carry over to these related problems. Is this merely a*

shortcoming of the proof as presented in [13], or is there really a significant difference in the complexity of these problems?

It is shown in [15] that the problem of factoring Blum integers is in $\mathsf{ZPP}^{R_{\mathrm{KB}}}$ and in $\mathsf{ZPP}^{R_{\mathrm{KF}}}$, and it is shown in [35] that factoring Blum integers is also in $\mathsf{ZPP}^{R_{\mathrm{DCC}}}$. (A number x is a Blum integer if $x = pq$ for two primes p and q such that p and q are both congruent to 3 mod 4. Factoring Blum integers is generally considered to be roughly as difficult as the general factoring problem.)

Open Question 3.2. *Is there a significant subclass of* SZK *that reduces to* R_{KF}*? In particular, if we consider the restriction of the standard* SZK*-complete problem* SD*, where instead of the input being a pair of* circuits (C, D)*, the input is a pair of* formulae (C, D) *does this restricted problem reduce to* R_{KF}*? And does this restricted problem capture many of the natural problems that are known to lie in* SZK*? (Related questions can be asked about* R_{KB}*.)*

Branching programs are restricted circuits, and formulae are restricted branching programs. This is the best explanation that we have, for the facts that we can currently reduce fewer problems to O_{FSize} than to O_{BPSize}, and we can currently reduce fewer problems to O_{BPSize} than to MCSP. But perhaps this intuition is not valid at all. It is known that if we further restrict the formula size problem O_{FSize} to formulae in disjunctive normal form, then the resulting problem is NP-complete under $\leq^{\mathrm{p}}_{\mathrm{m}}$ reductions [38]. (See also [16].) Contrariwise, the problem of computing nondeterministic communication complexity (a *more* powerful model than DCC) is also NP-complete [41, 37]. This might lead the reader to wonder whether MCSP and these other related problems are not *all* NP-complete. What reason is there to believe that we cannot reduce all of NP to problems such as MCSP? This is the topic that is addressed in the next section.

3.2 Complete, or Not Complete? That is the question.

Although there is currently no strong evidence that MCSP is *not* NP-complete, there *is* a great deal of evidence that no proof of NP-completeness will be found anytime soon.

In order to present this evidence, let us briefly recall some of the more common types of reductions. The reader is probably familiar with polynomial-time and logspace many-one reducibility, denoted by $\leq^{\mathrm{p}}_{\mathrm{m}}$ and $\leq^{\log}_{\mathrm{m}}$, respectively. With very few exceptions [1], problems that are known to be complete for NP and other complexity classes under $\leq^{\mathrm{p}}_{\mathrm{m}}$ and $\leq^{\log}_{\mathrm{m}}$ reductions are even complete under reductions computed by AC^0 circuits. Most of the important NP-complete problems are also complete under another type of restrictive reducibility: sublinear-time reductions. For a time bound $t(n) < n$, we say that $A \leq^t_m B$ if there is a polynomial-time computable function f such that for all $x, x \in A \Leftrightarrow f(x) \in B$ where in addition, the function that maps (x, i) to the i-th bit of $f(x)$ is computed in time $t(|x|)$ by a machine that has random access to the bits of x. Those reductions are all *more* restrictive than $\leq^{\mathrm{p}}_{\mathrm{m}}$ reductions. We also need to mention the *less* restrictive notion known as polynomial-time truth-table reductions $\leq^{\mathrm{p}}_{\mathrm{tt}}$, also known as *nonadaptive* reductions.

Table 1 presents information about the consequences that will follow if MCSP is NP-complete (or even if it is hard for certain subclasses of NP). The table is incomplete (since it does not mention the influential theorems of Kabanets and Cai [32] describing various consequences if MCSP were complete under a certain restricted type of $\leq^{\mathrm{p}}_{\mathrm{m}}$ reduction). It also fails to adequately give credit to all of the papers that have contributed to this line of work, since – for example – some of the important contributions of [40] have subsequently been slightly improved [30, 14]. But one thing should jump out at the reader from Table 1: All of the conditions listed in Column 3 (with the exception of "FALSE") are widely believed to be true, although they all seem to be far beyond the reach of current proof techniques.

Table 1: Summary of what is known about the consequences of MCSP being hard for NP under different types of reducibility. If MCSP is hard for the class in Column 1 under the reducibility shown in Column 2, then the consequence in Column 3 follows.

class $\mathcal{C}$	reductions $\mathcal{R}$	statement $\mathcal{S}$	Reference
TC^0	$\leq_m^{n^{1/3}}$	FALSE	[40]
TC^0	$\leq^{\mathsf{AC}^0}_{\mathrm{m}}$	LTH[3] $\not\subseteq$ io$-\mathsf{SIZE}[2^{\Omega(n)}]$ and $\mathsf{P} = \mathsf{BPP}$	[14, 40]
TC^0	$\leq^{\mathsf{AC}^0}_{\mathrm{m}}$	$\mathsf{NP} \not\subseteq \mathsf{P/poly}$	[14]
P	$\leq^{\log}_{\mathrm{m}}$	$\mathsf{PSPACE} \neq \mathsf{P}$	[14]
NP	$\leq^{\log}_{\mathrm{m}}$	$\mathsf{PSPACE} \neq \mathsf{ZPP}$	[40]
NP	$\leq^{\mathrm{P}}_{\mathrm{tt}}$	$\mathsf{EXP} \neq \mathsf{ZPP}$	[30]

The case in favor of MCSP being an NP-intermediate problem would be stronger if there were some *unlikely* consequences that were known to follow if MCSP were NP-complete. Some indirect evidence of this sort is available, if we consider relativized versions of MCSP and KT, such as $\mathsf{MCSP}^{\mathsf{QBF}}$ and $\mathrm{KT}^{\mathsf{QBF}}$. This is explained below.

Unlike R_{KT} and MCSP, for which no completeness results are known, $\mathsf{MCSP}^{\mathsf{QBF}}$ and $R_{\mathrm{KT}^{\mathsf{QBF}}}$ are both complete for PSPACE under ZPP reductions [8]. The analogous problems MCSP^E and R_{Kt} (and also $R_{K^{2^{n^2}}}$) are complete for EXP under P/poly-Turing reductions and NP-Turing reductions [8] (where E can be any standard complete problem for $\mathsf{DTIME}(2^{O(n)})$). However, it is rather unlikely that these are complete under *deterministic* (uniform) reductions, as is highlighted in Table 2.

The main lesson from Table 2 is that even problems that appear much harder than MCSP (such as the PSPACE-complete problem $\mathsf{MCSP}^{\mathsf{QBF}}$, and the EXP-complete problem MCSP^E) cannot be hard for NP unless unlikely consequences follow. However, this does still not imply that MCSP itself is unlikely to be NP-hard, since we know of no *deterministic* reduction from MCSP to the (apparently harder) problems $\mathsf{MCSP}^{\mathsf{QBF}}$ and MCSP^E.

Although it might seem intuitively clear that MCSP must be no harder than MCSP^A, this intuition is suspect. Hirahara and Watanabe have shown that, if $\mathsf{MCSP} \notin \mathsf{P}$, then there is an oracle A such that $\mathsf{MCSP} \notin \mathsf{P}^{\mathsf{MCSP}^A}$ [28]. In the same paper, they consider problems that are

[3] LTH is the linear-time analog of the polynomial hierarchy. Problems in LTH are accepted by alternating Turing machines that make only $O(1)$ alternations and run for linear time.

Table 2: Summary of what is known about the consequences of $\mathsf{MCSP}^{\mathsf{QBF}}$ and $R_{\mathrm{KT}^{\mathsf{QBF}}}$ (and related problems in EXP) being hard for various classes (under different types of reducibility). If the problem in the first column is hard for the class in Column 2 under the reducibility shown in Column 3, then the consequence in Column 4 follows.

problem	class $\mathcal{C}$	reductions $\mathcal{R}$	statement $\mathcal{S}$	Reference
$\mathsf{MCSP}^{\mathsf{QBF}}$	TC^0	$\leq_{\mathrm{m}}^{\mathsf{AC}^0}$	FALSE[4]	[14]
$\mathsf{MCSP}^{\mathsf{QBF}}$	P	$\leq_{\mathrm{m}}^{\log}$	$\mathsf{EXP} = \mathsf{PSPACE}$	[14]
$\mathsf{MCSP}^{\mathsf{QBF}}$	NP	$\leq_{\mathrm{m}}^{\log}$	$\mathsf{NEXP} = \mathsf{PSPACE}$	[14]
$\mathsf{MCSP}^{\mathsf{QBF}}$	PSPACE	$\leq_{\mathrm{m}}^{\log}$	FALSE	[14]
$R_{\mathrm{KT}^{\mathsf{QBF}}}$	PSPACE	$\leq_{\mathrm{T}}^{\log}$	FALSE	[8]
MCSP^{E}	NP	$\leq_{\mathrm{m}}^{\mathrm{p}}$	$\mathsf{NEXP} = \mathsf{EXP}$	[14]
R_{Kt}	EXP	$\leq_{\mathrm{tt}}^{\mathrm{p}}$	FALSE	[8]
$R_{K^{2^{n^2}}}$	EXP	$\leq_{\mathrm{T}}^{\mathrm{p}}$	FALSE	[19]

"oracle-independent" reducible to MCSP by *probabilistic* reductions that make only one query. (All known reductions to MCSP – other than the identity reduction of MCSP to itself – are "oracle-independent" reductions in the sense of [28].) They show that all such problems lie in the complexity class $\mathsf{AM} \cap \mathsf{coAM}$.

Open Question 3.3. *Could the* SZK *lower bound on the complexity of* MCSP *be tight? For instance, could* $\mathsf{Gap\text{-}MCSP}$ *lie in* SZK*? The results of [28] are intriguing here, since every promise problem in* SZK *has a solution in* $\mathsf{AM} \cap \mathsf{coAM} \subseteq \mathsf{NP/poly} \cap \mathsf{coNP/poly}$*. It would be very interesting to place* MCSP *or* $\overline{R_{\mathrm{KT}}}$ *in any subclass of* NP*, but we seem quite far from this goal.*

Open Question 3.4. *Contrariwise, might* MCSP *be complete for* NP *under* $\mathsf{P/poly}$ *reductions (in the same way that the corresponding problems are complete for* PSPACE *and* EXP *under* $\mathsf{P/poly}$ *reductions)? Or might it lie in the high hierarchy of [46]? If so, then it would be "nearly"* NP*-complete.*

Open Question 3.5. *Can one show unconditionally that* MCSP *(or* $\overline{R_{\mathrm{KT}}}$*) is not complete for* NP *under* $\leq_{\mathrm{m}}^{\mathsf{AC}^0}$*? Or can one derive some* unlikely *consequences from these sets being complete? (Some related questions are discussed in [4].)*

Open Question 3.6. *There are many intriguing questions concerning the complexity of the set of Levin-random strings,* R_{Kt}*. Although this set is complete for* EXP *(under* $\mathsf{P/poly}$ *and* NP *reductions), it is not known whether* R_{Kt} *is in* P*. Can this be resolved? Or would it imply the resolution of some long-standing open problem in complexity? Also, it is known that* R_{Kt} *is in* ZPP *if and only if* $\mathsf{EXP} = \mathsf{ZPP}$ *[8]. This is exactly the conclusion one would obtain if* R_{Kt} *were complete for* EXP *under* ZPP *reductions – and yet it remains unknown whether* R_{Kt} *is complete under this type of reducibility.*

[4] It is not explicitly stated in [14] that this consequence is FALSE, but it is stated that, under this condition, $\mathsf{NP}^{\mathsf{QBF}} \not\subseteq \mathsf{P}^{\mathsf{QBF}}/\mathsf{poly}$, which is equivalent to $\mathsf{PSPACE} \not\subseteq \mathsf{PSPACE/poly}$, which is, of course, false.

3.3 Relationships Among Measures

It is believed that most of the measures mentioned in this section are not polynomially-related to each other. In fact, a large table is presented in [15], showing that, for most pairs of measures $\mu 1$ and $\mu 2$ mentioned in this section, the question of whether $\mu 1$ and $\mu 2$ are polynomially-related is equivalent to some well-known open question in complexity theory.

However, there are some noteworthy relationships that should be mentioned, regarding DCC and FSize. Kushilevitz and Weinreb [35] showed that there is a polynomial-time routine that, given a bitstring x of length $N = 2^{2n}$, will produce a string M of length 2^{4n}, which can be interpreted as the matrix for a communication game, called ENE_x with the property that $\mathsf{DCC}(ENE_x) - n - 1 \leq \mathsf{FDepth}(f_x) \leq .886 \cdot \mathsf{DCC}(ENE_x) + n + O(\log n)$, where "$\mathsf{FDepth}$" measures the minimal formula depth of a function. (This is a consequence of the following results of [35]: Claim 3.3(iv) yields the first inequality, while the second inequality follows from Claim 3.3(iii) and Theorem 3.6.) Since $\mathsf{FSize}(f_x) \leq 2^{\mathsf{FDepth}(f_x)} \leq \mathsf{FSize}(f_x)^{1.71}$ [48], we have $\mathsf{FSize}(f_x) \leq 2^n (2^{.886 \cdot \mathsf{DCC}(ENE_x)}) n^{O(1)} \leq 2^{1.886n} \mathsf{FSize}(f_x)^{1.515} n^{O(1)}$. This appears to be a very poor approximation, but since f_x is a Boolean function on $2n$ variables, even a factor of $2^{1.886n}$ is not overwhelming, and this still means that, with an oracle for the overgraph of DCC, O_{DCC}, we can distinguish between those strings x where $\mathsf{FSize}(f_x)$ is very large, and those x where $\mathsf{FSize}(f_x)$ is very small; this is exploited in [35]. Some additional results relating DCC to problems such as MCSP have been explored by Raviv [42].

Open Question 3.7. *Is there a significant subclass of* SZK *that reduces to* O_{DCC}*? Are there more connections between* DCC *and the other complexity measures studied here?*

Shallit and Wang introduced a complexity measure on strings based on finite-state automata, called Automatic Complexity [47]. A related measure based on nondeterministic finite automata has been introduced by Hyde and Kjoos-Hanssen [31].

Open Question 3.8. *Are there any interesting relationships between the Automatic Complexity measure of Shallit and Wang, and any of the other measures mentioned in this section? Is there any evidence that Automatic Complexity (or the related measure of [31]) is computationally intractable?*

4 Complexity Classes and Noncomputable Complexity Measures

Up to now, this article has focused primarily on decidable problems such as MCSP and R_{KT^A} for *computable* oracles A. In this section, we survey some intriguing connections between computational complexity theory and *noncomputable* measures such as C and K.

As discussed in the previous section, there has not been much success using *deterministic* reductions to exploit the power of problems such as MCSP; $\mathsf{P}^{\mathsf{MCSP}}$ is not known to contain any problems of interest, other than MCSP itself.

The situation is different for reductions to R_C and R_K. (For ease of exposition, let "R" stand for either of these sets for the time being; the following results hold, no matter which of these measures is used.) Although there are some negative results, showing that EXP and problems outside of $\mathsf{P/poly}$ cannot be reduced to R using restricted $\leq^{\mathrm{p}}_{\mathrm{T}}$ reductions (such as disjunctive truth-table reductions or reductions that make a limited number of queries) [7, 29], there are also two striking positive results:

Theorem 4.1. *Let R denote either R_C or R_K. Then*

- $\mathsf{PSPACE} \subseteq \mathsf{P}^R$. *[8].*
- $\mathsf{BPP} \subseteq \mathsf{P}^R_{tt}$. *[18] (where P^R_{tt} is the class of problems reducible to R via $\leq^{\mathrm{p}}_{\mathrm{tt}}$ reductions.)*

Of course, since it is still an open question whether $\mathsf{PSPACE} = \mathsf{P}$, this does not unconditionally show that access to R provides a computational speed-up. But in the context of *nondeterministic* reductions to R, there is a striking speed-up:

Theorem 4.2. *Let R denote either R_C or R_K. Then*

- $\mathsf{NEXP} \subseteq \mathsf{NP}^R$ *[7].*
- $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}^{\mathsf{NP}^R}$ *[26].*

The initial reaction of the reader might be to ask if there is any real content to these theorems. Perhaps *every* computable set is efficiently reducible to R? Indeed, if we consider *nonuniform* reductions to be "efficient", then this is the case:

Theorem 4.3. $\mathsf{HALT} \in \mathsf{P}^R/\mathsf{poly}$ *[8].*

However, if we consider only reductions computed by Turing machines, then the situation becomes more complicated. Kummer showed that $\mathsf{HALT} \leq_{\mathrm{dtt}} R_C$ [33], but the running time of his reduction depends on the choice of the universal Turing machine U defining C complexity. For some choices of U the running time can be as little as doubly-exponential time [7], but for other choices of U the running time can be forced to be as slow as any given computable function [7]. On the other hand, it is still open whether or not $\mathsf{HALT} \in \mathsf{P}^{R_{C_U}}$ for some (or even every) choice of U. If that is the case, then indeed Theorems 4.1 and 4.2 are of little interest when $R = R_C$.

Open Question 4.4. *Is there any universal machine U such that $\mathsf{HALT} \in \mathsf{P}^{R_{C_U}}$?*

Much more is known about the case where $R = R_K$. The question of whether or not $\mathsf{HALT} \leq_{\mathrm{tt}} R_{K_U}$ depends on the choice of U [39]; see also [11] for an alternate proof. (Day has explored the analogous question for other Kolmogorov complexity measures [22].)

The inclusion from Theorem 4.1 that states $\mathsf{PSPACE} \subseteq \mathsf{P}^{R_K}$ is actually shorthand for $\mathsf{PSPACE} \subseteq \bigcap_U \mathsf{P}^{R_{K_U}}$, since the inclusion holds for every choice of universal machine U. It turns out that by explicitly considering the intersection over all U, one can obtain useful *upper* bounds:

Theorem 4.5. *[12, 21]* $\bigcap_U \mathsf{P}_{tt}^{R_{K_U}} \subseteq \mathsf{PSPACE}$ *and* $\bigcap_U \mathsf{NP}^{R_{K_U}} \subseteq \mathsf{EXPSPACE}$. *The techniques of [12] also allow one to show* $\bigcap_U \mathsf{P}^{\mathsf{NP}^{R_{K_U}}} \subseteq \mathsf{EXPSPACE}$.

This theorem relies crucially on the result of [21] that there are no undecidable sets in $\bigcap_U \mathsf{P}^{R_{K_U}}$

Resorting back to using P^{R_K} as a shorthand for $\bigcap_U \mathsf{P}^{R_{K_U}}$, we can thus summarize our knowledge about these classes as:

$$\mathsf{BPP} \subseteq \mathsf{P}_{tt}^{R_K} \subseteq \mathsf{PSPACE} \subseteq \mathsf{P}^{R_K}$$

$$\mathsf{PSPACE} \subseteq \mathsf{NEXP} \subseteq \mathsf{NP}^{R_K}$$

$$\mathsf{NEXP} \subseteq \mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}^{\mathsf{NP}^{R_K}} \subseteq \mathsf{EXPSPACE}$$

That is, although the oracle R_{K_U} is not even decidable (for any U), the class $\mathsf{P}_{tt}^{R_K}$ yields a complexity class between BPP and PSPACE. Similarly, the complexity of PSPACE is bounded above and below by adaptive and non-adaptive access to the oracle R_K.

4.1 Can the PSPACE Bound Be Improved?

In this section, let us focus on the inclusions $\mathsf{BPP} \subseteq \mathsf{P}_{tt}^{R_K} \subseteq \mathsf{PSPACE}$.

The paper [10] investigates whether the PSPACE upper bound on $\mathsf{P}_{tt}^{R_K}$ can be improved to $\mathsf{PSPACE} \cap \mathsf{P/poly}$. [10] found a connection to proof theory, and presented a collection of theorems (provable in ZF) with the property that, if the theorems were provable in certain extensions of Peano Arithmetic, then the $\mathsf{PSPACE} \cap \mathsf{P/poly}$ upper bound would hold. Any hopes that this might be how to show a $\mathsf{P/poly}$ bound were dashed by the next paper [6], which showed that the statements in question really *are* independent of the given extensions of PA.

However, on the positive side, [6] showed that the $\mathsf{PSPACE} \cap \mathsf{P/poly}$ upper bound *does* hold, in a related setting. [6] defined a class analogous to $\mathsf{P}_{tt}^{R_K}$ in terms of time-bounded K-complexity (with *very* large time bounds, so that they can be considered to be reasonable approximations to R_K). More precisely, consider the class of problems that are in $\mathsf{P}_{tt}^{R_{K^t}}$ for *all* large-enough time bounds (such as Ackermann's function, and beyond). [6] shows that this class lies between BPP and $\mathsf{PSPACE} \cap \mathsf{P/poly}$. Hirahara and Kawamura present a different restriction on reductions to R_K and R_C, yielding a class that lies between BPP and $\mathsf{NP}^{\mathsf{NP}}$ [27].

This is one of the main considerations that leads us to conjecture that $\mathsf{P}_{tt}^{R_K}$ actually coincides with BPP [3]. Although it is still open whether $\mathsf{P}_{tt}^{R_K}$ is contained in $\mathsf{P/poly}$, the results of [6] show that, if containment in $\mathsf{P/poly}$ does not hold, then it relies on the ability of nonadaptive poly-time reductions to distinguish between R_K and (for example) Ackermann-time-bounded K-random strings.

Let us assume for the moment that $\mathsf{P}_{tt}^{R_K} \subseteq$ P/poly. This means, in particular, that it is unlikely that $\mathsf{P}_{tt}^{R_K}$ contains NP. Yet $\mathsf{EXP}_{tt}^{R_K}$ contains NEXP, and thus there must be some critical time bound T when $\mathsf{DTIME}(T(n^{O(1)}))_{tt}^{R_K}$ first contains NP.

Open Question 4.6. *Does this occur for subexponential T? More generally, if* $\mathsf{BPP} = \mathsf{P}_{tt}^{R_K}$, *and the popular conjecture that* $\mathsf{BPP} = \mathsf{P}$ *also holds, then R_K provides no useful power for nonadaptive poly-time reductions. On the other hand, if* $\mathsf{EXP} \neq \mathsf{NEXP}$, *then nonadaptive* EXP*-reductions to R_K do provide significant power. At what point does this additional computational advantage kick in?*

4.2 Can One Characterize NEXP?

The same paper [3] that contains the conjecture $\mathsf{BPP} = \mathsf{P}_{tt}^{R_K}$ also contains a conjecture that $\mathsf{NEXP} = \mathsf{NP}^{R_K}$. (Weak) support for this conjecture comes largely from the fact that the inclusion $\mathsf{NP}^{R_K} \subseteq \mathsf{EXPSPACE}$ is proved by first observing that every NP-Turing reduction can be simulated by a *nonadaptive* EXP-reduction that asks only queries of polynomial length, and then observing that the inclusion $\mathsf{P}_{tt}^{R_K} \subseteq \mathsf{PSPACE}$ scales up to show that $\mathsf{EXP}_{tt}^{R_K} \subseteq \mathsf{EXPSPACE}$. It seems that one is throwing a *lot* away by replacing an NP-reduction by an EXP reduction, which would seem to indicate that the $\mathsf{EXPSPACE}$ upper bound is not tight.

Hirahara's recent result that $\mathsf{EXP}^{\mathsf{NP}} \subseteq (\Delta_2^p)^{R_K}$ [26] might, at first blush, seem to argue against the conjecture that $\mathsf{NEXP} = \mathsf{NP}^{R_K}$. That is, giving NP^{R_K} to a poly-time oracle Turing machine yields not merely $\mathsf{P}^{\mathsf{NEXP}}$, but also all of $\mathsf{EXP}^{\mathsf{NP}}$, which seems to be significantly larger than $\mathsf{P}^{\mathsf{NEXP}}$. (For instance, $\mathsf{P}^{\mathsf{NEXP}}$ is contained in $\mathsf{NEXP/poly}$, whereas $\mathsf{EXP}^{\mathsf{NP}}$ is in $\mathsf{NEXP/poly}$ iff it collapses to $\mathsf{P}^{\mathsf{NEXP}}$.) However, this heuristic argument implicitly assumes that $\bigcap_U (\Delta_2^p)^{R_{K_U}}$ is equal to $\mathsf{P}^{\bigcap_U \mathsf{NP}^{R_{K_U}}}$ – and it is not at all clear that this equality should hold. Thus it is conceivable that $\mathsf{NEXP} = \mathsf{NP}^{R_K}$ and $(\Delta_2^p)^{R_K} = \mathsf{EXP}^{\mathsf{NP}}$.

Currently, the best upper bound for NP^{R_K} is $\mathsf{EXPSPACE}$. However, there has been movement on a related front, as explained in the next paragraph.

The study of distinguishing random from pseudorandom distributions has led to very powerful insights and techniques. In this context, Vadhan and Gutfreund explored the class of problems that can be reduced to distinguishing random from pseudorandom in a very general sense [25]. They showed that all languages that can be reduced in a *restricted* sense to distinguishing random from pseudorandom lie in PSPACE, but the best upper bound for the *general* class of languages is $\mathsf{EXPSPACE}$ [12].

Hirahara has also shown [26] a better upper bound for this class: $\mathrm{S}_2^{\mathsf{EXP}}$ – the exponential-time analog of S_2^p (which is a class that lies in $\mathsf{ZPP}^{\mathsf{NP}}$ [20]). In particular, this class can be recognized by exponential-time alternating machines that make at most one alternation (in contrast to the $\mathsf{EXPSPACE}$ upper bound, which is exponential time with *no* bound on the number of alternations).

Open Question 4.7. *Can the* $\mathsf{EXPSPACE}$ *upper bound on* NP^{R_K} *be improved, to something much closer to* NEXP*?*

4.3 Promise Problems, Again

An alternative approach is to seek characterizations of BPP and NEXP in terms of reductions to the *promise problem* with "yes instances" consisting of those x such that $K(x) \geq |x|/2$, and "no instances" consisting of x such that $K(x) < \sqrt{|x|}$. Let us call this the Gap-K-complexity problem.

Open Question 4.8. *We have not succeeded in characterizing* BPP *or* NEXP *in terms of efficient reductions to R_K. Might one have a greater chance of success by considering efficient reductions to the Gap-K-complexity problem?*

The complexity classes that reduce to R_K also reduce to *any* solution to the Gap-K-complexity problem. Furthermore, all of the upper bounds that are proved in [12] carry over to this setting, and one obtains several other side-benefits. Note that it is no longer necessary to take the intersection over all universal machines U, since they all satisfy the promise. In a similar say, it is no longer necessary to distinguish between C and K complexity, since the gap between the YES and NO instances dwarfs the difference between the different measures. Also, there is a useful quantifier swap that applies: If a language B reduces to a promise problem (Y, N), it means that for every solution A to the promise problem (Y, N), there is a reduction from B to A. However, it is known [24, 44] that if this happens then there is also a *single* reduction that reduces B to *every* solution of (Y, N).

There is a long history of promise problems being used to understand complexity classes (such as SZK among many other examples), and this might be a better way of elucidating the connection between Kolmogorov complexity and complexity classes.

5 Conclusions

This rambling account is intended to gather together some recent (and not-so-recent) developments, regarding the complexity and computational power of determining the "complexity" of a string, using various notions of complexity. The reader should be cautioned that some of the open questions that are listed occurred to the author while he was writing the paper, and some of them might be quite easy to answer. Happy hunting!

Acknowledgments

The author acknowledges the support of NSF grant CCF-1555409, and thanks Diptarka Chakraborty (for helpful comments on an earlier draft of this work), Shuichi Hirahara (for allowing mention of his recent unpublished results), and Toni Pitassi (for helpful discussions).

References

[1] M. Agrawal, E. Allender, R. Impagliazzo, R. Pitassi, and S. Rudich. Reducing the complexity of reductions. *Computational Complexity*, 10:117–138, 2001.

[2] E. Allender. NL-printable sets and nondeterministic Kolmogorov complexity. *Theor. Comput. Sci.*, 355(2):127–138, 2006.

[3] E. Allender. Curiouser and curiouser: The link between incompressibility and complexity. In *Proc. Computability in Europe (CiE)*, volume 7318 of *Lecture Notes in Computer Science*, pages 11–16. Springer, 2012.

[4] E. Allender. Investigations concerning the structure of complete sets. In *Perspectives in Computational Complexity: The Somenath Biswas Anniversary Volume*, volume 26 of *Progress in Computer Science and Applied Logic*, pages 23–35, 2014.

[5] E. Allender, H. Buhrman, L. Friedman, and B. Loff. Reductions to the set of random strings: The resource-bounded case. In *Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 7464 of *Lecture Notes in Computer Science*, pages 88–99. Springer, 2012.

[6] E. Allender, H. Buhrman, L. Friedman, and B. Loff. Reductions to the set of random strings: The resource-bounded case. *Logical Methods in Computer Science*, 10(3):1–18, 2014. CiE 2012 Special Issue.

[7] E. Allender, H. Buhrman, and M. Koucký. What can be efficiently reduced to the Kolmogorov-random strings? *Annals of Pure and Applied Logic*, 138:2–19, 2006.

[8] E. Allender, H. Buhrman, M. Koucký, D. van Melkebeek, and D. Ronneburger. Power from random strings. *SIAM Journal on Computing*, 35:1467–1493, 2006.

[9] E. Allender and B. Das. Zero knowledge and circuit minimization. In *Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 8635 of *Lecture Notes in Computer Science*, pages 25–32. Springer, 2014. Winner of the Best Paper Award.

[10] E. Allender, G. Davie, L. Friedman, S. B. Hopkins, and I. Tzameret. Kolmogorov complexity, circuits, and the strength of formal theories of arithmetic. *Chicago Journal of Theoretical Computer Science*, 2013(5), April 2013.

[11] E. Allender, L. Friedman, and W. Gasarch. Exposition of the Muchnik-Positselsky construction of a prefix free entropy function that is not complete under truth-table reductions. Technical Report TR10-138, Electronic Colloquium on Computational Complexity (ECCC), 2010.

[12] E. Allender, L. Friedman, and W. Gasarch. Limits on the computational power of random strings. *Information and Computation*, 222:80–92, 2013. ICALP 2011 Special Issue.

[13] E. Allender, Joshua Grochow, and Cristopher Moore. Graph isomorphism and circuit size. Technical Report TR15-162, Electronic Colloquium on Computational Complexity (ECCC), 2015.

[14] E. Allender, D. Holden, and V. Kabanets. The minimum oracle circuit size problem. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, LIPIcs, pages 21–33. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[15] E. Allender, M. Koucký, D. Ronneburger, and S. Roy. The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory. *Journal of Computer and System Sciences*, 77:14–40, 2010.

[16] Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM Journal on Computing*, 38(1):63–84, 2008.

[17] S. Arora and B. Barak. *Computational complexity: a modern approach*, volume 1. Cambridge University Press, 2009.

[18] H. Buhrman, L. Fortnow, M. Koucký, and B. Loff. Derandomizing from random strings. In *25th IEEE Conference on Computational Complexity (CCC)*, pages 58–63. IEEE, 2010.

[19] H. Buhrman and E. Mayordomo. An excursion to the Kolmogorov random strings. *Journal of Computer and System Sciences*, 54:393–399, 1997.

[20] Jin-yi Cai. $S_2^p \subseteq \mathrm{ZPP}^{\mathrm{NP}}$. *Journal of Computer and System Sciences*, 73(1):25–35, 2007.

[21] M. Cai, R. Downey, R. Epstein, S. Lempp, and J. Miller. Random strings and tt-degrees of Turing complete c.e. sets. *Logical Methods in Computer Science*, 10(3):1–24, 2014.

[22] A. Day. On the computational power of random strings. *Annals of Pure and Applied Logic*, 160:214–228, 2009.

[23] R. Downey and D. Hirschfeldt. *Algorithmic Randomness and Complexity*. Springer, 2010.

[24] Joachim Grollmann and Alan L. Selman. Complexity measures for public-key cryptosystems. *SIAM J. Comput.*, 17(2):309–335, 1988.

[25] D. Gutfreund and S. P. Vadhan. Limitations of hardness vs. randomness under uniform reductions. In *APPROX-RANDOM*, volume 5171 of *Lecture Notes in Computer Science*, pages 469–482. Springer, 2008.

[26] Shuichi Hirahara. Personal Communication, 2015.

[27] Shuichi Hirahara and Akitoshi Kawamura. On characterizations of randomized computation using plain Kolmogorov complexity. In *Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 8635 of *Lecture Notes in Computer Science*, pages 348–359. Springer, 2014.

[28] Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *31st Conference on Computational Complexity, CCC*, LIPIcs, pages 18:1–18:20. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[29] John M. Hitchcock. Limitations of efficient reducibility to the kolmogorov random strings. *Computability*, 1(1):39–43, 2012.

[30] John M. Hitchcock and Aduri Pavan. On the NP-completeness of the minimum circuit size problem. In *Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS)*, volume 45 of *LIPIcs*, pages 236–245. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[31] Kayleigh Hyde and Bjørn Kjos-Hanssen. Nondeterministic automatic complexity of overlap-free and almost square-free words. *Electr. J. Comb.*, 22(3):P3.22, 2015.

[32] Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In *ACM Symposium on Theory of Computing (STOC)*, pages 73–79. ACM, 2000.

[33] M. Kummer. On the complexity of random strings. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 1046 of *Lecture Notes in Computer Science*, pages 25–36. Springer, 1996.

[34] Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

[35] Eyal Kushilevitz and Enav Weinreb. On the complexity of communication complexity. In *ACM Symposium on Theory of Computing (STOC)*, pages 465–474, 2009.

[36] L. A. Levin. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61:15–37, 1984.

[37] Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM*, 41(5):960–981, 1994.

[38] W.J. Masek. Some NP-complete set covering problems. Unpublished manuscript, 1979.

[39] A. A. Muchnik and S. Positselsky. Kolmogorov entropy in the context of computability theory. *Theoretical Computer Science*, 271:15–35, 2002.

[40] C. Murray and R. Williams. On the (non) NP-hardness of computing circuit complexity. In *30th Conference on Computational Complexity (CCC)*, LIPIcs, pages 365–380. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[41] J. Orlin. Contentment in graph theory: Covering graphs with cliques. *Indagationes Mathematicae (Proceedings)*, 80(5):406–424, 1977.

[42] Netanel Raviv. Truth table minimization of computational models, 2013.

[43] A. Razborov and S. Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55:24–35, 1997.

[44] Kenneth W. Regan. A uniform reduction theorem - extending a result of J. Grollmann and A. Selman. In *International Conference on Automata, Languages, and Programming (ICALP)*, volume 226 of *Lecture Notes in Computer Science*, pages 324–333. Springer, 1986.

[45] Michael Rudow. Discrete logarithm and minimum circuit size. Technical Report TR16-108, Electronic Colloquium on Computational Complexity (ECCC), 2016.

[46] Uwe Schöning. A low and a high hierarchy within NP. *Journal of Computer and System Sciences*, 27(1):14–28, 1983.

[47] Jeffrey Shallit and Ming-wei Wang. Automatic complexity of strings. *Journal of Automata, Languages and Combinatorics*, 6(4):537–554, 2001.

[48] Philip M. Spira. On time-hardware complexity tradeoffs for Boolean functions. In *Proc. 4th Hawaii Symp. on System Sciences*, pages 525–527, 1971.

[49] B. A. Trakhtenbrot. A survey of Russian approaches to perebor (brute-force searches) algorithms. *IEEE Annals of the History of Computing*, 6(4):384–400, 1984.

[50] H. Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer-Verlag New York Inc., 1999.