Electronic Colloquium on Computational Complexity, Report No. 109 (2017)

Does Looking Inside a Circuit Help?

Russell Impagliazzo[*] Valentine Kabanets[†] Antonina Kolokolova[‡]
Pierre McKenzie[§] Shadab Romani[¶]

June 22, 2017

Abstract

The Black-Box Hypothesis, introduced by Barak et al. [BGI+12], states that any property of boolean functions decided efficiently (e.g., in BPP) with inputs represented by circuits can also be decided efficiently in the black-box setting, where an algorithm is given an oracle access to the input function and an upper bound on its circuit size. If this hypothesis is true, then $\mathsf{P} \neq \mathsf{NP}$. We focus on the consequences of the hypothesis being false, showing that (under general conditions on the structure of a counterexample) it implies a non-trivial algorithm for Circuit-SAT. More specifically, we show that if there is a property F of boolean functions such that F has high sensitivity on some input function f of subexponential circuit complexity (which is a sufficient condition for F being a counterexample to the Black-Box Hypothesis), then Circuit-SAT is solvable by a subexponential-size circuit family. Moreover, if such a counterexample F is symmetric, then Circuit-SAT $\in \mathsf{P/poly}$. These results provide some evidence towards the conjecture (made in this paper) that the Black-Box Hypothesis is false if and only if Circuit-SAT is easy.

1 Introduction

Given access to a boolean function $f\colon \{0,1\}^n \to \{0,1\}$, how fast can we decide if $f \not\equiv 0$? If we can only access f as an oracle (i.e., in the "black-box" fashion), then it is well-known that one needs time $\Omega(2^n)$ for any deterministic or randomized algorithm (and time $\Omega(2^{n/2})$ for any quantum algorithm). What if f is computable by some small boolean circuit C, and we are given this circuit C (i.e., we can access f in the "white-box" fashion)? Then the question of deciding if $f \not\equiv 0$ is exactly the famous Circuit-SAT problem, and no nontrivial complexity lower bounds are known.

One possible approach to proving that $\mathsf{P} \neq \mathsf{NP}$ is to argue that being given an actual small circuit C computing a given boolean function f does not help much, compared to being given just oracle access to f, and being told the size of C. This could be formalized as the *Black-Box Hypothesis (BBH)* (introduced by Barak et al. [BGI+12] as "Scaled-down Rice's Theorem" conjecture), which can be informally stated as follows:

[*]University of California, San Diego; russell@cs.ucsd.edu
[†]Simon Fraser University; kabanets@cs.sfu.ca
[‡]Memorial University of Newfoundland; kol@cs.mun.ca
[§]Université de Montréal; mckenzie@iro.umontreal.ca
[¶]Simon Fraser University; sromani@sfu.ca

ISSN 1433-8092

> If a property F of boolean functions can be decided efficiently on circuits computing input functions, then F can also be decided efficiently in the black-box setting (that is, given oracle access to the input function and its circuit size bound).

If this hypothesis is true, then, for $F = \{f \colon \{0,1\}^n \to \{0,1\} \mid f \not\equiv 0\}$, we conclude that Circuit-SAT cannot be solved efficiently, since there are exponential lower bounds for deciding F in the black-box setting.

So proving the BBH is hard, as it would imply that $\mathsf{P} \neq \mathsf{NP}$. The hypothesis may well be false. Barak et al. [BGI+12] already proved that a version of the BBH (for promise problems) is false, assuming that one-way functions exist. Can we just disprove it then?

In this paper, we give some evidence that disproving the BBH is also hard, as it would have nontrivial algorithmic applications for Circuit-SAT. Note that if Circuit-SAT is efficiently solvable, then, as observed above, the Black-Box Hypothesis must be false. We conjecture that the converse implication also holds. Thus we conjecture the following:

> The BBH is false iff Circuit-SAT has a (somewhat) efficient algorithm.

We make a step towards proving this conjecture by showing that if the BBH fails *in a particular way*, then Circuit-SAT can be decided by a nonuniform family of subexponential-size circuits, which would disprove the nonuniform analogue of the Exponential-Time Hypothesis (ETH) of [IPZ01].

1.1 Our results

Before stating our results formally, let us discuss what it means for the BBH to fail. Clearly, if the BBH fails, there is a property F that is easy in the white-box setting (say, is in BPP), but requires superpolynomial complexity in the black-box setting. Note that for n-variate boolean functions f of circuit complexity $2^{\Omega(n)}$, there can't be any superpolynomial gap between the white-box and black-box complexities of deciding a given property F. This is because a white-box algorithm has to look at the input circuit, which is of size at least $2^{\Omega(n)}$, and the black-box algorithm can read the entire truth-table of f, build a trivial circuit of size about 2^n, and then just simulate the white-box algorithm on it, running in overall time at most $\mathsf{poly}(2^n)$. Thus any "magic" speed-up that we get for a property F violating the BBH must necessarily manifest itself over "easy" inputs, boolean n-variate functions f of circuit complexity at most $2^{o(n)}$. In other words, any black-box algorithm for F must be "slow" even if we care only about inputs f of low circuit complexity.

Recall that the sensitivity of a function F is the maximum, over all its inputs $x \in \{0,1\}^N$, of the number of positions $i \in [N]$ such that $F(x) \neq F(x^i)$, where x^i is x with the ith bit flipped. It is well-known that every F with sensitivity s requires $\Omega(s)$ queries to decide by any (randomized) black-box algorithm [Nis91]. Thus, a sufficient condition for any black-box algorithm deciding F to be "slow" (taking time at least T) is that F has "high" sensitivity (at least $\Omega(T)$). In fact, the same argument from [Nis91] actually implies that if F has a sensitive input x^*, then F requires large query complexity even when restricted to the inputs $x^*, (x^*)^1, (x^*)^2, \ldots, (x^*)^N$. The latter can be used to show (see Lemma 3.2 below) that a *sufficient condition for any black-box algorithm deciding F to be "slow" on all inputs f of subexponential circuit complexity* is the following:

> there exists a function $f^* \colon \{0,1\}^n \to \{0,1\}$ of circuit complexity $2^{o(n)}$ such that F has "high" sensitivity at f^*.

An important feature of the OR function (which explains why it requires high black-box complexity) is the existence of a highly sensitive input, the all-zero string. Moreover, this sensitive input has a very low circuit complexity (as a boolean function). We show that if the BBH fails because of a property F with similar conditions (i.e., that F has an "easy" but "highly sensitive" input), then Circuit-SAT admits a nontrivial algorithm.

Theorem 1.1 (Main theorem: Informal version)**.** *Suppose there is a property F of n-variate boolean functions such that*

1. *F is decidable in BPP in the white-box setting, but,*
2. *for almost all n, F has an input $f^*\colon \{0,1\}^n \to \{0,1\}$ of sensitivity $2^{\Omega(n)}$ and of circuit complexity $2^{o(n)}$ (which implies that F requires exponential time $2^{\Omega(n)}$ to decide in the black-box setting, even on inputs f of circuit complexity $2^{o(n)}$).*

Then Circuit-SAT for n-input circuits of size at most $2^{o(n)}$ can be decided by a nonuniform family of circuits of size $2^{o(n)}$.

Intuitively, Theorem 1.1 says that if the BBH fails in a strong way for some property F, with an exponential gap between the white-box and the black-box complexities, so that the high black-box complexity of F can be explained through the existence of a highly sensitive input f^* (of relatively low circuit complexity), then Circuit-SAT is decidable by a subexponential-time nonuniform algorithm.

We also observe that the assumption of Theorem 1.1 holds for any property F violating the BBH whenever F is one of the following:

- F is a symmetric function, or
- F is a subset of easy functions (i.e., $F \subseteq \{f \mid size(f) \leq 2^{o(n)}\}$).

Hence, if a counterexample to the BBH is of this kind, then Circuit-SAT is easy for nonuniform algorithms.

Finally, for the special case of *monotone* properties F, we get a version of Theorem 1.1 where it suffices to assume that a sensitive input in item (2) of Theorem 1.1 has just superpolynomial sensitivity $s > n^{\omega(1)}$ and circuit complexity $s^{o(1)}$ (rather than requiring an exponential sensitivity $s \geq 2^{\Omega(n)}$). More precisely, we prove the following.

Theorem 1.2 (Monotone Properties)**.** *Let F be a monotone property such that*

1. *F is decidable in BPP in the white-box setting, but,*
2. *for almost all n, F has an input $f^*\colon \{0,1\}^n \to \{0,1\}$ of sensitivity $s \geq n^{\omega(1)}$ and of circuit complexity $s^{o(1)} \geq \mathsf{poly}(n)$ (which implies that F requires superpolynomial time to decide in the black-box complexity setting, even on inputs of circuit complexity $s^{o(1)}$).*

Then Circuit-SAT for n-input circuits of size at most $2^{o(n)}$ can be decided by a nonuniform family of circuits of size $2^{o(n)}$.

We also use a "win-win" argument to show the following: If a monotone property is a counterexample to the Block-box Hypothesis (with appropriate parameters), then either Circuit-SAT is nonuniformly easy infinitely often, or BPP $\subseteq$ NP (see Theorem 5.2).

1.2 Related work

The Black-Box Hypothesis has its roots in a classical result of computability theory, Rice's theorem, which says that any non-trivial property of languages accepted by Turing machines is undecidable. There are two ways of interpreting Rice's theorem: (1) Given a Turing machine M, the only thing one can do is to run it, or (2) the Halting problem is the easiest non-trivial property of languages of Turing machines, in the sense that if any non-trivial property is decidable, then so is the Halting problem.

The intuition that it may be hard to understand what an algorithm does by looking at the algorithm description naturally extends to the class of non-uniform algorithms (i.e., circuits). The focus of this paper is on the second interpretation of Rice's theorem, with $\mathsf{Circuit\text{-}SAT}$ as a complexity counterpart of the Halting problem. In other words, we would like to show any "non-trivial" counterexample to the Black-Box Hypothesis implies a somewhat efficient algorithm for SAT.

There have been several attempts to scale down Rice's theorem to the complexity-theoretic realm, with different notions of "non-trivial" and "hard". In Rice's theorem, "non-trivial" means neither F nor $\bar{F}$ is empty, and "hard" = undecidable. Borchert and Stephan [BS00] pioneered a line of research that looked at counting properties of circuits and stated an analogue of Rice's theorem for such properties: if a counting property is non-empty, then it is UP-hard. There, a property F is a counting property if it only depends on the number of solutions (i.e., F is a symmetric function). Subsequently, Hemaspaandra and Rothe [HR00] and Hemaspaandra and Thakur [HT04] improved the hardness result, obtaining a version of Rice's theorem with NP-hardness.

Barak et al. [BGI+12] also look at the properties of boolean functions computed by circuits, but consider a property trivial if it can be decided by checking the circuit value on relatively few points. That is, in their setting, the semantic property $f(00\dots0) = f(11\dots1)$ is trivial, but $\exists x\ f(x) = 1$ is not. Their "Scaled-down Rice's theorem" conjecture states that every property of boolean functions f that can be computed in BPP given a circuit for f can be also computed in comparable probabilistic polynomial time given only oracle access to f and an upper bound on its circuit complexity. There is a clear relation to obfuscation: if it were possible to produce a circuit for any f so garbled that access to it is not much better than the black-box access, that would prove the conjecture. However, in the same paper they show impossibility of achieving such obfuscation. Nonetheless, [BGI+12] is able to disprove a certain "promise" version of the conjecture, under the assumption that one-way functions exist (using a special family of unobfuscatable circuits). The main statement, which we will call here "the Black-Box Hypothesis", remains open.

1.3 Our techniques

Our starting point is the isolation lemma of Valiant and Vazirani [VV86], which can be interpreted to say that any white-box BPP algorithm deciding the property $F = XOR$ yields a BPP algorithm for $\mathsf{Circuit\text{-}SAT}$. This can be extended to any property F computing a symmetric function, at the expense of introducing a small (polynomial) amount of nonuniformity. The main idea is to take advantage of the existence of a very sensitive input f for any symmetric property F. (For example, for the case of XOR, every input $f\colon \{0,1\}^n \to \{0,1\}$ has maximum sensitivity 2^n. In general, every symmetric F has a polysize input f of sensitivity at least $2^n/2$.)

Suppose that $f\colon \{0,1\}^n \to \{0,1\}$ is such a sensitive input for the property F, and moreover, suppose that f is computable by a small circuit C_f (say of $\mathsf{poly}(n)$ size). To decide if a given circuit C on n inputs is satisfiable, we first use the Valiant-Vazirani result to get from C a new circuit C' such that C' is uniquely satisfiable if C is satisfiable, and C' is unsatisfiable otherwise. By XORing the circuits C_f and C', we get a new (small) circuit that leaves f unchanged if C is unsatisfiable, and flips f in exactly one location if C is satisfiable. If the flipped location happens to land among the sensitive locations of f, we can detect this by running our assumed white-box algorithm on $C_f \oplus C'$ and noting that its output is different from that on C_f. To make sure that the flipped location is among the sensitive ones for f, we consider a random-shift version of C' so that its unique satisfying assignment (if it exists) will be in a uniformly random location. As, by assumption, f has very many sensitive locations, this randomization will ensure that we detect if C is satisfiable with high probability. The runtime of the described algorithm is polynomial in the sizes of C_f and C. We think of a small circuit C_f as nonuniform advice, thereby getting a nontrivial nonuniform algorithm for Circuit-SAT.

The (nonuniform) algorithm for Circuit-SAT described above achieves high success probability in case a sensitive input $f\colon \{0,1\}^n \to \{0,1\}$ (provided as advice via a small circuit computing f) has very large sensitivity $s \geq \Omega(2^n)$. What if the sensitivity is only as large as $2^{\Omega(n)}$? (Such a lower bound is the best one can hope for if one assumes the Sensitivity Conjecture and that the given property F has exponential decision tree complexity.) In this case, our described algorithm would have success probability only about $2^{-\delta n}$, for some constant $0 < \delta < 1$, for solving Circuit-SAT on n-input circuits. However, if the algorithm runs in (non-uniform) time at most $2^{o(n)}$ (which will happen if the advice circuit C_f is of size at most $2^{o(n)}$), then we can use the amplification technique of Paturi and Pudlák [PP10] to get a new algorithm in non-uniform time $2^{o(n)}$ that succeeds with probability 1.

For the special case of monotone properties F, we show how to make do with even smaller sensitivity assumption on the advice function f, getting a subexponential-size Circuit-SAT algorithm for any superpolynomial sensitivity $s > n^{\omega(1)}$. The idea is to use hashing (which is also the main ingredient in the aforementioned result of [PP10]).

If we don't assume that a sensitive input f for a given property F would have a small circuit, we can still say something interesting by applying a "win-win" argument. Informally, we get that if F has sensitive inputs and an efficient white-box algorithm, then either Circuit-SAT is nonuniformly easy (in subexponential size, infinitely often), or we get an efficient "hardness tester": a polytime algorithm that accepts only truth tables of boolean functions of exponential circuit complexity, and accepts at least one such truth table. Getting such a hardness tester is a highly nontrivial task, and is not known unconditionally. Once you have this tester, you can, for example, conclude that $\mathsf{BPP} \subseteq \mathsf{NP}$, using standard "hardness-randomness" trade-offs [NW94, BFNW93, IW97].

Remainder of the paper. We give some basic definitions and facts in Section 2. We state and discuss the Black-Box Hypothesis in Section 3. We prove Theorem 1.1 in Section 4. In Section 5, we consider the special case of monotone properties as counterexamples to the Black-Box Hypothesis, getting a proof of Theorem 1.2. In Section 6, we consider the case of properties defined using succinct versions of the Minimal Circuit Size Problem (MCSP). We consider some variants of the BBH for restricted circuit classes in Section 7. We conclude with some open problems in Section 8.

2 Preliminaries

The truth table of a boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is denoted by $tt(f)$. With a boolean circuit C on n inputs, we associate the boolean function $f_n = [C]$ computed by C. Slightly abusing the notation, we use $tt(C)$ to denote the truth table of a boolean function computed by the circuit C. A standard encoding of C as a binary string is denoted $desc(C)$.

A *property of boolean functions* is a function $F\colon \{0,1\}^{2^n} \to \{0,1\}$, where strings over $\{0,1\}^{2^n}$ are interpreted as truth tables of boolean functions on n variables, for every n. A *meta-language* over circuits corresponding to a property F is $L_F = \{desc(C) \mid C$ is a boolean circuit and $tt(C) \in F\}$. In particular, if L_F is a meta-language over circuits, then for any circuits C_1 and C_2, if $[C_1] = [C_2]$ then $C_1 \in L_F \Leftrightarrow C_2 \in L_F$.

The size of a boolean circuit C is the number of gates plus the number of wires. Let $size(f) = \min_{C,[C]=f} |C|$. We say that $f \in \mathsf{SIZE}(t(n))$ if $size(f) \le t(n)$.

We denote by $\mathsf{Circuit\text{-}SAT}_{n,m}$ the problem of deciding the satisfiability of a given n-input circuit of size at most m. For a time bound $t = t(n)$, we denote by $\mathsf{RTIME}(t)$ the class of languages decidable by randomized algorithms, with one-sided error at most $1/2$, in time t; as usual, $\mathsf{RP} = \mathsf{RTIME}(\mathsf{poly})$. For an advice size function $a = a(n)$, we denote by $\mathsf{RTIME}(t)/a$ the class of languages decidable by an $\mathsf{RTIME}(t)$ algorithm, given the correct advice of size at most a.[1]

For a function $F\colon \{0,1\}^N \to \{0,1\}$, with $N = 2^n$, we can think of inputs to F as truth tables of n-variate boolean functions $f\colon \{0,1\}^n \to \{0,1\}$. For a circuit size bound $t = t(n)$, we define the *randomized decision tree complexity of F on inputs of complexity at most t*, denoted $Rt_t(F)$, as the minimal depth of a randomized decision tree deciding F, with error probability at most $1/3$, on all inputs $f\colon \{0,1\}^n \to \{0,1\}$ of $size(f) \le t(n)$.

A boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is *sensitive* on the ith bit of input x if flipping that bit changes the value of $f(x)$. Sensitivity of f on input $x \in \{0,1\}^n$, denoted by $\mathsf{sens}(f,x)$, is the number of bits in x to which f is sensitive. The sensitivity of f, denoted $\mathsf{sens}(f)$, is $\max_{x\in\{0,1\}^n} \mathsf{sens}(f,x)$.

Simon's lemma [Sim83] gives a weak lower bound on $\mathsf{sens}(f)$. We will use the following corollary of this lemma from [AV15]:

Lemma 2.1 ([Sim83])**.** *For every non-constant n-variate boolean function f, there exists an input $x \in f^{-1}(1)$ with $\mathsf{sens}(f,x) \ge n - \log |f^{-1}(1)|$.*

Although decision tree complexity of a boolean function is polynomially related to many other measures that we do not define here (see, for example, [BdW02, HKP11]), its relationship with the sensitivity remains elusive. The question of whether there is a polynomial relation between $\mathsf{sens}(f)$ and the decision tree complexity $Dt(f)$, known as the Sensitivity Conjecture, has been formulated already in [Nis91]. However, despite much work, it is still unresolved.

Conjecture 2.1 (Sensitivity conjecture)**.** There exists an integer k such that, for any function f, $Rt(f) \le \mathsf{sens}(f)^k$.

[1]For semantic complexity classes such as RTIME, it is customary to use the weaker notion of a class with advice, where the algorithm is required to behave as a true RTIME-type algorithm only when given a correct advice string, and can behave arbitrarily otherwise.

3 Black-Box Hypothesis

3.1 Defining BBH

To investigate whether having a circuit C_f for an input function f helps decide a property F of boolean functions, we compare the complexity of deciding F on f given a circuit C_f versus given an oracle access to f. In the latter case, following [BGI+12], an algorithm deciding $F(f)$ is also given as its input the size m of some C_f (or, rather, an upper bound on C_f), in unary (that is, the algorithm can "see how large the box is", but cannot peek inside). This makes the comparison of the running time in both frameworks more meaningful. With this intuition, we define "white-box" and "black-box" algorithms as follows.

Definition 3.1 (White-box vs. black-box algorithms)**.** An algorithm A decides a property F in *white-box* if A decides the corresponding meta-language L_F. That is, given as input a string $desc(C)$ A accepts iff $[C] \in F$.

An algorithm A decides F in *black-box* if $A^f(1^n, 1^m)$ accepts iff $f \in F$, where $f\colon \{0,1\}^n \to \{0,1\}$, m is an upper bound on the circuit size of f and A^f denotes that the algorithm A has oracle access to the boolean function f; as usual, 1^n and 1^m represent n and m in unary.

Definition 3.2. A property F is in *white-box* BPP, denoted $F \in \mathsf{wbBPP}$, if there is a BPP algorithm deciding L_F. We say F is in *black-box* BPP, denoted $F \in \mathsf{bbBPP}$, if there is a black-box randomized algorithm $A^f(1^n, 1^m)$ deciding F in time polynomial in $n+m$, with the probability of error at most $1/3$ over the choice of randomness, for every f, n, m.

With the above definitions, the Black-Box Hypothesis can be stated concisely as follows.

Hypothesis 3.1 (Black-Box Hypothesis (BBH))**.** *For any property F of boolean functions,*

$$F \in \mathsf{wbBPP} \iff F \in \mathsf{bbBPP}.$$

If the BBH holds, then $\mathsf{P} \neq \mathsf{NP}$, as the well-known exponential black-box lower bounds for SAT would rule out even a subexponential-time probabilistic algorithm for SAT. On the other hand, if $\mathsf{NP} \subseteq \mathsf{BPP}$, then the BBH is false, with SAT as a counterexample. Suppose the BBH is false. Would that imply that SAT is easy? We make the following conjecture.

Conjecture 3.1. (Informal) BBH is false iff Circuit-SAT is easy.

As a step towards proving the conjecture, we show that if the BBH fails in a particular way (see the next subsection for the definition), then there is a family of circuits of subexponential size that decides Circuit-SAT.

3.2 Defining a Strong Counter-Example to BBH

As noted before, a property $F \in \mathsf{BPP}$ can only be a counterexample to BBH when any black-box algorithm requires superpolynomial time on some input of subexponential size (otherwise white-box complexity and black-box complexity are polynomially related).

Thus, if F is not in black-box BPP, then any black-box algorithm deciding F requires superpolynomial time on some input of subexponential circuit size, which we call an easy input.

Ideally, we would like to prove that if the BBH fails, then Circuit-SATis easy. We do not know how to show such an implication yet. Instead, we consider the following *sufficient* condition for the BBH to fail.

Definition 3.3 (Strong counterexample to the BBH)**.** A property F is an *s-strong counterexample to the BBH* if

1. F is in wbBPP, but
2. for almost all n, F has an input $f^* : \{0,1\}^n \to \{0,1\}$ of $size(f^*) \leq 2^{o(n)}$ such that $\mathsf{sens}(F, f^*) \geq s$.

We call a property a *strong counterexample* if it is $2^{\Omega(n)}$-strong.

Next we argue that a strong counterexample to the BBH as defined above would indeed violate the BBH. First, we recall the following result.

Lemma 3.2 (implicit in [Nis91])**.** *Let F be a property of n-variate boolean functions. If $\mathsf{sens}(F, f) \geq s$ for some boolean function $f \in \mathsf{SIZE}(t)$, then $Rt_{(t+cn)} \geq (2/3)s$ (for some constant $c > 0$).*

Proof. Let f^i be the function that disagrees with f on the ith bit of the output, which is a sensitive bit of f. Thus, (the truth tables of) f and f^i are Hamming neighbours and circuit complexity of f^i is greater than f by at most a linear factor, i.e., $size(f^i) \leq size(f) + O(n)$. Now to distinguish f from each Hamming neighbour f^i with probability at least $2/3$, any randomized decision tree needs to query the ith bit with probability at least $2/3$. As there are s many sensitive bits for f, the expected number of queries is $(2/3)s$. Thus, there is one branch on which the randomized decision tree has to query $(2/3)s$ of the bits. □

Applying Lemma 3.2 immediately yields the required implication.

Corollary 3.3. *If F is a $n^{\omega(1)}$-strong counterexample to the BBH, then $F \notin \mathsf{bbBPP}$ (and hence, the BBH is false).*

3.3 Examples of properties with easy sensitive inputs

We give a few examples of properties with easy sensitive inputs. For each of these properties, violating the BBH is actually *equivalent* to being a strong counterexample to the BBH.

Symmetric properties. A property F is symmetric if the membership of $tt(f) \in F$ depends only on the number of 1s in $tt(f)$. Such properties were the focus of one of the previous formulations of a possible complexity analogue of Rice's theorem, due to Borchert and Stephan [BS00] (though their notion of hardness was somewhat different). A basic symmetric property of N-bit strings such as OR or XOR has an easy input (the all-0 string) of sensitivity N. We note that every symmetric property has an easy input of sensitivity at least $N/2$.

Lemma 3.4. *If F is a non-trivial symmetric property of n-variate boolean functions, then there is a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ with $\mathsf{sens}(F, f) \geq 2^n/2$ such that f is computable by an AC^0 circuit of polynomial size.*

Proof. As F is a non-trivial property, there is a number $1 \leq k \leq 2^n$ such that a $tt(f)$ with $k-1$ ones is accepted by F (wlog), but any $tt(f)$ with k ones is rejected by F. If $k \geq 2^n/2$, then any string with k ones has sensitivity k. Otherwise, any string with $k-1$ ones has sensitivity $2^n - (k-1) \geq 2^n/2$.

Let k be the number of 1s in an input with sensitivity at least $2^n/2$. Define a required boolean function f with exactly k ones in its truth table by $f(x) = 1$ iff $x < k$, where x is interpreted as an integer in binary. It is easy to see that f has a polynomial-size circuit, even of AC^0 type (as the comparison of two n-bit integers can be implemented in AC^0 [CSV84]). □

Subsets of easy functions. Consider a property F that only contains a subset of easy functions, that is, only functions of circuit complexity at most $t = 2^{o(n)}$. Easy functions form a very sparse set (the number of n-bit functions of circuit size at most t is at most 2^{t^2}). So by Simon's lemma (Lemma 2.1), F contains an (easy) instance of sensitivity at least $2^n - t^2 = 2^n - 2^{o(n)} = \Omega(2^n)$.

4 Circuit-SAT algorithm from strong counterexamples

The main theorem of this section shows that a strong counterexample to the BBH (as in Definition 3.3) implies that Circuit-SAT on n-input circuits of subexponential size can be decided by subexponential-size circuits. Formally, we have the following.

Theorem 4.1. *If there is a strong counterexample to the BBH, then*

$$\mathsf{Circuit\text{-}SAT}_{n,2^{o(n)}} \in \mathsf{SIZE}(2^{o(n)}).$$

We prove this theorem in two steps. First we show (in Section 4.1) how sensitivity can be exploited for deriving a randomized algorithm for satisfiability, whose success probability depends on the assumed sensitivity of a given counterexample to the BBH. Then (in Section 4.2) we amplify the success probability of our algorithm.

4.1 From high sensitivity to Circuit-SAT

Here we prove the following.

Lemma 4.2. *Let F be an s-strong counterexample to the BBH, with an s-sensitive function family $f \in \mathsf{SIZE}(t)$. Then $\mathsf{Circuit\text{-}SAT}_{n,m}$ is decidable in randomized time $\mathsf{poly}(t, m)$, with success probability $\Omega(s/2^n)$, given the advice of size $\mathsf{poly}(t)$. In particular, we have that*

$$\mathsf{Circuit\text{-}SAT}_{n,m} \in \mathsf{SIZE}(\mathsf{poly}(n \cdot (t(n) + m) \cdot 2^n/s(n))).$$

Proof. Let A_F be a BPP algorithm for L_F. By Adleman's argument [Adl78], we can assume that A_F is a deterministic algorithm, using at most $\mathsf{poly}(m)$ bits of advice on inputs of length m.

As a warm-up, suppose that F has maximal sensitivity 2^n, and, moreover, for each n there is a maximally sensitive input $tt(f)$ where f has a circuit C_f of size t. Now, if C has at most 1 satisfying assignment, it is enough to check whether $A_F(C \oplus C_f) = A_F(C_f)$: if there is a satisfying assignment for C, it flips a sensitive bit of $tt(C_f)$, otherwise $tt(C \oplus C_f) = tt(C_f)$.

To use the idea described above we need to guarantee that the circuit C for which we want to decide satisfiability has at most one satisfiable assignment. This can be done by applying the Valiant-Vazirani reduction [VV86] to get new circuit C'. Assuming that f is a highly sensitive input, we have a non-trivial chance of hitting one of its sensitive bits if we randomly shift a unique satisfying assignment of C'. That is, we check $A_F(C'(x \oplus r) \oplus C_f)$, where r is a random binary string of length $|x|$. More formally, our algorithm for Circuit-SAT is as follows.

Algorithm for Circuit-SAT
Input: A circuit C on n inputs.
Advice: A circuit C_f of size at most t such that $tt(C_f)$ is an s-sensitive string for F.

1. Apply the Valiant-Vazirani reduction to C to obtain a list $C_1, \dots, C_n$ satisfying the following: if C is unsatisfiable then so is every C_i on the list, and if C is satisfiable, then, with probability at least $1/2$, at least one C_i on the list has a unique satisfying assignment.
2. Pick a random $r \in \{0,1\}^n$. For each C_i on the list, check if
$$F([C_f]) \neq A_F(C_i(x \oplus r) \oplus C_f).$$
If the check passes for at least one $1 \leq i \leq n$, then accept; otherwise, reject.

The running time of the described algorithm is $\mathsf{poly}(n, t+m)$. The advice size is $\mathsf{poly}(t)$, as we need C_f, plus the advice of size $\mathsf{poly}(|C| + |C_f|)$ used in Adleman's averaging argument. If C is unsatisfiable, then the algorithm rejects C with probability 1. If C is satisfiable, then the algorithm accepts with probability at least $(1/2) \cdot s/2^n$ (the success probability of the Valiant-Vazirani reduction in Step (1), times the probability of hitting a sensitive bit of the advice $tt(C_f)$ by a random shift r in Step (2)).

Finally, applying Adleman's argument to the randomized algorithm above, we get a nonuniform circuit family solving Circuit-SAT with the stated parameters. □

Corollary 4.3. *Let F be a nontrivial symmetric property such that $L_F \in$ BPP. Then* Circuit-SAT $\in$ RP/poly $\subseteq$ P/poly.

Proof. The proof follows from Lemma 3.4 and Lemma 4.2. □

4.2 Amplifying the success probability

Lemma 4.2 is a weaker version of Theorem 4.1 which needs the sensitivity bound $s \geq 2^{n-o(n)}$. To handle a smaller sensitivity $2^{\delta n}$, for any $\delta > 0$, we need a better way of amplifying the success probability of our randomized Circuit-SAT algorithm above, without increasing the circuit size by too much. We will use the following Exponential Amplification lemma by Paturi and Pudlák [PP10].

Lemma 4.4 (Exponential amplification lemma[PP10])**.** *Let $\mathcal{G}$ be a family of probabilistic circuits of size bounded by $g(m,n)$ such that $\mathcal{G}$ decides* Circuit-SAT *with one-sided error, achieving the success probability $2^{-\delta n}$ on satisfiable instances. Then there exist a circuit family $\mathcal{G}'$ deciding* Circuit-SAT *with success probability $2^{-\delta^2 n}$ on satisfiable instances, for all large enough n, where the circuit size of $\mathcal{G}'$ is bounded by $g'(n,m) = O(g(\lceil \delta n \rceil) + 5, \tilde{O}(g(n,m)))$.*

Now we can prove Theorem 4.1.

Proof of Theorem 4.1. Let $G^0_{m,n}$ be the circuit family encoding the randomized algorithm from Lemma 4.2. For concreteness, let $desc(C_f) = 2^{n^\gamma}$ denote a bound on the size of $|C_f|$. The size of the complete circuit $G^0_{m,n}$ is $O(2^{kn^\gamma} \cdot n^{k\gamma+1} \cdot m^k)$, where k is the exponent of the running time of A_F. Assuming that $m \leq |C_f|$ to bound smaller factors, $|desc(G^0_{m,n})| = O(2^{kn^\gamma} \cdot n^{(k+1)\gamma+1} \cdot m^k)$.

Apply the Exponential amplification lemma for t iteration to $G^0_{m,n}$, where $t \in \omega(1)$ is a very slow growing function. If $2^{o(n)} = 2^{\alpha(n)}$ is the bound on the advice circuit $|C_f|$, then we need $k^t \cdot \alpha(n) < \beta(n)$, where $\beta(n) \in o(n)$. As t is non-constant, success probability becomes $2^{\delta^t n} \in 2^{o(n)}$. Now, using the standard techniques to amplify the success probability (with $2^{\delta^t n} + O(n)$ trials and fixing randomness by the averaging argument), we obtain a deterministic circuit of subexponential size solving Circuit-SAT for circuits of description size m on n variables. $\square$

5 Monotone properties

Here we consider a special case of monotone properties F. First, we argue that it suffices to have a monotone counterexample to the BBH with just superpolynomial sensitivity in order to obtain a non-trivial Circuit-SAT algorithm (Section 5.1). Then we show that having a monotone property F in white box P such that F requires high decision tree complexity implies either a nontrivial Circuit-SAT algorithm or nontrivial derandomization of BPP (Section 5.2).

5.1 Handling a lower sensitivity bound

So far, to get a nontrivial Circuit-SAT algorithm from a counterexample F to the BBH, we assumed that we have an easy sensitive input $f^*\colon \{0,1\}^n \to \{0,1\}$ with $\mathsf{sens}(F, f^*) \geq 2^{\Omega(n)}$. Here we show that for a special case of *monotone* properties F, any superpolynomial sensitivity $s \in n^{\omega(1)}$ would suffice to get the same kind of Circuit-SAT algorithms.

Theorem 5.1. *Let F be a monotone property such that*

1. *F is decidable in* BPP *in the white-box setting, but,*
2. *for almost all n, F has an input $f^*\colon \{0,1\}^n \to \{0,1\}$ of sensitivity $s \geq n^{\omega(1)}$ and of circuit complexity $s^{o(1)} \geq$* poly(n) *(which implies that F requires superpolynomial time to decide in the black-box complexity setting, even on functions of circuit complexity $s^{o(1)}$).*

Then $\mathsf{Circuit\text{-}SAT}_{n,2^{o(n)}} \in \mathsf{SIZE}(2^{o(n)})$.

Proof. Without loss of generality, assume that $F(f^*) = 1$. Given f^* as advice, we describe a Circuit-SAT algorithm for circuits on $k = \log_2 s$ inputs. We will use random hash functions. Recall that a universal hash family $\mathcal{H}_{n,k} = \{h\colon \{0,1\}^n \to \{0,1\}^k\}$ has the properties: (1) for every fixed $x \in \{0,1\}^n$, the value $h(x)$, for a random $h \in \mathcal{H}_{n,k}$, is uniform over $\{0,1\}^k$, and (2) for every $x \neq y \in \{0,1\}^n$, the values $h(x)$ and $h(y)$, for a random $h \in \mathcal{H}_{n,k}$, are independent and uniform over $\{0,1\}^k$. Our Circuit-SAT algorithm is as follows:

Given a Circuit-SAT instance C on k inputs of size $2^{o(k)}$,

1. pick a random hash function $h\colon \{0,1\}^n \to \{0,1\}^k$ from the universal hash family $\mathcal{H}_{n,k}$, and build a circuit for the following function f': for every $x \in \{0,1\}^n$, set

$$f'(x) = \begin{cases} f^*(x) & \text{if } f^*(x) = 0 \\ f^*(x) \oplus C(h(x)) & \text{otherwise} \end{cases}$$

2. Run the white-box BPP algorithm to decide $F(f')$. If $F(f') = 0$, output "C is satisfiable"; otherwise, output "C is unsatisfiable".

For the time analysis, note that the circuit size for f' defined above is $O(s^{o(1)}) + \mathsf{poly}(n) \le O(s^{o(1)})$, as $f'(x) = f^*(x) \wedge \neg C(h(x))$, and h has a circuit of size $\mathsf{poly}(n)$.

Thus, the described algorithm runs in time $\mathsf{poly}(s^{o(1)}) \le s^{o(1)}$, which is $2^{o(k)}$ for k-input $\mathsf{Circuit\text{-}SAT}$ instances C.

For correctness, note that if C is unsatisfiable, then $f' = f^*$, and so $F(f') = 1$. If C is satisfiable, say by an assignment $y \in \{0,1\}^k$, then, with probability at least $1/2$ over the choice of h, the set $h^{-1}(y)$ will contain at least one sensitive location $x \in \{0,1\}^n$ such that $f^*(x) = 1$, but flipping f^* at x results in the new function g such that $F(g) = 0$.

By monotonicity of F, flipping f^* at x *and at any other locations* x' *where* $f(x') = 1$ results in a new function f' such that $F(f') = 0$. $\square$

5.2 Win-win analysis

As the Sensitivity Conjecture is true for monotone properties, assuming that a monotone property F requires high decision tree complexity (i.e., non-uniform black-box complexity) implies that F has a (not necessarily easy) sensitive input. We use a "win-win" argument to prove the following.

Theorem 5.2. *Let F be any monotone property such that*

1. *F is in P in the white-box setting, but,*
2. *for almost all input lengths n, F requires decision tree complexity at least $s > n^{\omega(1)}$ on inputs $f\colon \{0,1\}^n \to \{0,1\}$.*

Then either $\mathsf{Circuit\text{-}SAT}_{n,2^{o(n)}} \in \mathsf{SIZE}(2^{o(n)})$ *infinitely often, or* $\mathsf{BPP} \subseteq \mathsf{NP}$.

Proof. We get that F has inputs $f\colon \{0,1\}^n \to \{0,1\}$ of sensitivity at least $s' \ge s^{\Omega(1)}$, for almost all n. By monotonicity of F, we may assume, without loss of generality, that such sensitive inputs f are minimal for F, meaning that for f such that $F(f) = 1$, every $x \in \{0,1\}^n$ where $f(x) = 1$ is a sensitive location (and similarly for f such that $F(f) = 0$, for x's where $f(x) = 0$). Below, for simplicity, we assume that f is such that $F(f) = 1$; the other case is symmetric.

It follows that the truth table of f is fully determined by $s'' = s' \cdot n$ bits needed to describe s' locations $x \in \{0,1\}^n$ where $f(x) = 1$. Think of this s''-bit string as a truth table of a new boolean function g on $k = \log s''$ bits. If such a function g is computable by a circuit of size $(s'')^{o(1)} \le 2^{o(k)}$, then the original function f also has a circuit of size $(s')^{o(1)}$ and sensitivity s', yielding a $2^{o(k)}$ circuit for $\mathsf{Circuit\text{-}SAT}$ instances on k-bit inputs, as in the proof of Theorem 5.1 above.

If such easy sensitive inputs f exist for infinitely many input lengths n, we get that $\mathsf{Circuit\text{-}SAT}$ is infinitely often decided by circuits of subexponential size. Otherwise, we get an efficient test that only accepts s''-bit strings of circuit complexity $(s'')^{\Omega(1)}$. The test needs to check that a given s''-bit string Z encodes a description of a function $f\colon \{0,1\}^n \to \{0,1\}$ such that $F(f) = 1$ and f is a minimal input for F. To this end, the test first builds a circuit of size $\mathsf{poly}(s'')$ deciding the boolean function f such that $f(x) = 1$ iff $x \in \{0,1\}^n$ is one of the strings described by Z. Then we use the white-box P algorithm for F to decide if $F(f) = 1$, and if every f' that differs from f in exactly one position $x \in \{0,1\}^n$ where $f(x) = 1$ is such that $F(f') = 0$. For the latter, we need to construct at most s'' circuits for each such f', and each such f' has a circuit of size at most $O(s'')$. Thus the overall runtime of the test is $\mathsf{poly}(s'')$.

Once we have an efficient test for truth tables of exponential circuit complexity for almost all input lengths, we conclude by the "hardness-randomness" trade-offs [NW94, BFNW93, IW97] that $\mathsf{BPP} \subseteq \mathsf{NP}$ (as we can nondeterministically guess and efficiently certify a truth table of exponential circuit complexity, and then use it to derandomize any BPP algorithm with only polynomial blowup in the run time). □

6 Circuit-SAT algorithm from variants of MCSP

So far we have considered a Circuit-SAT algorithm that relies on the sensitivity of a given counterexample F to the BBH. In this section we will show a different approach to designing Circuit-SAT algorithm from properties that are subsets of easy functions, the one that does not explicitly use the notion of sensitivity.[2]

We consider the following *succinct* version of MCSP, denoted SuccinctMCSP, where one is given a circuit as input, and is asked to determine if there is a smaller circuit computing the same boolean function; see, e.g., [AHK17] for a recent use and some basic results about SuccinctMCSP. More formally, for $t = t(n)$, $\mathsf{SuccinctMCSP}_t(C)$ asks to decide if $f = [C]$ is in $\mathsf{SIZE}(t)$.

Theorem 6.1. *For any efficiently computable $t(n) \in \omega(n)$, if $\mathsf{SuccinctMCSP}_t \in \mathsf{BPP}$, then*

$$\mathsf{Circuit\text{-}SAT}_{n,m} \in \mathsf{RTIME}(\mathsf{poly}(t(n), m)).$$

Proof. Let $F = \{tt(f) \mid f \in \mathsf{SIZE}(t(n))\}$. As before, we describe only a BPP algorithm for Circuit-SAT. The algorithm to decide if a circuit $C(x_1, \ldots, x_n)$ is satisfiable is as follows:

1. Take a random boolean function h on $r := k \log t(n)$ variables, for k such that $t(n + k \log t(n)) \ll t(n)^k/(k \log t(n))$.
2. Construct a circuit C_r computing h, of size $O(2^r)$.
3. Construct the circuit
$$C'(x_1, \ldots, x_n, y_1, \ldots, y_r) = C(x_1, \ldots, x_n) \wedge C_r(y_1, \ldots, y_r).$$
4. Test if $C' \in \mathsf{SuccinctMCSP}_t$. If 'Yes', output 'Unsatisfiable'; otherwise, output 'Satisfiable'.

For the analysis, observe that if C is unsatisfiable, then $[C'] \equiv 0$, and thus $size([C']) \leq t(n + k \log t(n))$. So, in this case, we have $[C'] \in F$. Next, with high probability over the choice of an r-input random boolean function h, the circuit complexity of h is at least $2^r/r$, and so $[C_r] \notin F$. For each satisfying assignment $a_1, \ldots, a_n$ to the inputs to C, we have that $C'(a_1, \ldots, a_n, y_1, \ldots, y_r) = C_r(y_1, \ldots, y_r)$. In particular, $size([C']) \geq size([C_r]) > t(n + k \log t(n))$. So, if C is satisfiable, $[C'] \notin F$. □

[2]Of course, as noted earlier, Simon's lemma implies that any such property does have an easy sensitive input, and so one can use the sensitivity-based Circuit-SAT algorithm described above. The point here, however, is to have a different type of a Circuit-SAT algorithm.

Theorem 6.2. *Let F be a non-empty (for all n) property that contains only a subset of functions $f \in \mathsf{SIZE}(t(n))$, for some efficiently computable $t(n) \in \omega(n)$. If $F \in \mathsf{wbBPP}$, then*

$$\mathsf{Circuit\text{-}SAT}_{n,m} \in \mathsf{RTIME}(\mathsf{poly}(t(n), m))/t(n).$$

Proof. The structure of this proof resembles the original Rice's theorem proof, with the main idea as in Theorem 6.1. Suppose we are given a circuit C on n variables. Construct circuit C_r on $k \log t(n)$ variables for a random function as in Theorem 6.1. Now, $C(x_1, \ldots, x_n) \wedge C_r(y_1, \ldots, y_r)$ has the circuit complexity at least $t(n+r)$ iff C is satisfiable; in this case, $[C \wedge C_r] \notin F$. When C is unsatisfiable, circuit $C \wedge C_r$ is unsatisfiable. However, now it is possible that the constant 0 function is not in F. In this case, the algorithm needs to know a circuit C_f on $n+r$ variables for some $f \in F$. Given such a circuit as advice, consider the new circuit

$$C' = (C(\vec{x}) \wedge C_r(\vec{y})) \oplus C_f(\vec{x}, \vec{y}).$$

Clearly, this C' still likely has high circuit complexity when C is satisfiable; however, when C is unsatisfiable, C' is equivalent to C_f, and so is in F. Thus, with high probability, we have $[C'] \notin F$ iff C is satisfiable. □

7 BBH for restricted circuit classes

We formulated the BBH with general circuits as inputs to the white-box algorithm. It is natural to consider its variants with other types of circuits. Already, [BGI+12] have shown that the BBH becomes false with BPP replaced by PromiseBPP (assuming one-way functions exist); here both the property F and the class of allowable input functions come from the unobfuscatable circuit ensemble used in the proof of impossibility of obfuscation.

We observe that for a very weak type of circuits, e.g., read-once branching programs, the corresponding version of the BBH is unconditionally false. This is because the satisfiability of a given roBP is exactly the reachability problem in the graph representing the roBP, and so SAT for roBPs is in P in the white-box setting. However, SAT for roBPs is still exponentially hard in the black-box setting.

Note that if a model of computation is exactly and properly learnable by a randomized algorithm using polynomially many membership queries to the function, then the BBH holds for that model. For example, since Angluin et al. [AHK93] showed that monotone read-once formulas are learnable in $O(n^3)$ time with $O(n^2)$ queries, the BBH *holds* for *monotone read-once formulas* in place of circuits.

What about other natural circuit classes? We observe that for AC^0 circuits as inputs to the white-box algorithm, some of our results still hold. Let us call a property F of n-variate boolean functions an *s-strong counterexample to the AC^0-BBH* if the following conditions hold:

1. there is a BPP algorithm that, given an AC^0 circuit C, decides if $[C] \in F$.

2. for almost all n, F has an input $f^* \colon \{0,1\}^n \to \{0,1\}$ computable by an AC^0 circuit of size $2^{o(n)}$ such that $\mathsf{sens}(F, f^*) \geq s$.

Below we denote by $\mathsf{AC}^0\text{-}\mathsf{Circuit\text{-}SAT}_{n,m}$ the satisfiability problem for n-input AC^0 circuits of size at most m. We have the following analog of Lemma 4.2.

Theorem 7.1. *If there is an s-strong counterexample to the* AC^0*-BBH, then*

$$\mathsf{AC}^0\text{-}\mathsf{Circuit}\text{-}\mathsf{SAT}_{n,m} \in \mathsf{SIZE}(\mathsf{poly}(2^{o(n)}, m) \cdot 2^n/s(n).$$

Proof. The algorithm given in Lemma 4.2 can be adapted for AC^0 circuits. The main ingredient there is the Valiant-Vazirani reduction that needs circuits for computing the parities of (random) subsets of n input bits (random hash functions). Each such parity is computable by a depth-d AC^0 circuit of size $O(2^{n^{1/(d-1)}})$, which is $2^{o(n)}$ for $d > 2$. □

Theorem 7.1 yields a nontrivial $\mathsf{Circuit}\text{-}\mathsf{SAT}$ algorithm for AC^0 circuits when the sensitivity bound s is at least $\Omega(2^n)$. In particular, we get the following.

Corollary 7.2. *If a symmetric property is a counterexample to the* AC^0*-BBH, then*

$$\mathsf{AC}^0\text{-}\mathsf{Circuit}\text{-}\mathsf{SAT}_{n,m} \in \mathsf{SIZE}(\mathsf{poly}(2^{o(n)}, m)).$$

Proof. By Lemma 3.4, we know that a symmetric property has an input $f\colon \{0,1\}^n \to \{0,1\}$ of sensitivity $2^n/2$, which is computable by a polynomial-size AC^0 circuit. The required conclusion now follows by Theorem 7.1. □

For $s \geq 2^{\Omega(n)}$, we are unable to amplify the success probability here using the recursive approach of [PP10], as that approach doesn't preserve the AC^0-type of input circuits. We leave it as an open question to handle the case of $s \geq 2^{\Omega(n)}$.

8 Conclusions

We conjecture that the falsehood of the BBH is equivalent to the easiness of $\mathsf{Circuit}\text{-}\mathsf{SAT}$. In the present paper, we make a step in that direction, but many interesting questions remain open. Below we list a few of them.

1. Is it possible to prove our conjecture, assuming the Sensitivity Conjecture is true?
2. Is it possible to get a *uniform* algorithm for $\mathsf{Circuit}\text{-}\mathsf{SAT}$ for a general class of counterexamples to BBH, thereby (conditionally) violating the ETH?
3. Are there any algorithmic SAT consequences from the assumption that there is a strong counterexample to the BBH for *CNF formulas* (rather than AC^0 or general circuits)?
4. The initial formulation of BBH by Barak et al. [BGI+12] was mainly inspired by the idea of virtual black-box obfuscation. Is it possible to use indistinguishability obfuscators for proving or disproving BBH?

Acknowledgements. We are very grateful to Marco Carmosino, Shachar Lovett and Avi Wigderson for many insightful discussions. We also thank Rahul Santhanam for his comments and suggestions on the earlier version of this work.

References

[Adl78] Leonard Adleman. Two theorems on random polynomial time. In *Proceedings of the Nineteenth Annual IEEE Symposium on Foundations of Computer Science*, pages 75–83, 1978. 9

[AHK93] Dana Angluin, Lisa Hellerstein, and Marek Karpinski. Learning read-once formulas with queries. *Journal of the ACM (JACM)*, 40(1):185–210, 1993. 14

[AHK17] Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. *Computational Complexity*, 26(2):469–496, 2017. 13

[AV15] Andris Ambainis and Jevgēnijs Vihrovs. Size of sets with small sensitivity: A generalization of Simon's lemma. In *International Conference on Theory and Applications of Models of Computation*, pages 122–133. Springer International Publishing, 2015. 6

[BdW02] Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, Oct 2002. 6

[BFNW93] Laci Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993. 5, 13

[BGI$^+$12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im) possibility of obfuscating programs. *Journal of the ACM (JACM)*, 59(2):6, 2012. 1, 2, 4, 7, 14, 15

[BS00] Bernd Borchert and Frank Stephan. Looking for an analogue of Rice's theorem in circuit complexity theory. *Math. Log. Q.*, 46(4):489–504, 2000. 4, 8

[CSV84] Ashok K Chandra, Larry Stockmeyer, and Uzi Vishkin. Constant depth reducibility. *SIAM Journal on Computing*, 13(2):423–439, 1984. 9

[HKP11] Pooya Hatami, Raghav Kulkarni, and Denis Pankratov. Variations on the sensitivity conjecture. *Theory of Computing, Graduate Surveys*, 2:1–27, 2011. 6

[HR00] Lane A. Hemaspaandra and Jörg Rothe. A second step towards complexity-theoretic analogs of Rice's theorem. *Theor. Comput. Sci.*, 244(1-2):205–217, 2000. 4

[HT04] Lane A. Hemaspaandra and Mayur Thakur. Lower bounds and the hardness of counting properties. *Theor. Comput. Sci.*, 326(1-3):1–28, 2004. 4

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. 2

[IW97] Russell Impagliazzo and Avi Wigderson. P=BPP if E requires exponential circuits: Derandomizing the XOR Lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 220–229, 1997. 5, 13

[Nis91] Noam Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991. 2, 6, 8

[NW94] Noam Nisan and Avi Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994. 5, 13

[PP10] Ramamohan Paturi and Pavel Pudlák. On the complexity of circuit satisfiability. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 241–250, 2010. 5, 10, 15

[Sim83] Hans-Ulrich Simon. A tight ω (loglog n)-bound on the time for parallel RAM's to compute nondegenerated boolean functions. In *Foundations of Computation Theory*, pages 439–444. Springer, 1983. 6

[VV86] Leslie Valiant and Vijay Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47:85–93, 1986. 4, 9