

Certification with an NP Oracle

Guy Blanc

Department of Computer Science, Stanford University, CA, USA

Caleb Koch

Department of Computer Science, Stanford University, CA, USA

Jane Lange

Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, USA

Carmen Strassle

Department of Computer Science, Stanford University, CA, USA

Li-Yang Tan

Department of Computer Science, Stanford University, CA, USA

Abstract

In the *certification problem*, the algorithm is given a function f with certificate complexity k and an input x^* , and the goal is to find a certificate of size $\leq \text{poly}(k)$ for f 's value at x^* . This problem is in NP^{NP} , and assuming $\text{P} \neq \text{NP}$, is not in P . Prior works, dating back to Valiant in 1984, have therefore sought to design efficient algorithms by imposing assumptions on f such as monotonicity.

Our first result is a BPP^{NP} algorithm for the general problem. The key ingredient is a new notion of the *balanced* influence of variables, a natural variant of influence that corrects for the bias of the function. Balanced influences can be accurately estimated via uniform generation, and classic BPP^{NP} algorithms are known for the latter task.

We then consider certification with stricter *instance-wise* guarantees: for each x^* , find a certificate whose size scales with that of the smallest certificate for x^* . In sharp contrast with our first result, we show that this problem is NP^{NP} -hard even to approximate. We obtain an optimal inapproximability ratio, adding to a small handful of problems in the higher levels of the polynomial hierarchy for which optimal inapproximability is known. Our proof involves the novel use of bit-fixing dispersers for gap amplification.

2012 ACM Subject Classification Theory of computation $\rightarrow$ Computational complexity and cryptography

Keywords and phrases Certificate complexity, Boolean functions, polynomial hierarchy, hardness of approximation

Digital Object Identifier 10.4230/LIPIcs.ITCS.2023.18

Related Version *Full Version:* <https://arxiv.org/abs/2211.02257>

Funding Guy, Caleb, Carmen, and Li-Yang are supported by NSF awards 1942123, 2211237, and 2224246. Jane is supported by NSF Award 2006664. Caleb is also supported by an NDSEG fellowship.

Acknowledgements We thank the ITCS reviewers for their useful comments and feedback.

1 Introduction

For a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and an input $x^* \in \{0, 1\}^n$, a *certificate* for f 's value at x^* is a set $S \subseteq [n]$ of coordinates such that:

$$f(x^*) = f(y) \quad \text{for all } y \text{ such that } y_S = x^*_S.$$

This is a set of coordinates that fully determines f 's value on x^* , and coordinates outside this set are irrelevant in the sense that changing any of them in any way cannot change f 's value.



© Guy Blanc, Caleb Koch, Jane Lange, Carmen Strassle, and Li-Yang Tan;
licensed under Creative Commons License CC-BY 4.0

14th Innovations in Theoretical Computer Science Conference (ITCS 2023).

Editor: Yael Tauman Kalai; Article No. 18; pp. 18:1–18:22



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We write $\text{Cert}(f, x^*)$ to denote the size of a smallest certificate for f 's value at x^* , and $\text{Cert}(f)$ to denote $\max_{x \in \{0,1\}^n} \{\text{Cert}(f, x)\}$, the certificate complexity of f . Certificate complexity is among the most basic and well-studied measures of boolean function complexity [6, 13].

With the notion of certificates in mind, an algorithmic question suggests itself: can we design efficient algorithms for finding small certificates? This leads us to the certification problem:

Certification Problem: Given the succinct description of a function f and an input $x^* \in \{0,1\}^n$, find a certificate of size $\leq \text{poly}(\text{Cert}(f))$ for f 's value at x^* .

The intractability of certification.

Valiant was the first to consider the certification problem [29]. He observed that it is likely intractable in its full generality, since even the task of *verifying* the output of a certification algorithm, i.e. checking that a purported certificate is indeed a certificate, is coNP -complete.

Furthermore, the NP-hardness of SAT easily implies the following:

► **Fact 1.** *Assuming $P \neq \text{NP}$, there is no efficient algorithm for the certification problem, even if the algorithm only has to return a certificate of size $\Phi(\text{Cert}(f))$ for any growth function $\Phi : \mathbb{N} \rightarrow \mathbb{N}$. Similarly, we can rule out randomized algorithms under the assumption that $\text{NP} \not\subseteq \text{BPP}$.*

For completeness, we include the proofs of Valiant's observation and Fact 1 in Section 4.

Prior certification algorithms.

In light of these intractability results, prior certification algorithms have relied on assumptions on f . Valiant gave a simple and efficient algorithm for *monotone* functions, which he used as a subroutine for PAC learning monotone DNF formulas (see also [1]). Recent works [4, 11] give new certification algorithms for monotone functions that are furthermore highly query efficient, with the latter paper by Gupta and Manoj achieving the optimal query complexity. In a separate line of work, Barceló, Monet, Pérez, and Subercaseaux [2] gave an efficient certification algorithm for *halfspaces*.

While the focus of these works and ours is theoretical in nature, there has also been a recent surge of interest in certification algorithms from an applied perspective. Motivation here comes from the growing field of explainable machine learning, where one thinks of f as a complicated model (e.g. a neural net) and small certificates as succinct explanations for its decisions. In this literature, certificates are more commonly referred to as “sufficient reasons” [20] and “anchors” [17]; see [2, 5, 4] for further discussions and references regarding this.

1.1 Our results

Departing from the theme of prior works, we consider the certification problem in its full generality, without any assumptions on f , but allow algorithms that do not necessarily run in polynomial time. There is a simple NP^{NP} algorithm for general problem: first guess a certificate for f 's value at x^* , and then use the NP oracle to verify that it is indeed a certificate. Therefore, the certification problem lies within (the function version of¹) NP^{NP} , and assuming $P \neq \text{NP}$, lies outside of P. This leaves a rather wide gap between P and NP^{NP} .

¹ Throughout this paper we conflate the decision and function versions of complexity classes, except in instances where there is a possibility of confusion.

At first glance, one may suspect that the true complexity of the certification problem is NP^{NP} . Afterall, the definition of certificate complexity inherently involves two quantifiers: *there exists a set S such that $f(x^*) = f(y)$ for all y that agrees with x^* on S* . Our first main result counters this intuition with a BPP^{NP} algorithm:

► **Theorem 2.** *There is a randomized polynomial-time algorithm with NP oracle access that takes a polynomial-size circuit representation of $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and a string $x^* \in \{0, 1\}^n$, and outputs w.h.p. a certificate $S \subseteq [n]$ for f 's value on x^* satisfying $|S| = O(\text{Cert}(f)^5)$.*

Our algorithm does not need to be given the value of $\text{Cert}(f)$ as an input. As an example setting of parameters, for a function f with certificate complexity $O(n^\epsilon)$ our algorithm always returns a certificate of size $O(n^{5\epsilon}) \ll n$. We note that the class of functions with certificate complexity $O(n^\epsilon)$ is very expressive and includes, among other functions, all decision trees of depth $O(n^\epsilon)$.

Our algorithm uses its NP oracle in two ways:

1. *Uniform generation.* Our algorithm uses as a key subroutine the ability to sample a uniform random satisfying assignment $x \sim f^{-1}(1)$ of f . Early and influential results of theoretical computer science show that this can be done efficiently with an NP oracle [12, 3].
2. *Verification.* Our randomized algorithm returns a set that is a certificate for f 's value at x^* with high probability. As mentioned, the task of verifying that a set is indeed a certificate is coNP -complete, and so we use the NP oracle for this purpose.

Regarding the BPP^{NP} algorithms for uniform generation, while they are not efficient in the traditional sense of worst-case complexity, there is an active line of research that seeks to make them as practical as possible: by replacing the NP oracle with state-of-the-art SAT solvers; optimizing the number of calls to these solvers; etc. See [30, 9] and the references therein.

Barriers to instance-wise guarantees

In the spirit of beyond worst-case analysis, it is natural to ask if our algorithm can be strengthened so its guarantees hold *instance-wise*: can it always return a certificate of size $\text{poly}(\text{Cert}(f, x^*))$ instead of $\text{poly}(\text{Cert}(f))$? There are two known lower bounds against such algorithms, both in the strictest setting where the algorithm has to return a certificate of size exactly $\text{Cert}(f, x^*)$. Gupta and Manoj [11] showed that any such algorithm for monotone functions must make $n^{\Omega(\text{Cert}(f, x^*))}$ many queries to f , in the setting where the algorithm only gets black-box queries to f rather than an explicit description. Barceló, Monet, Pérez, and Subercaseaux [2] showed that the decision version of the problem, where the goal is to decide if $\text{Cert}(f, x^*) \leq k$ for a given parameter k , is NP^{NP} -hard.

Neither of these results rule out algorithms that return a certificate of size $\leq \text{poly}(\text{Cert}(f, x^*))$ or even $O(\text{Cert}(f, x^*))$. Our second main result does so by showing that $\text{Cert}(f, x^*)$ is optimally NP^{NP} -hard to approximate:

► **Theorem 3.** *The following holds for every constant $\epsilon > 0$. Given a circuit representation of $f : \{0, 1\}^n \rightarrow \{0, 1\}$, $k \in \mathbb{N}$, and $x^* \in \{0, 1\}^n$, it is NP^{NP} -hard to distinguish between*

- YES: *there exists a certificate of size $\leq k$ for f 's value on x^* ;*
- NO: *Every certificate for f 's value on x^* has size $> k \cdot n^{1-\epsilon}$.*

There is by now a sizeable number of problems that are known to be hard for higher levels of the polynomial hierarchy [19, 18]. However, relatively few of them are known to be hard to *approximate*, and yet fewer for which optimal inapproximability ratios have been

established. The papers [24, 25, 28, 26, 23, 14] cover many of the existing results; see also the survey [27]. Theorem 3 thus adds an additional entry into this catalogue of natural problems known to be inapproximable for higher levels of the polynomial hierarchy.

2 Proof overviews

2.1 Overview of the proof of Theorem 2

Minimality no longer suffices

As mentioned, many prior certification algorithms have focused on the class of *monotone* functions [29, 1, 4, 11]. An especially nice feature of this setting is that it suffices to find *minimal* certificates: a certificate S such that no strict subset $S' \subsetneq S$ is a certificate:

► **Fact 4** (Minimality suffices for monotone functions). *If f is monotone, for every x^* , every minimal certificate S for f 's value at x^* has size $|S| \leq \text{Cert}(f)$.*

This simple fact is crucially used in all prior certification algorithms for monotone functions. While $\text{Cert}(f)$ is a global property of f , the minimality of a specific certificate can be recognized locally: if at any point we have a candidate certificate S such that dropping any of its coordinates results in it no longer being a certificate, Fact 4 tells us that $|S| \leq \text{Cert}(f)$ and we are done.

Unfortunately, Fact 4 is false for general functions: the size of a minimal certificate could be exponentially larger than its certificate complexity:

► **Fact 5** ([13, Exercise 1.7]). *For each $n \in \mathbb{N}$, there is a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and $x^* \in \{0, 1\}^n$ such that x^* has a minimal certificate $S \subseteq [n]$ of size $|S| \geq \Omega(n)$ even though $\text{Cert}(f) \leq O(\log n)$.*

The implication of Fact 5 for the certification problem is that algorithms for general functions can get stuck at (very bad) local minima, which necessitates a more global understanding of the structure of functions with low certificate complexity. We include the proofs of Fact 4 and Fact 5 in Section 4.

Our algorithm and its main subroutine

We now describe our algorithm. Its main subroutine is a BPP^{NP} algorithm for finding a restriction to a small number of variables under which f becomes constant:

► **Lemma 6.** *Given a succinct representation of $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and an NP oracle, there is a randomized polynomial-time algorithm that w.h.p. finds a restriction π to $O(\text{Cert}(f)^4)$ variables such that f_π is a constant function.*

Such an algorithm can be viewed as one that finds a certificate for *some* input of f (any of the inputs that are consistent with π), but not necessarily the specific input x^* that we are interested in certifying. We then convert such an algorithm into an actual certification algorithm by calling it $2 \text{Cert}(f)$ times, resulting in a certificate for x^* of size $O(\text{Cert}(f)^5)$. This conversion algorithm is a fairly straightforward consequence of a basic property of certificates, that every 1-certificate and every 0-certificate must share at least one variable.

Balanced influences

To describe our algorithm for Lemma 6, we need a new notion of the *balanced influence* of variables. Recall that the *influence* of a variable i on f is the quantity $\text{Inf}_i(f) := \Pr_{\mathbf{x} \sim \{0,1\}^n} [f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})]$, where $\mathbf{x}$ is uniform random and $\mathbf{x}^{\oplus i}$ denotes $\mathbf{x}$ with its i -th bit flipped. It is easy to see that $\text{Inf}_i(f) \leq \text{Var}(f)$, and so the more unbalanced f is, the smaller its influences are. Balanced influence corrects for this by measuring influence with respect to a distribution $\mathcal{D}_{\text{bal}}^{(f)}$ that places equal weight on $f^{-1}(1)$ and $f^{-1}(0)$:

To sample $\mathbf{x} \sim \mathcal{D}_{\text{bal}}^{(f)}$: first sample $\mathbf{b} \sim \{0,1\}$ uniformly and then $\mathbf{x} \sim f^{-1}(\mathbf{b})$ uniformly.

► **Definition 7** (Balanced influences). *The balanced influence of a variable $i \in [n]$ on $f : \{0,1\}^n \rightarrow \{0,1\}$ is the quantity:*

$$\text{BALINF}_i(f) := \Pr_{\mathbf{x} \sim \mathcal{D}_{\text{bal}}^{(f)}} [f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})].$$

Though simple, this variant of influence does not appear to have been explicitly considered before. This definition syncs up nicely with the notion of uniform generation: an algorithm for uniform generation can be used to efficiently estimate balanced influences to high accuracy. Given the classic BPP^{NP} algorithms for uniform generation [12, 3], we therefore get:

► **Lemma 8.** *There is a randomized algorithm that, given a succinct representation of $f : \{0,1\}^n \rightarrow \{0,1\}$, an NP oracle, $i \in [n]$ and $\varepsilon > 0$, runs in $\text{poly}(n, 1/\varepsilon)$ time and w.h.p. outputs an estimate $\eta_i = \text{BALINF}_i(f) \pm \varepsilon$.*

We can now state our algorithm for Lemma 6. It is simple and proceeds by iteratively and randomly restricting the variables of f with the largest balanced influence:

1. Estimate $\text{BALINF}_i(f)$ for each $i \in [n]$ to within $\pm \frac{1}{4n}$ and let i^* be the index with the largest estimate.
2. Randomly restrict $x_{i^*} \leftarrow \mathbf{b}$ where $\mathbf{b} \sim \{0,1\}$ is uniform random.
3. Recurse on $f_{x_{i^*}=\mathbf{b}}$.

We prove that f becomes constant w.h.p. within $O(\text{Cert}(f)^5)$ many iterations of this algorithm. It is crucial that we work with *balanced* influence here: this same algorithm fails (i.e. it requires $\Omega(n)$ many iterations before f becomes constant) if it is instead run on estimates of the usual notion of influence.²

2.2 Overview of the proof of Theorem 3

Monotone Minimum Weight Word and its inapproximability

We prove Theorem 3 by reducing from the MONOTONE MINIMUM WEIGHT WORD problem. To state this problem we first define *co-nondeterministic* circuits:

► **Definition 9** (Co-nondeterministic circuit). *A co-nondeterministic circuit $C : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$ accepts an input $x \in \{0,1\}^n$ iff for all $y \in \{0,1\}^n$ $C(x, y) = 1$.*

² Indeed, since the usual notion of influence can be estimated to high accuracy without an NP oracle, we have by Fact 1 that no algorithm that is based only on estimates of usual influences can succeed unless $\text{NP} \subseteq \text{BPP}$.

We say that a co-nondeterministic circuit C *accepts a non-empty monotone set* if it accepts at least one x , and for every x that it accepts, it also accepts every x' such that $x' \succeq x$. (Note that $C : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ itself need not be a monotone function.)

► **Definition 10** (MONOTONE MINIMUM WEIGHT WORD). *The MONOTONE MINIMUM WEIGHT WORD (MMWW) problem is the following: given a co-nondeterministic circuit $C : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ that accepts a non-empty monotone set and an integer $k \in \mathbb{N}$, does C accept an input $x \in \{0, 1\}^n$ with at most k ones?*

Umans [24] showed that the MMWW problem is NP^{NP} -hard to approximate within $n^{\frac{1}{5}-\varepsilon}$. This inapproximability ratio was subsequently improved to the optimal $n^{1-\varepsilon}$ by Ta-Shma, Umans, and Zuckerman [23]:

► **Theorem 11** (GAPPED-MMWW is NP^{NP} -hard). *The following holds for every constant $\varepsilon > 0$. Given a co-nondeterministic circuit $C : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ that accepts a non-empty monotone set and an integer $k \in \mathbb{N}$, it is NP^{NP} -hard to distinguish between:*

- YES: C accepts an x with $\leq k$ ones;
- NO: Every x that C accepts has $> k \cdot n^{1-\varepsilon}$ ones.

First attempt at a reduction

To describe the intuition behind our reduction, we first consider what happens when we take an instance of GAPPED-MMWW, which is specified by a co-nondeterministic circuit $C : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ and an integer k , and map it to the certification problem with $f(x, y) = C(x, y)$ and the input to be certified being $(x^*, y^*) = (1^n, 1^n)$. We consider the two possible cases:

- If (C, k) is a YES instance of GAPPED-MMWW, it accepts an $x \in \{0, 1\}^n$ with $\leq k$ ones. Let $S \subseteq [n]$ be the size- k set of 1-coordinates of this input x . It is straightforward to verify that S is a certificate for f 's value on (x^*, y^*) , and so $\text{Cert}(f, (x^*, y^*)) \leq k$.
- If (C, k) is a NO instance of GAPPED-MMWW, we would like it to be the case that $\text{Cert}(f, (x^*, y^*)) > k \cdot n^{1-\varepsilon}$. Let $S \subseteq [n] \times [n]$ be a smallest certificate for f 's value on (x^*, y^*) . If S only contains coordinates of x^* , it is again straightforward to verify that C accepts the input $x \in \{0, 1\}^n$ that is the indicator vector of S , and so $|S| > k \cdot n^{1-\varepsilon}$ as desired. However, S may contain one of more coordinates of y^* , and in that case we do not have a lower bound on its size.

The actual reduction

Intuitively, we would like to fix this issue by modifying C so that it becomes disproportionately more “expensive” to include y^* -coordinates in the certificate compared to x^* -coordinates, in the sense that including even a single y^* -coordinate contributes $> k \cdot n^{1-\varepsilon}$ to the size of the certificate, compared to just one in the case of single x^* -coordinate. We accomplish this with this use of *bit-fixing dispersers*, a well-studied construct in the pseudorandomness literature.

Given an instance $C : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ of GAPPED-MMWW, we map it to an instance of the certification problem where the function $f : \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}$ is:

$$f(x, z) := C(x, D(z)),$$

and $D : \{0, 1\}^m \rightarrow \{0, 1\}^n$ satisfies the following properties:

1. **D is explicit.** There is an efficient algorithm that, given m and n , produces the circuit description of $D : \{0, 1\}^m \rightarrow \{0, 1\}^n$ in $\text{poly}(m, n)$ time. This is so that our reduction is efficient.

2. **D retains full image under restrictions.** For any set $S \subseteq [m]$ of size $\leq k \cdot n^{1-\varepsilon}$ and any assignment $u \in \{0,1\}^{|S|}$, the image of $D_{S \leftarrow u}$ is all of $\{0,1\}^n$: for any $y \in \{0,1\}^n$ there is an $z \in \{0,1\}^{m-|S|}$ such that $D_{S \leftarrow u}(z) = y$. This ensures that one must fix $> k \cdot n^{1-\varepsilon}$ many z -variables of f in order to fix even a single y -variable of C .
3. **Small m .** For the second property to hold m certainly has to be at least n . Since f is a function over $m+n$ variables, would like m to be as close to n as possible in order to preserve the $n^{1-\varepsilon}$ ratio of GAPPED-MMWW.

A simple construction of a function satisfying the first two properties is the blockwise parity function:

$$\text{BlockwisePar} : (\{0,1\}^\ell)^n \rightarrow \{0,1\},$$

$$\text{BlockwisePar}(z^{(1)}, \dots, z^{(n)})^n = (\oplus_{j=1}^\ell z_j^{(1)}, \dots, \oplus_{j=1}^\ell z_j^{(n)})$$

where $\ell = k \cdot n^{1-\varepsilon} + 1$. In this case $m = \ell \cdot n = \Theta(n^{2-\varepsilon})$, and so the $n^{1-\varepsilon}$ ratio of GAPPED-MMWW translates into a gap of $n^{\frac{1}{2}-\varepsilon}$ for the certification problem.

Functions satisfying the first two properties are known as zero-error bit-fixing dispersers in the pseudorandomness literature. The current best construction, due to Gabizon and Shaltiel [10], gives $m = O(n)$ for our setting of parameters, and so using it in place of BlockwisePar enables us to achieve the optimal inapproximability ratio of $n^{1-\varepsilon}$, thereby yielding Theorem 3.

3 Discussion and future work

Since the certification problem is intractable unless $P = NP$, to further understand it we must either rely on assumptions about f or allow algorithms that are not necessarily efficient in the traditional sense of running in polynomial time. Complementing previous works that take the former route, in this work we consider the latter option and study the complexity of certification *relative to an NP oracle*, giving new algorithmic (BPP^{NP}) and hardness (NP^{NP}) results. Our motivation is twofold. First, given how natural the problem is, we believe that it is of independent interest to understand its inherent complexity. Second, given the empirical success of SAT solvers and their increasing adoption in a variety of real-world algorithmic tasks, the distinction between problems in BPP^{NP} (“exists an efficient algorithm assuming all calls to the SAT solver run quickly”) versus those that are hard for NP^{NP} (“intractable even if all calls to the SAT solver run in unit time”) is especially relevant in this context.

We list a couple of concrete open problems suggested by our work. A natural one is to improve on the bound of $O(\text{Cert}(f)^5)$ given by Theorem 2:

► **Open Problem 1.** *Is there a BPP^{NP} certification algorithm that returns certificates of length $O(\text{Cert}(f))$?*

Next, Theorem 3 shows that given f and x^* , it is optimally NP^{NP} -hard to approximate the size of the smallest certificate for f ’s value at x^* , i.e. to approximate the quantity $\text{Cert}(f, x^*)$. Since there is an NP^{NP} algorithm that computes it exactly, this settles the complexity of the problem. It is equally natural to consider the problem of computing/approximating $\text{Cert}(f)$. There is a simple Π_3 algorithm that computes it exactly. However, little is known in terms of lower bounds:

► **Open Problem 2.** *Is certificate complexity Π_3 -hard to compute and what is the complexity of approximating this quantity?*

Concluding on a speculative note, in the spirit of *interactive proofs*, it would be interesting to extend our BPP^{NP} algorithm to the setting where the algorithm interacts with a powerful but *untrusted* oracle. More broadly, there should be much to be gained by bringing techniques from interactive proofs to bear on problems, such as certification, that are motivated by explainable machine learning. Such an “interactive theory of explanations” was listed as a specific direction in the 2020 TCS Visioning Report [7] and aligns well with ongoing efforts in machine learning [31], but to our knowledge has thus far not been explored much.

4 Basic results regarding the certification problem

4.1 The necessity of an NP oracle

Verifying a certificate is coNP-complete

Given a candidate certificate for an input, verifying the certificate is computationally hard. Indeed, we observe that such a problem is coNP-complete. This lemma shows that we at least need access to a coNP oracle or equivalently an NP oracle.

► **Definition 12** (VERIFYCERT). *Given a polynomial-size circuit for $f : \{0, 1\}^n \rightarrow \{0, 1\}$, input $x \in \{0, 1\}^n$, and candidate certificate $S \subseteq [n]$, decide whether S is indeed a certificate for f ’s classification of x .*

► **Lemma 13** (Observed in [29]). *VERIFYCERT is coNP-complete.*

To prove this lemma, we give a reduction from the canonical coNP-complete problem TAUTOLOGY.

► **Definition 14** (TAUTOLOGY [8]). *Given a Boolean formula $\varphi : \{0, 1\}^n \rightarrow \{0, 1\}$, decide whether φ is a tautology: $\varphi(x) = 1$ for all $x \in \{0, 1\}^n$.*

Proof of Lemma 13. First, we show that $\text{VERIFYCERT} \in \text{coNP}$. Specifically, we observe that $\overline{\text{VERIFYCERT}} \in \text{NP}$. Given $f : \{0, 1\}^n \rightarrow \{0, 1\}$, an input $x \in \{0, 1\}^n$, and a candidate $S \subseteq [n]$, one can nondeterministically guess a string y such that $x_S = y_S$ and $f(x) \neq f(y)$. This string y exists if and only if S is *not* a certificate for x .

Next, we show VERIFYCERT is coNP-hard via a reduction from TAUTOLOGY. Let $\varphi : \{0, 1\}^n \rightarrow \{0, 1\}$ be a Boolean formula. We show how to determine if φ is a tautology using VERIFYCERT . Let $x \in \{0, 1\}^n$ be arbitrary. If $\varphi(x) = 0$ output “not a tautology”. Otherwise, call VERIFYCERT on the instance $f = \varphi, S = \emptyset$ and x . If $\emptyset$ is a certificate for x then φ is a tautology: all $y \in \{0, 1\}^n$ satisfy $\varphi(y) = \varphi(x) = 1$. Otherwise, there is some input y for which $\varphi(y) = 0$. ◀

Solving the certificate finding problem efficiently without an NP oracle would prove $\text{P} = \text{NP}$ (Fact 1)

Next, we show that any efficient algorithm for certification can be used to solve $\text{GAPPEDCERT}(0, \Psi)$: the promise problem of distinguishing whether $\text{Cert}(f)$ is 0 or every input requires size- $\Psi(n)$ certificates. Perhaps surprisingly, $\text{GAPPEDCERT}(0, \Omega(n))$ turns out to be NP-hard. As a result, we should not expect certification to be efficiently solvable without access to an NP oracle.

► **Definition 15** ($\text{GAPPEDCERT}(0, \Psi)$). *Given a polynomial-size circuit for $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and a growth function $\Psi : \mathbb{N} \rightarrow \mathbb{N}$, the $\text{GAPPEDCERT}(0, \Psi)$ problem is to distinguish between*

- YES: $\text{Cert}(f, x) \geq \Psi(n)$ for all $x \in \{0, 1\}^n$;
- NO: $\text{Cert}(f) = 0$.

As a promise decision problem, any algorithm for $\text{GAPPEDCERT}(0, \Psi)$ is allowed to answer arbitrarily on input instances which do not fall into the two cases. Note that SAT reduces to $\text{GAPPEDCERT}(0, 1)$ since $\text{GAPPEDCERT}(0, 1)$ is equivalent to determining whether f is constant. To obtain hardness against the certification problem, we require hardness for GAPPEDCERT where the growth function is nonconstant.

► **Lemma 16.** *$\text{GAPPEDCERT}(0, \sqrt{n})$ is NP-hard.*

Proof. Let $\varphi : \{0, 1\}^n \rightarrow \{0, 1\}$ be a Boolean formula. Recall the definition of $\text{BlockwisePar} : (\{0, 1\}^\ell)^n \rightarrow \{0, 1\}^n$

$$\text{BlockwisePar}(z^{(1)}, \dots, z^{(n)}) = (\oplus_{j=1}^\ell z_j^{(1)}, \dots, \oplus_{j=1}^\ell z_j^{(n)}).$$

We let $f = \varphi \circ \text{BlockwisePar}$. For all $z \in (\{0, 1\}^\ell)^n$, we have

$$\text{Cert}(f, z) \geq \text{Cert}(\varphi, \text{BlockwisePar}(z)) \cdot \ell$$

since ℓ coordinates of f need to be fixed in order to fix one coordinate of φ . In particular, if φ is nonconstant, then $\text{Cert}(\varphi, \text{BlockwisePar}(z)) \geq 1$ and so $\text{Cert}(f, z) \geq \ell$ for all $z \in (\{0, 1\}^\ell)^n$.

Therefore, to determine satisfiability of φ it is sufficient to solve $\text{GAPPEDCERT}(0, \ell)$ for the function $f : (\{0, 1\}^\ell)^n \rightarrow \{0, 1\}$. Choosing $\ell = n$ yields the hardness in the lemma statement. ◀

Lemma 16 is sufficient to establish Fact 1. However, we also note that a slightly more technical proof yields near-optimal gapped hardness.

► **Lemma 17** (Near-optimal gapped hardness). *$\text{GAPPEDCERT}(0, \Omega(n))$ is NP-hard.*

We provide a proof of Lemma 17 can be found in the appendix of the full version of the paper.

► **Lemma 18.** *If there is a polynomial-time algorithm for the certification problem which returns certificates of size $\Phi(\text{Cert}(f))$ for some growth function $\Phi : \mathbb{N} \rightarrow \mathbb{N}$, then there is a polynomial-time algorithm for $\text{GAPPEDCERT}(0, \Psi)$ for all growth functions $\Psi : \mathbb{N} \rightarrow \mathbb{N}$.*

Proof. Suppose such an algorithm exists for the certification problem. Let $\Psi : \mathbb{N} \rightarrow \mathbb{N}$ be an arbitrary growth function and let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be an instance of $\text{GAPPEDCERT}(0, \Psi)$. Let $S \subseteq [n]$ be a certificate obtained by running the algorithm for the certification problem on f for any input $x^* \in \{0, 1\}^n$. Output “NO” if $|S| < \Psi(n)$ and “YES” if $|S| \geq \Psi(n)$.

The correctness of our output follows from the observation that $\text{Cert}(f, x) \leq |S| \leq \Phi(\text{Cert}(f))$ and so

$$\begin{aligned} |S| \geq \Psi(n) &\Rightarrow \Psi(n) \leq \Phi(\text{Cert}(f)) \\ |S| < \Psi(n) &\Rightarrow \text{Cert}(f, x) < \Psi(n). \end{aligned}$$

Assuming n is large enough, the first case implies $\text{Cert}(f) > 0$ which allows us to rule out being in the NO case of $\text{GAPPEDCERT}(0, \Psi)$. The second case rules out being in the YES case of $\text{GAPPEDCERT}(0, \Psi)$. ◀

Fact 1 follows as an immediate consequence of Lemmas 16 and 18: any polynomial-time algorithm for the certification problem yields a polynomial-time algorithm for $\text{GAPPEDCERT}(0, \sqrt{n})$ which implies $\text{NP} = \text{P}$. Likewise, any randomized polynomial-time algorithm for the certification problem yields a randomized polynomial-time algorithm for $\text{GAPPEDCERT}(0, \sqrt{n})$ which implies $\text{NP} \subseteq \text{BPP}$.

4.2 Certificate complexity, minimal certificates, and monotonicity

In this section, we prove Fact 4 and Fact 5. We start with a formal definition of *minimal certificates*.

► **Definition 19** (Minimal certificates). *A certificate $S \subseteq [n]$ for $f : \{0,1\}^n \rightarrow \{0,1\}$ on $x \in \{0,1\}^n$ is a minimal certificate if no $S' \subsetneq S$ is a certificate for f on x .*

Proof of Fact 4. Suppose $S \subseteq [n]$ is a minimal 1-certificate for f 's value on a string $x^* \in \{0,1\}^n$ (if S is a 0-certificate the argument is symmetric). First, we observe that S can only contain 1-coordinates of x^* . Indeed, if S contained a 0-coordinate, the certificate S' obtained from S by removing that 0-coordinate would still certify x^* by monotonicity and would therefore contradict the minimality of S . Consider the string $x' \in \{0,1\}^n$ formed by setting all the coordinates in S to 1 and all the coordinates outside S to 0. Then, $|S| = \text{Cert}(f, x') \leq \text{Cert}(f)$. Specifically, any minimal certificate for f 's value on x' must be a subset of its 1-coordinates and hence a subset of S . The minimality of S implies no $S' \subsetneq S$ can certify f 's value on x' . ◀

Proof of Fact 5. This fact is witnessed by the *addressing function*, $\text{ADDRESS}_r : [r] \times \{0,1\}^r \rightarrow \{0,1\}$ defined as $\text{ADDRESS}_r(x, y) = y_x$. Viewing ADDRESS_r as a Boolean function on $\log r + r$ bits, we have $\text{Cert}(\text{ADDRESS}_r) = \log r + 1$ but any fixing of the r bits of y constitutes a minimal certificate. ◀

5 A structural result for functions with low certificate complexity

Notation and useful definitions. All distributions over $\{0,1\}^n$ are uniform unless otherwise specified. We use **boldface** to denote random variables (e.g. $\mathbf{x} \sim \{0,1\}^n$) and we write “w.h.p.” to mean with probability $\geq 1 - 1/n^{\omega(1)}$.

► **Definition 20** (Variance). *For any function $f : \{0,1\}^n \rightarrow \{0,1\}$, the variance of f is defined as*

$$\text{Var}(f) := \text{Var}_{\mathbf{x} \sim \{0,1\}^n} [f(\mathbf{x})] = \Pr_{\mathbf{x} \sim \{0,1\}^n} [f(\mathbf{x}) = 0] \cdot \Pr_{\mathbf{x} \sim \{0,1\}^n} [f(\mathbf{x}) = 1].$$

► **Definition 21** (Influence). *For any functions $f : \{0,1\}^n \rightarrow \{0,1\}$, we define the influence of the coordinate $i \in [n]$ to be*

$$\text{Inf}_i(f) := \Pr_{\mathbf{x} \sim \{0,1\}^n} [f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})]$$

where $\mathbf{x}^{\oplus i} \in \{0,1\}^n$ is the unique point that differs from $\mathbf{x}$ in only the i^{th} coordinate. We also define the total influence as

$$\text{Inf}(f) := \sum_{i \in [n]} \text{Inf}_i(f),$$

and the maximum influence as

$$\text{MaxInf}(f) := \max_{i \in [n]} \{\text{Inf}_i(f)\}.$$

Main structural result

In this section we prove the following structural result: iteratively and randomly restricting a function f by its most influential variable causes it to become constant w.h.p. after $O(\text{Cert}(f)^4)$ iterations. To do so, we analyze the following family of decision trees.

► **Definition 22.** For a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, the influence-maximizing tree of depth d for f , denoted $T_f(d)$, is the complete tree of depth d such that for each internal node v , the variable $i(v)$ queried is the one with the largest influence in the subfunction f_v :

$$\text{Inf}_{i(v)}(f_v) \geq \text{Inf}_j(f_v) \quad \text{for all } j \in [n].$$

Ties are broken arbitrarily, and for convenience, variables are still queried even if the function is already constant before depth d .

We interpret the leaves and internal nodes of $T_f(d)$ as functions obtained from f by restricting each variable corresponding to an edge in that node's path. Note that a random leaf $\ell \sim T_f(d)$ corresponds a random restriction where we iteratively and randomly restrict the most influential variable of f for d iterations.

► **Lemma 23.** Let f be a function with certificate complexity $\text{Cert}(f) \leq k$. Then, for a leaf ℓ drawn uniformly at random from $T_f(d)$ for $d = O(k^4 + k^3 \log(1/\varepsilon))$, we have

$$\Pr_{\ell} [f_{\ell} \text{ is constant}] \geq 1 - \varepsilon.$$

5.1 Useful facts for the proof of Lemma 23

The proof of Lemma 23 is by a potential function argument, where the potential function is the average total influence of leaf functions f_{ℓ} in $T_f(d)$:

$$\phi(d) := \mathbb{E}_{\ell \sim T_f(d)} [\text{Inf}(f_{\ell})].$$

In the remainder of this subsection, we will prove an upper bound on the initial value $\phi(0)$ and a lower bound on the drop in ϕ as we increment d . To do so, we use some basic facts about certificates and query complexity, and a well-known inequality of [16].

► **Definition 24** (Deterministic query complexity). The depth, or deterministic query complexity of f is the depth of the minimum-depth decision tree computing f :

$$\text{Depth}(f) := \min_{T \text{ computing } f} \max_{\ell \in T} [\text{depth}(\ell)].$$

Certificate complexity and query complexity are well known to be within a polynomial factor of one another. See e.g. [6].

► **Fact 25.** For any $f : \{0, 1\}^n \rightarrow \{0, 1\}$, $\text{Cert}(f) \leq \text{Depth}(f) \leq \text{Cert}(f)^2$.

We'll need to lower and upper bound $\text{Inf}(f)$ as a function of $\text{Var}(f)$.

► **Proposition 26.** For any $f : \{0, 1\}^n \rightarrow \{0, 1\}$, $4 \text{Var}(f) \leq \text{Inf}(f) \leq 4 \text{Var}(f) \cdot \text{Cert}(f)$.

Proof. The first inequality is the well-known Poincaré inequality for the Boolean cube [15]. For the second, let $\text{Sens}(f, x)$ be the number sensitive coordinates of x , i.e. the number distinct $i \in [n]$ for which $f(x) \neq f(x^{\oplus i})$. We rewrite the definition for the total influence of f ,

$$\begin{aligned}
\text{Inf}(f) &= \mathbb{E}_{\mathbf{x} \sim \{0,1\}^n} [\text{Sens}(f, \mathbf{x})] \\
&= \mathbb{E}_{\mathbf{x} \sim \{0,1\}^n} [\text{Sens}(f, \mathbf{x}) \cdot \mathbb{1}[f(\mathbf{x}) = 0] + \text{Sens}(f, \mathbf{x}) \cdot \mathbb{1}[f(\mathbf{x}) = 1]].
\end{aligned}$$

Every sensitive edge (i.e. $(x, x^{\oplus i})$ where $f(x) \neq f(x^{\oplus i})$) must contain one endpoint classified as 0 and one endpoint classified as 1. Therefore, the two terms in the above sum are equal. Furthermore, any certificate for x must include all $\text{Sens}(f, x)$ sensitive coordinates, so $\text{Sens}(f, x) \leq \text{Cert}(f, x) \leq \text{Cert}(f)$. We can therefore bound,

$$\begin{aligned}
\text{Inf}(f) &= 2 \cdot \min(\mathbb{E}[\text{Sens}(f, \mathbf{x}) \cdot \mathbb{1}[f(\mathbf{x}) = 0]], \mathbb{E}[\text{Sens}(f, \mathbf{x}) \cdot \mathbb{1}[f(\mathbf{x}) = 1]]) \\
&\leq 2 \cdot \text{Cert}(f) \cdot \min(\Pr[f(\mathbf{x}) = 0], \Pr[f(\mathbf{x}) = 1]) \\
&\leq 4 \cdot \text{Cert}(f) \cdot \Pr[f(\mathbf{x}) = 0] \cdot \Pr[f(\mathbf{x}) = 1] = 4 \cdot \text{Cert}(f) \cdot \text{Var}(f). \quad \blacktriangleleft
\end{aligned}$$

Using the bound $\text{Var}(f) \leq \frac{1}{4}$ for any f with range $\{0, 1\}$, we obtain the following corollary.

► **Corollary 27.** *For any $f : \{0, 1\}^n \rightarrow \{0, 1\}$, $\text{Inf}(f) \leq \text{Cert}(f)$.*

The corollary provides the upper bound on $\phi(0)$, since $\phi(0)$ is just $\text{Inf}(f)$. To lower bound the drop in ϕ as we increment d , we need the following two claims. Together, they lower bound the drop in influence of subfunctions if you query the most influential variable in a function with low query complexity.

► **Theorem 28** (Corollary of Theorem 1.1 from [16]). *For any $f : \{0, 1\}^n \rightarrow \{0, 1\}$,*

$$\text{MaxInf}(f) \geq \frac{4 \text{Var}(f)}{\text{Depth}(f)}.$$

► **Proposition 29.** *For any function f and variable x_i ,*

$$\text{Inf}(f) - \text{Inf}_i(f) = \frac{1}{2}(\text{Inf}(f_{x_i=0}) + \text{Inf}(f_{x_i=1})).$$

Finally, in addition to these statements about the behavior of ϕ , we also need a condition under which we can be assured that a leaf is constant.

► **Fact 30** (Granularity of variance). *Let f be a function with certificate complexity k . Then if $\text{Var}(f) < 2^{-k} - 2^{-2k}$, it must be the case that $\text{Var}(f) = 0$, and thus f is constant.*

Proof. $\text{Var}(f)$ is defined as $\Pr[f(\mathbf{x}) = 0] \cdot \Pr[f(\mathbf{x}) = 1]$. If $f(x) = 0$ for some x , then there must be a 0-certificate for x ; i.e. there must be a subcube of codimension $\leq k$ for which f is constant 0. Thus, $\Pr[f(\mathbf{x}) = 0] \geq 2^{-k}$, and more generally, if f is not constant then $\min(\Pr[f(\mathbf{x}) = 0], \Pr[f(\mathbf{x}) = 1]) \geq 2^{-k}$. The function $p(1-p)$, for p ranging from 2^{-k} to $1 - 2^{-k}$, is minimized at the endpoints of that interval, where it takes the value $2^{-k} - 2^{-2k}$. ◀

5.2 Proof of Lemma 23

With the claims presented in the previous subsection in hand, we can now prove Lemma 23. A lower bound on the drop in ϕ in terms of variable influences in $T_f(d-1)$ follows directly from Proposition 29:

► **Corollary 31** (Corollary of Proposition 29). *For any function f and depth $d \leq n$,*

$$\phi(d) = \phi(d-1) - \mathbb{E}_{\ell \in T_f(d-1)}[\text{Inf}_{i(\ell)}(f_\ell)],$$

where $i(\ell)$ is the variable queried at ℓ in $T_f(d-1)$.

► **Lemma 32** (Upper bound on $\phi(d)$). *For a function f with certificate complexity k , and any depth $d \leq n$, we have*

$$\phi(d) \leq k \cdot \left(1 - \frac{1}{k^3}\right)^d$$

Proof. Each depth- d node of $T_f(d)$ maximizes influence in the corresponding leaf function f_ℓ of $T_f(d-1)$. By Theorem 28, this means its influence on f_ℓ is at least $4 \text{Var}(f_\ell)/k^2$. Because f_v has certificate complexity at most k , Proposition 26 guarantees that $4 \text{Var}(f_v)/k^2 \geq \text{Inf}(f_v)/k^3$. We now apply Corollary 31:

$$\begin{aligned} \phi(d) &= \phi(d-1) - \mathbb{E}_{\ell \in T_f(d-1)}[\text{Inf}_{i(\ell)}(f_\ell)] && \text{(Corollary 31)} \\ &\leq \phi(d-1) - \mathbb{E}_{\ell \in T_f(d-1)}[\text{Inf}(f_\ell)/k^3] && \text{(Theorem 28)} \\ &\leq \phi(d-1) - \frac{\phi(d-1)}{k^3} && \text{(definition of } \phi) \\ &\leq \phi(0) \cdot \left(1 - \frac{1}{k^3}\right)^d. && \text{(solution to recurrence relation)} \end{aligned}$$

The lemma follows from the fact that $\phi(0) \leq k$, which is a consequence of Corollary 27. ◀

Proof of Lemma 23. Let $d = k^3(2k + \ln k + \log(1/\varepsilon))$, which is $O(k^4 + k^3 \log(1/\varepsilon))$. We begin with the fact that

$$\begin{aligned} \phi(d) &\leq k \cdot \left(\left(1 - \frac{1}{k^3}\right)^{k^3} \right)^{2k + \ln k + \log(1/\varepsilon)} \\ &\leq k \cdot e^{-(2k + \ln k + \log(1/\varepsilon))} \\ &< \varepsilon \cdot 2^{-2k}. \end{aligned}$$

By Markov's inequality, at least a $1 - \varepsilon$ fraction of the leaf functions must have influence at most 2^{-2k} . By Proposition 26, this implies that their variance is also at most 2^{-2k} . Since each leaf function has certificate complexity at most k , from Fact 30 we may infer that these leaves are constant, which concludes the proof. ◀

5.3 A robust version of Lemma 23

Algorithmically, we need not build the influence-maximizing tree in order for the potential function argument to go through. Suppose we estimate influences to within a factor of two instead. Then the influence of the queried variable is at least $2 \text{Var}(f_v)/k^2$, and the multiplicative drop in $\phi(d)$ becomes $(1 - 1/2k^3)$. Thus, to achieve the $1 - \varepsilon$ probability guarantee with 2-approximate influences, it suffices to grow the tree to twice the depth. We state the “robust” version of this statement as a corollary.

► **Corollary 33** (Robust version of Lemma 23). *Let f be a function with certificate complexity at most k , and T_f be any decision tree of depth $O(k^4 + k^3 \log(1/\varepsilon))$ such that for each internal node v , the variable $i(v)$ queried has influence within a factor of 2 of the largest influence in the subfunction f_v :*

$$\text{Inf}_{i(v)}(f_v) \geq \frac{1}{2} \text{MaxInf}(f_v).$$

Then, for a leaf ℓ drawn uniformly at random from T_f , we have

$$\Pr_{\ell}[f_{\ell} \text{ is constant}] \geq 1 - \varepsilon.$$

6 Balanced influences and an algorithmic version of Corollary 33

In this section, we will prove Lemma 6, restated below.

► **Lemma 34** (Formal version of Lemma 6). *For any $n \in \mathbb{N}$ and $\delta \in (0, 1)$, given any $\text{poly}(n)$ -sized circuit $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and an NP oracle, there is a randomized algorithm running in $\text{poly}(n, \log(1/\delta))$ -time that, with probability at least $1 - \delta$, finds a restriction π to $O(\text{Cert}(f)^4)$ variables such that f_{π} is a constant function.*

FIND-RESTRICTION(f):

Given: A circuit representation of $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and an NP oracle.

Output: A restriction π such that f_{π} is constant.

1. Initialize $\pi \leftarrow \emptyset$.
2. While f_{π} is not constant (which is checked using the NP oracle), {
 - a. Using Lemma 8, estimate $\text{BALINF}_i(f_{\pi})$ for each $i \in [n]$ to within $\pm \frac{1}{4n}$ with success probabilities $\geq 1 - \frac{1}{4n^2}$ and let i^* be the index with the largest estimate.
 - b. Update π with a random restriction to x_{i^*} : $\pi \leftarrow \pi \cup \{x_{i^*} \leftarrow \mathbf{b}\}$ where $\mathbf{b} \sim \{0, 1\}$ is uniform random.
3. Return π .

■ **Figure 1** Algorithm for finding a restriction that fixes f to a constant.

Lemma 34 is an easy consequence of Lemma 35: The algorithm in Lemma 34 simply repeats **FIND-RESTRICTION(f)** $\log(1/\delta)$ times and returns the shortest certificate it outputted.

► **Lemma 35.** *The algorithm **FIND-RESTRICTION(f)** from Figure 1, with probability at least $\frac{1}{2}$, returns a restriction π of length at most $O(\text{Cert}(f)^4)$.*

Before proving Lemma 35 we discuss why it uses balanced influences rather than influences. As we are applying Corollary 33, we wish to determine a variable with influence at least half as large as $\text{MaxInf}(f)$. The definition of influence immediately suggests an algorithm for estimating influences: Randomly sample $\mathbf{x} \sim \{0, 1\}^n$ and then compute whether $f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})$. Using $\text{poly}(1/\varepsilon)$ iterations of that procedure, we can estimate influences to *additive* accuracy $\pm \varepsilon$. To guarantee we pick a variable with influence at least half of $\text{MaxInf}(f)$, we would want to estimate influences to accuracy in $\pm \frac{\text{MaxInf}(f)}{4}$. Unfortunately, the naive influence estimator requires $1/\text{poly}(\text{MaxInf}(f))$ queries to do so, which is intractable when MaxInf is too small.

A first hope is that the bound of $\text{MaxInf}(f) \geq 4 \text{Var}(f)/\text{Depth}(f)$ from Theorem 28 will ensure that MaxInf is not too small. Unfortunately, in the proof of Lemma 23, we can have Var as small as $2^{-\text{Cert}(f)}$. When $\text{Var}(f)$ is small, almost all inputs are labeled the same way by f , so it is unlikely that a random edge $(\mathbf{x}, \mathbf{x}^{\oplus i})$ will be labeled differently. *Balanced influences* (Definition 7) correct for this effect by ensuring that the initial point, $\mathbf{x}$, is equally likely to be labeled 0 or 1 by f .

If f is already balanced, meaning it is equally likely to output 0 or 1, then $\mathcal{D}_{\text{bal}}^{(f)}$ is just the uniform distribution and $\text{BALINF}_i(f) = \text{Inf}_i(f)$. Otherwise, as we show in the following Lemma, balanced influences are proportional to influences scaled by variance.

► **Lemma 36** (Balanced influences proportional to influences). *For any non-constant function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and coordinate $i \in [n]$,*

$$\text{BALINF}_i(f) = \frac{\text{Inf}_i(f)}{4 \text{Var}(f)}.$$

Proof. Let $S_i := \{x \in \{0, 1\}^n \mid f(x) \neq f(x^{\oplus i})\}$ be the set of all inputs to f that are sensitive at the i^{th} coordinate. Then,

$$\text{Inf}_i(f) = \Pr_{\mathbf{x} \sim \{0, 1\}^n} [\mathbf{x} \in S_i] = \frac{|S_i|}{2^n}.$$

Similarly,

$$\text{BALINF}_i(f) = \frac{1}{2} \left(\Pr_{\mathbf{x} \sim f^{-1}(0)} [\mathbf{x} \in S_i] + \Pr_{\mathbf{x} \sim f^{-1}(1)} [\mathbf{x} \in S_i] \right).$$

The key observation is that for every $x \in S_i$ it is also true that $x^{\oplus i} \in S_i$, so f classifies exactly half the inputs in S_i as 0, and the other half as 1. Therefore,

$$\text{BALINF}_i(f) = \frac{1}{2} \left(\frac{\frac{|S_i|}{2}}{|f^{-1}(0)|} + \frac{\frac{|S_i|}{2}}{|f^{-1}(1)|} \right).$$

For each $b \in \{0, 1\}$, we use p_b as shorthand for $\Pr_{\mathbf{x} \sim \{0, 1\}^n} [f(\mathbf{x}) = b]$. Then,

$$\begin{aligned} \text{BALINF}_i(f) &= \frac{1}{4} \left(\frac{|S_i|}{2^n p_0} + \frac{|S_i|}{2^n p_1} \right) \\ &= \frac{|S_i|}{2^n} \cdot \frac{p_1 + p_0}{4 p_0 p_1} \\ &= \text{Inf}_i(f) \cdot \frac{1}{4 p_0 p_1} & (\text{Inf}_i(f) = \frac{|S_i|}{2^n}, p_1 + p_0 = 1) \\ &= \frac{\text{Inf}_i(f)}{4 \text{Var}(f)}. & (\text{Var}(f) = p_0 p_1) \end{aligned}$$

◀

The following is a direct corollary of Lemma 36, Theorem 28, and the fact that $\text{Depth}(f) \leq n$ for any $f : \{0, 1\}^n \rightarrow \{0, 1\}$.

► **Corollary 37.** *For any non-constant function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, there is a coordinate $i \in [n]$ satisfying*

$$\text{BALINF}_i(f) \geq \frac{1}{\text{Depth}(f)} \geq \frac{1}{n}.$$

We are now able to prove Lemma 35 contingent on Lemma 8 stating that balanced influences can be efficiently estimated, which will appear in the next subsection.

Proof of Lemma 35. FIND-RESTRICTION makes at most n^2 estimates of balanced influence, so with probability at least $\frac{3}{4}$, all those estimates are within $\pm \frac{1}{4n}$ of the true balanced influences. By Corollary 37, there is always a variable with balanced influence at least

$\frac{1}{n}$, so as long as all estimates succeed, FIND-RESTRICTION will always choose an i^* such that $\text{BALINF}_{i^*}(f) \geq \max_i(\text{BALINF}_i(f))$. By Lemma 36, that also implies that $\inf_{i^*}(f) \geq \text{MaxInf}(f)/2$.

As a result, FIND-RESTRICTION builds a uniformly random path of the tree T_f described in Corollary 33. All but $\frac{1}{4}$ -fraction of paths in T_f at depth $O(\text{Cert}(f)^4)$ reach a restriction π for which f_π is constant. Hence, by a union bound, the probability FIND-RESTRICTION does not terminate with $|\pi| \leq O(\text{Cert}(f)^4)$ is at most $\frac{1}{4} + \frac{1}{4} \leq \frac{1}{2}$. ◀

6.1 Estimating balanced influences

Definition 7 suggests a procedure for estimating balanced influences: Randomly sample $\mathbf{x} \sim \mathcal{D}_{\text{bal}}^{(f)}$ and compute whether $f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})$. Using $\text{poly}(1/\varepsilon)$ iterations of that procedure is sufficient to estimate balanced influences to additive accuracy $\pm\varepsilon$.

The challenge in sampling from $\mathcal{D}_{\text{bal}}^{(f)}$ is that when $\text{Var}(f)$ is small, the $\mathcal{D}_{\text{bal}}^{(f)}$ is far from the uniform distribution. Here, we utilize an NP oracle for a *uniform generation* algorithm.³

► **Theorem 38** (Uniform generation with an NP oracle, [3]). *There is an efficient randomized algorithm $\mathcal{A}$ which, given a satisfiable poly-sized circuit $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and NP oracle, with high probability, outputs a uniform $\mathbf{x} \sim f^{-1}(1)$. In the failure case, $\mathcal{A}$ outputs $\perp$.*

As an easy corollary, we can sample from $\mathcal{D}_{\text{bal}}^{(f)}$.

► **Corollary 39** (Sampling from $\mathcal{D}_{\text{bal}}^{(f)}$). *There is an efficient randomized algorithm $\mathcal{A}$ which, given a poly-sized non-constant circuit $f : \{0, 1\}^n \rightarrow \{0, 1\}$, with high probability outputs a uniform $\mathbf{x} \sim \mathcal{D}_{\text{bal}}^{(f)}$. In the failure case, $\mathcal{A}$ outputs $\perp$.*

Proof. $\mathcal{A}$ first samples a uniform $\mathbf{b} \sim \{0, 1\}$. If $\mathbf{b}$ is 1, it uses the uniform generation algorithm from Theorem 38 on f . Otherwise, it uses that uniform generation algorithm on $\neg f$, which is still a poly-sized circuit.

By union bound, the failure probability (of outputting $\perp$) of $\mathcal{A}$ is still small. When it does not fail, $\mathcal{A}$ outputs a uniform sample from $\mathcal{D}_{\text{bal}}^{(f)}$. ◀

Finally, we note that Lemma 8 is a direct consequence of Corollary 39 and Definition 7: The algorithm with failure probability $1 - \delta$ takes $\text{poly}(1/\varepsilon, \log(1/\delta))$ samples $\mathbf{x} \sim \mathcal{D}_{\text{bal}}^{(f)}$ and counts the fraction of those samples for which $f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})$.

6.2 Technical remarks

Comparison with [4]

[4] solved the certification problem for *monotone* f , and also computed influences over a certain balanced distribution as a key step. For any non-constant monotone $f : \{0, 1\}^n \rightarrow \{0, 1\}$, there is guaranteed to be some $p^* \in (0, 1)$, called the *critical probability*, satisfying,

$$\mathbb{E}_{\mathbf{x} \sim \mathcal{D}_{p^*}} [f(\mathbf{x})] = \frac{1}{2}$$

where $\mathbf{x} \sim \mathcal{D}_{p^*}$ means each x_i is independently 1 with probability p^* and 0 otherwise. [4] are able to show that there is a coordinate with high influence at the critical probability, meaning $\Pr_{\mathbf{x} \sim \mathcal{D}_{p^*}} [f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})]$ is large.

³ [12] gave an algorithm for *approximate* uniform generation, which would have also sufficed for our purpose. For simplicity, we cite the more recent work that gives *exact* uniform generation.

While the ways in which [4] and this work use the existence of a coordinate with high influence is quite different, we find it intriguing that both rely on finding such a coordinate on a distribution on which f is balanced. A natural avenue for future work is to establish a formal connection between the two works. Is there some definition of “balancing a distribution for f ,” that encompasses both this work and [4]’s techniques, and is sufficient for certification?

An alternative approach through approximate counting

Given a poly-sized circuit $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and coordinate $i \in [n]$, we can construct the poly-sized circuit $g_i(x) := \mathbb{1}[f(x) \neq f(x^{\oplus i})]$, which measures whether f is sensitive in the i^{th} direction at the input. In the proof of Lemma 36, we saw that $\text{Inf}_i(f) = \frac{|g_i^{-1}(1)|}{2^n}$. This gives an alternative approach to estimating the coordinate with largest influence: Approximately count the number of accepting inputs of g_i for each $i \in [n]$ and output the i maximizing that count. This approximate count just needs to be accurate to a constant *multiplicative* accuracy (i.e., $\text{count}_i \in [\frac{3}{4} \cdot |g_i^{-1}(1)|, \frac{5}{4} \cdot |g_i^{-1}(1)|]$).

Classical results of [21, 22] show that approximate counting can be done efficiently and deterministically given a Σ_2 oracle, and [12] observed that implicit in those results, an NP oracle is sufficient if we allow for randomized algorithms. Furthermore, they proved an equivalence between (almost) uniform generation and approximate counting, so it’s not surprising that our algorithm has two alternative approaches, one through uniform generation and one through approximate counting.

Given two equivalent approaches, we chose to highlight that through uniform generation as we find it more intuitive. It also illuminates the possible connection, detailed earlier, with [4]’s algorithm for certifying monotone functions.

7 Our certification algorithm: Proof of Theorem 2

In this section, we show how any procedure that takes a function as input and outputs a certificate for an *arbitrary* input can be converted into a procedure that takes a function and a string as inputs and outputs a certificate for the function’s value on that string. Using this construction and the certification procedure from Section 5, we obtain our main certification algorithm.

FIND-CERTIFICATE(f, x^*):

Given: A circuit representation of $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and an input $x^* \in \{0, 1\}^n$.

Output: A certificate $S \subseteq [n]$ for f ’s value on x^*

1. Initialize $S \leftarrow \emptyset$
2. While S is not a certificate for f ’s value on x^* :
 - a. $S \leftarrow S \cup \text{RESTRICTION}(f_{S \leftarrow x^*})$
3. Return S

■ **Figure 2** Algorithm for the certification problem using **RESTRICTION** as a subroutine.

The procedure **RESTRICTION**(f) in Figure 2 is a (possibly randomized) procedure that takes as input a circuit representation of $f : \{0, 1\}^n \rightarrow \{0, 1\}$ outputs a set of coordinates $S \subseteq [n]$ such that there is some $u \in \{0, 1\}^{|S|}$ such that $f_{S \leftarrow u}$ is a constant function (equivalently S is a certificate for f ’s value on some input).

The algorithm works by iteratively building a certificate S using $\text{RESTRICTION}(\cdot)$ as a subroutine. While S is not yet a certificate for f 's value on x^* , the algorithm calls $\text{RESTRICTION}(\cdot)$ on the subfunction obtained by restricting f according to S and x^* . It then augments the candidate certificate with the output from $\text{RESTRICTION}(\cdot)$.

► **Lemma 40** (Solving the certification problem using $\text{RESTRICTION}(\cdot)$). *The algorithm in Figure 2 runs in polynomial-time with access to an NP oracle and outputs a certificate of size at most $2 \cdot \text{Cert}(f) \cdot \gamma(\text{Cert}(f))$ for f 's value on x^* where $\gamma : \mathbb{N} \rightarrow \mathbb{N}$ is any nondecreasing function satisfying $|\text{RESTRICTION}(f)| \leq \gamma(\text{Cert}(f))$.*

We write $\text{Cert}_0(f) = \max_{x \in f^{-1}(0)} \{\text{Cert}(f, x)\}$ to denote the 0-certificate complexity of f and $\text{Cert}_1(f) = \max_{x \in f^{-1}(1)} \{\text{Cert}(f, x)\}$ for the 1-certificate complexity of f . Our proof relies on the following fact which states that the 0-certificate complexity of a function decreases when restricting the coordinates of a 1-certificate to any set of values and vice versa for a 1-certificate. For the proof of this result, see [4, Theorem 6]. The main idea underlying the proof is that every 0-certificate has a nonempty intersection with every 1-certificate.

► **Proposition 41** (See [4, Theorem 6]). *Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$, $S \subseteq [n]$ and $u \in \{0, 1\}^{|S|}$. If $f_{S \leftarrow u}(x)$ is the constant function for all $x \in \{0, 1\}^{n-|S|}$, then*

$$\text{Cert}_1(f) + \text{Cert}_0(f) - 1 \geq \text{Cert}_0(f_{S \leftarrow u}) + \text{Cert}_1(f_{S \leftarrow u})$$

for all $u' \in \{0, 1\}^{|S|}$.

Proof of Lemma 40. We claim that the algorithm depicted in Figure 2 satisfies the lemma statement.

Each step of the algorithm can be implemented efficiently using an NP oracle. In particular, determining if $S \subseteq [n]$ is a certificate for f 's value on x^* is equivalent to checking if $f_{S \leftarrow x^*}$ is the constant function. This task can be accomplished by restricting the circuit for f to obtain a circuit for $f_{S \leftarrow x^*}$ and checking whether the restricted circuit is satisfiable (or, possibly unsatisfiable depending on whether $f(x^*) = 1$). Moreover, the correctness of the algorithm follows immediately from the condition of the while loop. Therefore, it suffices to bound the runtime.

If $S = \text{RESTRICTION}(f)$, then

$$\begin{aligned} \text{Cert}_1(f) + \text{Cert}_0(f) - 1 &\geq \text{Cert}_1(f_{S \leftarrow u}) + \text{Cert}_0(f_{S \leftarrow u}) \\ &> 0 \end{aligned} \quad (\text{Proposition 41})$$

for all u such that $f_{S \leftarrow u}$ is nonconstant. Therefore, each step of the algorithm decreases the sum $\text{Cert}_1(f_{S \leftarrow x^*}) + \text{Cert}_0(f_{S \leftarrow x^*})$ by at least 1 and terminates when this quantity reaches 0. It follows by induction that the main loop terminates after at most $2 \cdot \text{Cert}(f) \geq \text{Cert}_0(f) + \text{Cert}_1(f)$ calls to $\text{RESTRICTION}(\cdot)$. Each loop iteration adds

$$\begin{aligned} |\text{RESTRICTION}(f_{S \leftarrow x^*})| &\leq \gamma(\text{Cert}(f_{S \leftarrow x^*})) \\ &\leq \gamma(\text{Cert}(f)) \end{aligned} \quad (\text{Cert}(f_{S \leftarrow x^*}) \leq \text{Cert}(f))$$

coordinates to the candidate certificate. Hence, the overall size of the final certificate is bounded by $2 \cdot \text{Cert}(f) \cdot \gamma(\text{Cert}(f))$. ◀

Proof of Theorem 2 using Lemmas 6 and 40. By Lemma 6, there is a randomized polynomial-time algorithm with an NP oracle that implements $\text{RESTRICTION}(f)$. This algorithm has the guarantee that $|\text{RESTRICTION}(f)| \leq O(\text{Cert}(f)^4)$. Therefore, the algorithm in Figure 2 satisfies Theorem 2 by Lemma 40: it runs in polynomial-time with NP oracle access and w.h.p. (over the randomness of $\text{RESTRICTION}(f)$) outputs a certificate of size $O(\text{Cert}(f)^5)$ for f 's value on a given input. ◀

8 Hardness of instance-wise guarantees: Proof of Theorem 3

Theorem 3 follows as a consequence of the following lemma.

► **Lemma 42.** *There is a polynomial-time algorithm that takes a co-nondeterministic circuit $C : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ accepting a nonempty monotone set and a parameter $m \geq n$ and outputs a circuit representation of a function $f : \{0, 1\}^{n+m} \rightarrow \{0, 1\}$ which satisfies the following.*

1. *If C accepts an input with $\leq k$ ones, then $\text{Cert}(f, (1^n, 1^m)) \leq k$.*
2. *If every x that C accepts has $> k'$ ones, then $|S| > \min\{k', m - n - \log^c(m)\}$ for every certificate $S \subseteq [m + n]$ of f 's value on $(1^n, 1^m)$ where $k' > k$ and $c > 1$ is an absolute constant.*

The proof of this lemma involves zero-error bit-fixing dispersers. At a high level, a disperser is a nearly-surjective function that remains nearly-surjective even after restricting all but a small number of the input bits. More specifically, an ε -error disperser with entropy threshold λ is an efficiently constructible function $D : \{0, 1\}^m \rightarrow \{0, 1\}^n$ such that the image of the function after restricting all but λ input bits is at least an $(1 - \varepsilon)$ -fraction of $\{0, 1\}^n$. For our application, we require $\varepsilon = 0$ so that the function remains fully surjective after restricting the input bits.

► **Definition 43** (Zero-error bit-fixing disperser with entropy threshold λ). *Consider a function $D : \{0, 1\}^m \rightarrow \{0, 1\}^n$. Let $S \subseteq [m]$ be a subset of size $|S| \leq m - \lambda$ and let $u = \{0, 1\}^{|S|}$ be an assignment to that subset. Then $D_{S \leftarrow u} : \{0, 1\}^{m-|S|} \rightarrow \{0, 1\}^n$ is the function D with the variables in S fixed according to u . We say that D is a zero-error bit-fixing disperser with entropy threshold λ if for all such S and u , the image of $D_{S \leftarrow u}$ is $\{0, 1\}^n$.*

We use the following key result due to [10] about explicit zero-error bit-fixing dispersers.

► **Theorem 44** (Explicit constructions of zero-error bit-fixing dispersers [10, Theorem 9]). *There exists a constant $c > 1$ such that for large enough m and $\lambda \geq \log^c m$, there is an explicit zero-error bit-fixing disperser $D : \{0, 1\}^m \rightarrow \{0, 1\}^{\lambda - \log^c m}$ with entropy threshold λ .*

With this theorem in hand, we are able to prove Lemma 42.

Proof of Lemma 42. Let $C : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be co-nondeterministic circuit that accepts a non-empty monotone set and let $m \geq n$ be a parameter. We define the function $f : \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}$ as

$$f(x, z) := C(x, D(z))$$

where $D : \{0, 1\}^m \rightarrow \{0, 1\}^n$ is a zero-error bit-fixing disperser as in Theorem 44. Note that D 's entropy threshold is $\lambda = n + \log^c(m)$. We show that solving the certification problem for f 's value on the input $\vec{1} = (1^n, 1^m)$ satisfies the lemma statement. We first consider the case where C accepts a low Hamming weight input and then consider the case where all accepted strings have large Hamming weight.

C accepts an input with $\leq k$ ones. Suppose C accepts an input $x \in \{0, 1\}^n$ with $\leq k$ ones. Then, by the monotonicity of C , the set $S = \{i : x_i = 1\} \subseteq [n + m]$ consisting of the indices on which x is one is a certificate of size $\leq k$ for f 's value on the input $\vec{1}$.

Every x that C accepts has $> k'$ ones. In this case, we show that every certificate S for f 's value on $\vec{1}$ has at least $\min\{k', m - n - \log^c(m)\}$ coordinates. Let $S \subseteq [n + m]$ be an arbitrary certificate for f 's value on $\vec{1}$. We partition S into two sets $S = S_n \cup S_m$ where $S_n \subseteq [n]$ consists of the indices from the first n -coordinates of $(1^n, 1^m)$ and $S_m \subseteq [m]$ consists of the indices from the last m -coordinates of $(1^n, 1^m)$. We split into two cases depending on the size of S_m . In the first case, we lower bound $|S|$ by k' and in the second case we lower bound $|S|$ by $m - n - \log^c(m)$. Since the two cases are exhaustive, the lemma follows.

1. **S_m is small:** $|S_m| \leq m - (n + \log^c(m))$. The image of $D_{S_m \leftarrow \vec{1}}$ is $\{0, 1\}^n$ because D is a disperser with entropy threshold $\lambda = n + \log^c(m)$. It follows that for every $x \in \{0, 1\}^{n-|S_n|}$ and every $y \in \{0, 1\}^n$,

$$C_{S_n \leftarrow \vec{1}}(x, y) = f(\vec{1}).$$

Therefore, S_n alone is a certificate for f 's value on $\vec{1}$. But this implies $|S| \geq |S_n| > k'$ by our assumption that every x that C accepts has at least k' ones.

2. **S_m is large:** $|S_m| > m - (n + \log^c(m))$. In this case, we have $|S| \geq |S_m| > m - (n + \log^c(m))$ which gives the desired lower bound.

Finally, we observe that our algorithm is efficient since a circuit for f can be constructed by adding wires from the output gates of D to the appropriate input gates of C . ◀

8.1 Putting it all together: Proof of Theorem 3

Proof. Suppose the theorem were false. That is, there is an efficient algorithm that can distinguish between inputs having certificate size k versus requiring size at least $k \cdot n^{1-\varepsilon}$ for some fixed constant $\varepsilon > 0$. Then, we will obtain a contradiction to Theorem 11 by solving MMWW with a gap of $n^{1-\varepsilon}$.

Let (C, k) be an instance of MMWW. Let $f : \{0, 1\}^{n+m} \rightarrow \{0, 1\}$ and $x^* = \vec{1}$ be the function and input obtained from Lemma 42 for $m = 3n$. Run the certification algorithm on $(f, \vec{1})$ and output YES if the algorithm outputs YES and NO otherwise. To show correctness, we consider the two cases of Theorem 11: either C accepts an input of length k or every accepted input has length at least $k \cdot n^{1-\varepsilon}$.

C accepts an input with k ones. By the guarantee of Lemma 42, there is a certificate of size $\leq k$ for f 's value on $\vec{1}$. Therefore, our algorithm correctly outputs YES.

All strings C accepts have at least $k \cdot n^{1-\varepsilon}$ ones. Using $k' = kn^{1-\varepsilon}$ and $m = 3n$, Lemma 42 guarantees that all certificates $|S|$ for f 's value on $\vec{1}$ have size at least $\min\{kn^{1-\varepsilon}, 2n - \log^c(3n)\}$. We observe

$$2n - \log^c(3n) > n \geq k \cdot n^{1-\varepsilon}$$

where the last step follows by our assumption on C (trivially, any input that C accepts can have at most n ones so $n < kn^{1-\varepsilon}$ would be impossible). Therefore,

$$|S| > \min\{kn^{1-\varepsilon}, 2n - \log^c(3n)\} = kn^{1-\varepsilon}$$

for all certificates S of f 's value on $\vec{1}$ and therefore our algorithm correctly outputs NO. ◀

References

- 1 Dana Angluin. Queries and concept learning. *Machine learning*, 2(4):319–342, April 1988. doi:10.1023/A:1022821128753.
- 2 Pablo Barceló, Mikaël Monet, Jorge Pérez, and Bernardo Subercaseaux. Model interpretability through the lens of computational complexity. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:15487–15498, 2020.
- 3 Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of np-witnesses using an np-oracle. *Information and Computation*, 163(2):510–526, 2000.
- 4 Guy Blanc, Caleb Koch, Jane Lange, and Li-Yang Tan. The query complexity of certification. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, pages 623–636, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519935.3519993.
- 5 Guy Blanc, Jane Lange, and Li-Yang Tan. Provably efficient, succinct, and precise explanations. *Advances in Neural Information Processing Systems*, 34, 2021.
- 6 Harry Buhman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002.
- 7 Shuchi Chawla, Jelani Nelson, Chris Umans, and David Woodruff. Visions in theoretical computer science: A report on the TCS visioning workshop 2020. *arXiv preprint*, 2021. arXiv:2107.02846.
- 8 Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. Association for Computing Machinery. doi:10.1145/800157.805047.
- 9 Remi Delannoy and Kuldeep S. Meel. On almost-uniform generation of SAT solutions: The power of 3-wise independent hashing. In *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 17:1–17:10, 2022. doi:10.1145/3531130.3533338.
- 10 Ariel Gabizon and Ronen Shaltiel. Invertible zero-error dispersers and defective memory with stuck-at errors. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, 2012.
- 11 Meghal Gupta and Naren Sarayu Manoj. An optimal algorithm for certifying monotone functions. *arXiv preprint*, 2022. arXiv:2204.01224.
- 12 Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical computer science*, 43:169–188, 1986.
- 13 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*. Springer Publishing Company, Incorporated, 2012.
- 14 Elchanan Mossel and Christopher Umans. On the complexity of approximating the VC dimension. *Journal of Computer and System Sciences*, 65(4):660–671, 2002.
- 15 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.
- 16 Ryan O'Donnell, Michael Saks, Oded Schramm, and Rocco Servedio. Every decision tree has an influential variable. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 31–39, 2005.
- 17 Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1527–1535, 2018.
- 18 Marcus Schaefer and Chris Umans. Completeness in the polynomial-time hierarchy: Part ii. *SIGACT News*, 33(4):22–36, 2002.
- 19 Marcus Schaefer and Christopher Umans. Completeness in the polynomial-time hierarchy: A compendium. *SIGACT news*, 33(3):32–49, 2002.
- 20 Andy Shih, Arthur Choi, and Adnan Darwiche. A symbolic approach to explaining bayesian network classifiers. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 5103–5111, 2018.
- 21 Michael Sipser. A complexity theoretic approach to randomness. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 330–335, 1983.

- 22 Larry Stockmeyer. The complexity of approximate counting. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 118–126, 1983.
- 23 Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Loss-less condensers, unbalanced expanders, and extractors. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 143–152, 2001.
- 24 Christopher Umans. Hardness of approximating Σ_2^P minimization problems. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 465–474, 1999.
- 25 Christopher Umans. On the complexity and inapproximability of shortest implicant problems. In *Proceedings of the 26th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 687–696, 1999.
- 26 Christopher Umans. The minimum equivalent DNF problem and shortest implicants. *Journal of Computer and System Sciences*, 63(4):597–611, 2001.
- 27 Christopher Umans. Optimization problems in the polynomial-time hierarchy. In *International Conference on Theory and Applications of Models of Computation*, pages 345–355. Springer, 2006.
- 28 Christopher Matthew Umans. *Approximability and completeness in the polynomial hierarchy*. University of California, Berkeley, 2000.
- 29 Leslie Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.
- 30 Moshe Vardi. The SAT revolution: Solving, sampling, and counting. Available at <https://www.cs.rice.edu/~vardi/papers/highlights15.pdf>.
- 31 Daniel S Weld and Gagan Bansal. The challenge of crafting intelligible intelligence. *Communications of the ACM*, 62(6):70–79, 2019.