DCFS 2008 (Draft)

Descriptional Complexity of Formal Systems
Deadline for submissions: April ~~7~~ 27, 2008(strict)
Final versions: June 25, 2008

Circuit Complexity, Kolmogorov Complexity, and Prospects for Lower Bounds

Eric Allender

Department of Computer Science, Rutgers University
Piscataway, NJ 08855, USA
allender@cs.rutgers.edu

Abstract. This invited *Descriptional Complexity of Formal Systems* lecture provides an opportunity to raise awareness of the tight connection that exists between Kolmogorov Complexity and Circuit Complexity, and to argue that this connection is useful and interesting. Kolmogorov complexity has been used to provide examples of complete sets for classes such as EXP and PSPACE that are fundamentally different than the usual complete sets (in the sense that they are provably not complete under the usual reducibilities). Furthermore, there are connections between Kolmogorov complexity and recent approaches to proving lower bounds on circuit size.

Keywords: Kolmogorov Complexity, Constant-Depth Circuits, Threshold Circuits

1 Establishing the Connection

Here's an outrageous statement that nonetheless contains a lot of truth:

Kolmogorov Complexity is roughly the same thing as Circuit Complexity.

Why is this outrageous? After all, there are certain obvious formal similarities between Kolmogorov complexity and circuit complexity. Kolmogorov complexity provides a measure $C(x)$ of the "complexity" of a string x:

$$C(x) = \min\{|d| : U(d) = x\}$$

where U is (some fixed choice of) a universal Turing machine. (One appealing aspect of this definition is the *invariance* property: it doesn't really matter which universal Turing machine one picks, since the resulting measures are within $O(1)$ of each other. See [11].) In circuit complexity, we study the circuit size required to compute various functions, leading to the definition

$$\text{Size}(f) = \min\{\text{Size}(D) : D \text{ computes } f\}$$

where $\text{Size}(D)$ is the number of wires in a circuit D, and f is a Boolean function on n variables, for some n. In both settings, we're interested in the size of a small

Supported in part by NSF Grant DMS-0652582.

"description" (d or D). One minor difference between the two settings is that C assigns a measure of complexity to a *string* x, whereas Size assigns a measure of complexity to a *function* f – but this "difference" is easily dealt with. For any string x of length m, let the function f_x be the string on $m' = \lceil \log m \rceil$ variables, whose truth table is the string of length $2^{m'}$ whose first m bits are x and whose last $m' - m$ bits are zero. Thus one can extend the circuit size measure to all of Σ^*, where $\mathrm{Size}(x)$ is simply $\mathrm{Size}(f_x)$.

No, the reason why this is an outrageous statement is because $\mathrm{Size}(x)$ *is* computable, and $C(x)$ is *not*. The complexity of computing $\mathrm{Size}(x)$ has long been of interest in computer science. The Minimum Circuit Size Problem (MCSP) ($=\{(f,i) : f$ has a circuit of size $i\}$ is obviously in NP, but is not known (and perhaps not believed) to be NP-complete [8]. It is known that factoring and discrete log lie in $\mathrm{BPP}^{\mathrm{MCSP}}$ [3].

Given this mismatch (non-computability on one side, and perhaps less complexity than NP-completeness on the other), how can one possibly claim that circuit complexity and Kolmogorov complexity are the same in any sense? The answer lies in the following theorem:

Theorem 1. *[3]* $C(x) \approx \mathrm{Size}^H(x)$*, where* H *is the halting problem.*

(A note on notation: For an oracle A, $\mathrm{Size}^A(x)$ measures the size of the smallest *oracle circuit* computing f_x. I am using the notation $f \approx g$ to mean that f and g are polynomially related.)

This does establish a connection – but is it an interesting connection?

2 How "useful" is Kolmogorov complexity?

As a first attempt to claim that this is a useful connection, let us first observe that it is frequently felt to be a "defect" of Kolmogorov complexity that it is not computable. For example, there are many examples of applications of Kolmogorov complexity theory to real-world computing problems (such as [7], to pick one example) where Kolmogorov complexity is used to provide a theoretical justification for why a given approach should work, and then a gross approximation to Kolmogorov complexity is used, since $C(x)$ itself is not computable. But is there any indication that having access to the Kolmogorov complexity function would actually allow us to do *anything* interesting in a reasonable amount of time that we cannot already do?

Let us provide some definitions, and then re-state this question more formally. Let R_{C} be the set $\{x : C(x) \geq |x|\}$. Of course, R_{C} is not computable, and thus $\mathrm{P}^{R_{\mathrm{C}}}$ contains some noncomputable sets. But what *decidable* sets are in $\mathrm{P}^{R_{\mathrm{C}}}$? Is R_{C} NP-hard? I'll allow you to ponder this question for a few paragraphs before I tell you the answer.

Since R_{C} is undecidable, one naturally expects that there is a reduction from H to R_{C}. It is an easy exercise to convince yourself that there is no many-one reduction from H to either R_{C} or to its complement. For many years, there was no *time-bounded* Turing reduction known at all, but then in the mid-90's Kummer showed that there is a disjunctive truth-table reduction from H to R_{C} that is computable in some computable

time bound t [9]. How "efficient" is this reduction? Provably, no such disjunctive truth-table reduction can be performed in less than exponential time [2], and in fact, the run-time of such a reduction provably depends on which universal Turing machine one picks to define the Kolmogorov function C [2]. For some choices, the run-time of the reduction must exceed Ackerman's function. Clearly, such reductions are not of much use in determining what decidable sets are in $\mathrm{P}^{R_\mathrm{C}}$.

These remarks apply only to *disjunctive* truth-table reductions. It remains an open question whether $H \in \mathrm{P}^{R_\mathrm{C}}$ (and it is also open whether the answer depends on the choice of universal Turing machine).

I've kept you in suspense long enough, about whether R_C is NP-hard. Here's the answer:

Theorem 2. *[3, 2] The following hold for all choices of universal Turing machine:*

- $\mathrm{PSPACE} \subseteq \mathrm{P}^{R_\mathrm{C}}$.
- $\mathrm{NEXP} \subseteq \mathrm{NP}^{R_\mathrm{C}}$.

I'd be *very* interested to know if $\mathrm{P}^{R_\mathrm{C}}$ contains any larger complexity classes. This question is discussed further in [2].

I'm not aware of any simple direct proof of these inclusions. The proofs given in [3, 2] make crucial use of the connection between Kolmogorov complexity and circuit complexity, and are fairly intuitive if one is allowed to make free use of some heavy machinery from the theory of derandomization and interactive proofs. Similar techniques allow one to show that there is a P/poly reduction from H to R_C [3]; it would be very interesting to know if nonuniformity is essential, in order to present a "feasible" reduction.

This is exhibit number 1 in my argument that the connection between Kolmogorov complexity and circuit complexity is interesting. My argument continues in the next section.

3 Time-Bounded Kolmogorov Complexity

The "standard" way to define time-bounded Kolmogorov complexity is to use some variant of the following definition: Let t be a computable time bound. Define $C^t(x)$ to be $\min\{|d| : U(d) = x \text{ in time } t\}$.

One problem with this definition is that there is no invariance theorem; changing the choice of universal Turing machine U can result in *drastic* changes to the value of $C^t(x)$. The usual way that people try to side-step this problem is to observe that if U and U' are two different "efficient" universal Turing machines, then there is a time bound t' not much larger than t such that $C^t_U(x)$ is bounded from below by $C^{t'}_{U'}(x) - O(1)$ (but this ignores the fact that $C^{t'}_{U'}(x)$ may be *much* smaller).

Another problem with this definition (from my point of view) is that I am aware of no nice way to connect this definition to circuit complexity.

Fortunately, there is another approach to time-bounded Kolmogorov complexity. Levin introduced the following function [10]:

$$\mathrm{Kt}(x) = \min\{|d| + \log t : U(d) = x \text{ in time } t\}.$$

One advantage of Levin's definition is that there is an invariance theorem: changing the choice of universal Turing machine changes $\mathrm{Kt}(x)$ by at most $O(\log |x|)$. The motivation for this particular way of combining run-time and description length is that a provably optimal strategy for looking for accepting computations of a nondeterministic Turing machine involves searching in order of increasing Kt complexity.

There is also a pleasant connection between Kt complexity and circuit complexity:

Theorem 3. *[3] Let A be complete for $\mathrm{E} = \mathit{Dtime}(2^{O(n)})$. Then $\mathrm{Kt}(x) \approx \mathrm{Size}^A(x)$.*

The complexity of the set of strings having high Kt complexity raises some irritating questions. Let R_{Kt} denote the set of strings x such that $\mathrm{Kt}(x) \geq |x|$. On the one hand, it seems to be clear that R_{Kt} is intractible:

Theorem 4. *[3] $\mathrm{EXP} = \mathrm{NP}^{R_{\mathrm{Kt}}}$, and R_{Kt} is complete for EXP under P/poly reductions. (R_{Kt} is not complete for EXP under polynomial-time many-one or polynomial-time truth-table reductions, and thus is a fairly "natural" example of a set that is complete under more powerful reductions but not under the most commonly-studied reductions.))*

On the other hand, the question of whether R_{Kt} is in P is still open! It is also unknown if R_{Kt} is complete under polynomial-time Turing reductions.

Thus far, I have presented two connections between variants of Kolmogorov complexity and *oracle* circuit size. It is time to get rid of the oracles.

4 Oracle-free Circuit Size

The additive log term in the definition of $\mathrm{Kt}(x)$ (minimizing $|d| + \log t$ looks odd at first glance. What would happen if one were to replace $\log t$ with some other function of t? Note that the running time of any machine that outputs x will always be at least $|x|$, and thus defining the complexity of x in terms of minimizing $|d| + t$ would result in every string x having "complexity" at least $|x|$, leading to a definition of dubious utility.

This problem can be avoided by re-defining what it means for a description to "describe" a string. A string d is a perfectly good "description" of x if, given d, one can compute each bit of x. This leads to the following re-definitions of C and of Kt:

$$C(x) = \min\{|d| : \forall i \leq |x| + 1 \ \forall b \in \{0, 1, *\}, U(d, i, b) = 1 \iff b = x_i\}.$$

$$\mathrm{Kt}(x) = \min\{|d| + \log t : \forall i \leq |x| + 1 \ \forall b \in \{0, 1, *\}, U(d, i, b) = 1 \iff b = x_i \text{ in time } t\}.$$

In these definitions, we consider $x_i = *$ for $i > |x|$. The reader can verify that these definitions do not differ from the earlier definitions in any essential way. However, the run-time of U can now be much less than $|x|$.

This allows one to investigate some variants of Kt. The following definition was introduced in [3]:

$$\mathrm{KT}(x) = \min\{|d| + t : \forall i \leq |x| + 1 \ \forall b \in \{0,1,*\}, U(d,i,b) = 1 \iff b = x_i \text{ in time } t\}.$$

The following theorem provides motivation for studying KT.

Theorem 5. *[3]* $\mathrm{KT}(x) \approx \mathrm{Size}(x)$.

In case you still need evidence that the connection between circuit complexity and Kolmogorov complexity is interesting, let me mention that this connection played a central role in the proof that factoring BPP-reduces to MCSP [3].

Of course, circuit size is not the only meaningful and interesting measure of the complexity of a function. Much has been written about the formula size complexity and branching program size complexity of functions. These measures can also be viewed through the lens of Kolmogorov complexity. Consider the following two definitions that were introduced in [4]:

$$\mathrm{KB}(x) = \min\{|d|+2^s : \forall i \leq |x|+1 \ \forall b \in \{0,1,*\}, U(d,i,b) = 1 \iff b = x_i \text{ in space } s\}.$$

$$\mathrm{KF}(x) = \min\{|d|+2^t : \forall i \leq |x|+1 \ \forall b \in \{0,1,*\}, U'(d,i,b) = 1 \iff b = x_i \text{ in time } t\}.$$

In the definition of KF, U' is an *alternating* universal Turing machine.

Theorem 6. *[4]* $\mathrm{KF}(x) \approx$ *Formula-Size*(f_x).
$\mathrm{KB}(x) \approx$ *Branching-Program-Size*(f_x).

It is also possible to draw relationships to the depth-k threshold circuit size of a function, by formulating similar definitions in terms of the "threshold" Turing machines of Parberry and Schnitger [13].

5 Prospects for Lower Bounds in Circuit Complexity

There has been a great deal of pessimism about the likelihood of anyone making significant new progress in circuit complexity. Much of this pessimism can be traced to the fact that there has been very little progress on separating circuit complexity classes in the two decades that have passed since the work of Razborov [15] and Smolensky [16]. An additional factor is that Razborov and Rudich identified some significant obstacles that must be overcome before circuit lower bounds can be proved, in their work on "Natural Proofs" [14].

Here is an informal overview of the Natural Proofs framework, framed in terms of Kolmogorov complexity. Razborov and Rudich analyzed existing lower bound arguments, and observed that a proof that a function f is "hard" can typically be cast in terms of presenting some property Q that f has that is shared by no "simple" function (i.e., by no function in a small circuit class). In order to present such an argument, the property Q will typically be "constructive" (i.e., given a truth table of a function g, it should be "easy" to figure out if g has property Q). Since the proof relies on

showing that some "hard" functions have property Q, it is typically going to be the case that Q is "large" in the sense that a large fraction of the truth tables of a given size have property Q (since most functions are "hard"). A "Natural Proof" is one that proceeds by presenting a large, constructive property Q. Razborov and Rudich show that if there is a large constructive property Q that one can use to prove lower bounds against a circuit class $\mathcal{C}$ (Razborov and Rudich call this being "useful against $\mathcal{C}$)", then $\mathcal{C}$ is too weak to compute pseudorandom function generators. Since there are pseudorandom generators computable in TC^0 that are secure if factoring Blum integers is hard [12], this means that no "natural proof" can prove lower bounds against TC^0 if popular cryptographic assumptions hold.

What does this have to do with Kolmogorov complexity?

Recall that truth tables with small circuit complexity have small KT complexity. (For small circuit complexity classes such as L/poly, NC^1, or TC^0, one might want to consider other Kolmogorov measures such as KB and KF or other variants.) Thus a large combinatorial property Q that is useful against $\mathcal{C}$ is simply a set that contains many strings, but no string of small complexity. R_{Kt} and R_{KT} are canonical examples of such sets. Pseudorandom generators, by definition, are efficient algorithms that take short inputs and produce long outputs. Thus the strings output by a pseudorandom generator have small time-bounded Kolmogorov complexity. If there is a large constructive combinatorial property that is useful against $\mathcal{C}$ (for instance, if R_{KT} is in P/poly), then there is an efficient test to distinguish pseudorandom strings from random strings, and thus the generator cannot be secure. This is a high-level view of the Razborov-Rudich argument. A more detailed discussion along these lines can be found in [5]. Related observations (not phrased in terms of Kolmogorov complexity) can be found in [8].

Recently it was observed that if some of the standard complete problems for NC^1 are in TC^0, then they have TC^0 circuits of size n^{1+c} for every $c > 0$ [6]. This observation follows from the fact that these problems have a very strong self-reducibility property: there are linear-size AC^0-Turing reductions that solve the problem on instances of size n, asking queries only to instances of size n^ϵ. The authors speculate [6, 1] that this could provide an avenue for formulating proofs that avoid the pitfalls inherent in any "natural" proof that were identified by Razborov and Rudich [14]. Namely, there seems to be no reason why a natural proof cannot prove a lower bound of $n^{1.5}$ for the size constant-depth threshold circuits computing a complete language for NC^1. But by [6], this is sufficient to obtain a superpolynomial lower bound and separate NC^1 from TC^0. This would not be a "natural" proof, since very few functions have the strong self-reducibility property (i.e., the truth table of such a function has small Kolmogorov complexity), and thus this would not give rise to a "large" combinatorial property Q.

6 Conclusions

There are other ways of drawing meaningful connections between Kolmogorov complexity and circuit complexity, and I cannot pretend that this lecture does more than survey some aspects of this connection in which I have been personally involved, and

which I am glad to have the opportunity to share with you. Many more connections remain to be explored. If time permits, the lecture will also discuss some speculative approaches involving Kolmogorov complexity, in order to address the question of whether NEXP has polynomial-size depth-three threshold circuits.

References

[1] E. Allender. Cracks in the defenses: Scouting out approaaches on circuit lower bounds. In Computer Science – Theory and Applications (CSR 2008), volume 5010 of Lecture Notes in Computer Science, pages 3–10, 2008.

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

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

[4] E. Allender, M. Koucký, D. Ronneburger, and S. Roy. Derandomization and distinguishing complexity. In Proceedings of the 18th IEEE Conference on Computational Complexity, pages 209–220, 2003.

[5] Eric Allender. Circuit complexity before the dawn of the new millennium. In Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS), volume 1180 of Lecture Notes in Computer Science, pages 1–18, 1996.

[6] Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. In IEEE Conference on Computational Complexity, 2008. To appear.

[7] R. Cilibrasi, P.M.B. Vitanyi, and R. de Wolf. Algorithmic clustering of music based on string compression. Computer Music Journal, 28(4):49–67, 2004.

[8] V. Kabanets and J.-Y. Cai. Circuit minimization problem. In ACM Symposium on Theory of Computing (STOC), pages 73–79, 2000.

[9] 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, 1996.

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

[11] M. Li and P. Vitanyi. Introduction to Kolmogorov Complexity and its Applications. Springer, 1993.

[12] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. J. ACM, 51(2):231–262, 2004.

[13] I. Parberry and G. Schnitger. Parallel computation with threshold functions. Journal of Computer and System Sciences, 36:278–302, 1988.

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

[15] A. A. Razborov. Lower bounds on the size of bounded depth networks over a complete basis with logical addition. Mathematicheskie Zametki, 41:598–607, 1987. English translation in Mathematical Notes of the Academy of Sciences of the USSR 41:333-338, 1987.

[16] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In ACM Symposium on Theory of Computing (STOC), pages 77–82, 1987.