Algorithmica (2017) 79:1014–1051
DOI 10.1007/s00453-016-0218-8

On Virtual Grey Box Obfuscation for General Circuits

Nir Bitansky[1] · **Ran Canetti**[2] · **Yael Tauman Kalai**[3] · **Omer Paneth**[4]

Received: 1 July 2015 / Accepted: 19 September 2016 / Published online: 27 September 2016

Abstract An obfuscator $\mathcal{O}$ is Virtual Grey Box (VGB) for a class $\mathcal{C}$ of circuits if, for any $C \in \mathcal{C}$ and any predicate π, deducing $\pi(C)$ given $\mathcal{O}(C)$ is tantamount to deducing $\pi(C)$ given unbounded computational resources and polynomially many oracle queries to C. VGB obfuscation is often significantly more meaningful than indistinguishability obfuscation (IO). In fact, for some circuit families of interest VGB is equivalent to full-fledged Virtual Black Box obfuscation. We investigate the feasibility of obtaining VGB obfuscation for general circuits. We first formulate a natural strengthening of IO, called *strong IO* (SIO). Essentially, $\mathcal{O}$ is SIO for class $\mathcal{C}$ if $\mathcal{O}(C_0) \approx \mathcal{O}(C_1)$ whenever the pair (C_0, C_1) is taken from a distribution over $\mathcal{C}$

A preliminary version of this work appears in the proceedings of CRYPTO 2014.
N. Bitansky's Research done while in Tel Aviv University and supported by an IBM Ph.D. Fellowship, the Check Point Institute for Information Security, and The Israeli Ministry of Science and Technology.
R. Canetti Supported by the Check Point Institute for Information Security, an ISF Grant 20006317, an NSF EAGER grant, and an NSF Algorithmic foundations Grant 1218461.
O. Paneth Supported by the Simons award for graduate students in theoretical computer science and an NSF Algorithmic foundations Grant 1218461.

✉ Nir Bitansky
nirbitan@csail.mit.edu

Ran Canetti
canetti@bu.edu

Omer Paneth
omer@bu.edu

1 MIT, Cambridge, MA, USA

2 Boston University and Tel Aviv University, Tel Aviv, Israel

3 Microsoft Research, Cambridge, MA, USA

4 Boston University, Boston, MA, USA

where, for all x, $C_0(x) \neq C_1(x)$ only with negligible probability. We then show that an obfuscator is VGB for a class $\mathcal{C}$ if and only if it is SIO for $\mathcal{C}$. This result is unconditional and holds for any $\mathcal{C}$. We also show that, for some circuit collections, SIO implies virtual black-box obfuscation. Finally, we formulate a slightly stronger variant of the semantic security property of graded encoding schemes [Pass-Seth-Telang Crypto 14], and show that existing obfuscators, such as the obfuscator of Barak et al. [Eurocrypt 14], are SIO for all circuits in NC^1, assuming that the underlying graded encoding scheme satisfies our variant of semantic security. *Put together, we obtain VGB obfuscation for all NC^1 circuits under assumptions that are almost the same as those used by Pass et al. to obtain IO for NC^1 circuits.* We also observe that VGB obfuscation for all polynomial-size circuits implies the existence of semantically-secure graded encoding schemes with limited functionality known as *jigsaw puzzles*.

Keywords Cryptography · Obfuscation · Simulation · Learning

1 Introduction

Program obfuscation, namely the ability to efficiently compile a given program into a functionally equivalent program that is "unintelligible", is an intriguing concept. Indeed, much effort has been devoted to understanding this concept from the definitional aspect, the algorithmic aspect, and the applications aspect. Here we concentrate on the first two aspects.

The first formulation of secure program obfuscation as suggested by Hada [23]. It requires that the output of any efficient adversary given an obfuscated program can be efficiently simulated given only black box access to the program. Hada observed that this strong security requirement is achievable only for *learnable* programs.

Following the work of Hada [23] a number of weaker security definitions have been proposed. We briefly review three notions of interest. The first, *virtual black box* (VBB) obfuscation [7], requires that the obfuscation hides any deterministic predicate of the program. Concretely, focusing on programs represented as circuits, an obfuscator $\mathcal{O}$ for a family of circuits is worst-case VBB if for any polynomial-time adversary $\mathcal{A}$, there exists a polynomial-time simulator $\mathcal{S}$, such that for any circuit C from the family, and any predicate $\pi(\cdot)$, $\mathcal{A}$ cannot learn $\pi(C)$ from $\mathcal{O}(C)$ with noticeably higher probability than $\mathcal{S}$ can, given only oracle access to C. The obfuscator $\mathcal{O}$ is average-case VBB if the above is only required to hold for circuits C that are sampled at random from some distribution on the family.

While VBB obfuscation is natural and expressive notion, Barak et al. [7] showed that this definition, and variants thereof, are unobtainable in general by demonstrating a family of *unobfuscatable functions* where any circuit computing the function inherently leaks secrets that are infeasible to compute given only black-box access. Moreover, it turns out that, under cryptographic assumptions, if the simulator $\mathcal{S}$ is universal (or equivalently, works for any adversarial auxiliary input) then VBB obfuscation is unobtainable for *any* circuit family whose functionality has super-polynomial "pseudo entropy" [5,20].

A weaker variant of VBB, called *indistinguishability obfuscation* (IO) [7], allows the simulator $\mathcal{S}$ to be computationally unbounded. Equivalently, $\mathcal{O}$ is an IO for a circuit collection if for any two circuits C_0 and C_1 in the collection, having the same size and functionality, the obfuscated circuits $\mathcal{O}(C_0)$ and $\mathcal{O}(C_1)$ are indistinguishable. While IO has some attractive properties, and strong cryptographic applications [19,22,26], the security guarantees provided by IO are significantly weaker than those provided by VBB obfuscation. Consider for example the task of obfuscating a *point circuit* that outputs 1 on a single secret point and 0 everywhere else. VBB obfuscation guarantees that guessing the secret point given the obfuscated circuit is as hard as given no information at all. Such a guarantee, however, is not known to follow from IO.

A third notion that lies between VBB and IO is *virtual grey-box* (VGB) obfuscation [4] where the simulator $\mathcal{S}$ is allowed to be *semi-bounded,* namely it can be computationally unbounded, while still making only a polynomial number of queries to the circuit C. While weaker than VBB in general, VGB is still meaningful for circuits that are unlearnable even by semi-bounded learners. This includes, for example, point circuits (or, more generally, *evasive circuit* studied in [3]) where IO may not provide meaningful security. Furthermore, VGB obfuscators for circuits escape the general impossibility results that apply to VBB obfuscators.

On the algorithmic level, for many years we had candidate obfuscators only for very simple functions such as point functions and variants. The landscape has changed completely with the recent breakthrough work of [19], which proposed a candidate general-purpose obfuscation algorithm for all circuits. [19] show that their scheme resists some simple attacks; but beyond that, they do not provide any analytic evidence for security.

Considerable efforts have been made to analyze the security of the [19] obfuscator and variants. The difficulty appears to be in capturing the security properties required from the *graded encodings schemes* [18], which is a central component in the construction. Next, we discuss one line of work that is the starting point for this work.

As a first step towards understanding the security of the [19] obfuscator, [8,9] consider an ideal algebraic model, where the adversary is given "generic graded encodings" that can only be manipulated via admissible algebraic operations. They show that, in this model, variants of the [19] scheme are VBB obfuscators for all polynomial-size circuits. In fact, for circuits in NC^1, they show security even against *semi-bounded* adversaries. In this context, semi-bounded means that the adversary is computationally unbounded, but makes only a polynomial number of generic graded encoding operations. [8,9] demonstrate an efficient simulator that only invokes the semi-bounded adversary as a black box.

Pass et al. [25] made the first step towards proving the security of a general obfuscation scheme based on some natural hardness assumption in the plain model. Specifically, they define a *semantic-security* property for graded encoding schemes, which is aimed at capturing what it means for a graded encoding scheme to "behave essentially as an ideal multilinear graded encoding oracle". They then show that a specially-crafted variant of the [8] obfuscator, with the ideal graded encoding scheme replaced by a semantically-secure graded encoding scheme, is IO for all circuits.

While much of the recent progress in obfuscation constructions has been confined to the notion of IO, in this work we ask whether we can construct obfuscators that go beyond IO, and under which assumptions.

Our contributions We first put forth a new strengthened variant of indistinguishability obfuscation, called *strong IO* (SIO). Informally, an obfuscator $\mathcal{O}$ is SIO for a class of circuits $\mathcal{C}$ if $\mathcal{O}(C_0) \approx \mathcal{O}(C_1)$ not only when $C_0, C_1 \in \mathcal{C}$ have the same functionality, but also when C_0 and C_1 come from distributions over circuits in $\mathcal{C}$ that are "close together", in the sense that for any given input x, the probability that $C_0(x) \neq C_1(x)$ is negligible.[1]

We then show that:

1. SIO for a given class of circuits is in fact *equivalent* to worst-case VGB obfuscation for the same class. Furthermore, for certain classes of functions, such as point functions, hyperplanes, or fuzzy point functions, we show that SIO is equivalent to full-fledged worst-case VBB obfuscation. These equivalences hold unconditionally.
2. Assuming the existence of graded encoding schemes that satisfy a somewhat stronger variant of the semantic-security notion of Pass et al. [25], we show that known obfuscation schemes are SIO for all circuits in NC^1. More generally, we show that *any* obfuscator for a class of circuits $\mathcal{C}$ that is VBB against semi-bounded adversaries in the ideal graded encoding model is SIO in the plain model, when the ideal graded encoding oracle is replaced by a graded encoding scheme that satisfies the mentioned variant of the [25] assumption. (Currently, VBB obfuscation against semi-bounded adversaries in the ideal graded encoding model is only known for NC^1.)

We also give evidence for the *necessity* of semantically-secure graded encoding for obtaining VGB. Specifically we show that, assuming existence of VGB obfuscators for all circuits, there exist *multilinear jigsaw puzzles* satisfying a form of semantic security. Multilinear jigsaw puzzles, defined in [19], are a limited-functionality variant of multilinear maps. They suffice for obtaining the positive result described in Item 2 above.

The rest of the introduction provides a more detailed overview of our results. Section 1.1 presents the implication from SIO to VGB and VBB obfuscation. Section 1.2 provides background on graded encoding schemes and the semantic security assumption. Section 1.3 presents the construction of SIO from semantically-secure graded encoding schemes.

1.1 From SIO to VGB and VBB Obfuscation

We first define SIO a bit more precisely. A distribution $\tilde{C}$ over circuits is said to be ν-*concentrated* around a boolean function f if for any value x in the domain of f we have that $\Pr[\tilde{C}(x) \neq f(x)] \leq \nu$. We say that $\tilde{C}$ is simply *concentrated* if it is

[1] An alternative view of the definition (which turns out to be equivalent) is that $\mathcal{O}(C_0) \approx \mathcal{O}(C_1)$ if no semi-bounded adversary can distinguish oracle access to a circuit sampled from C_0 from oracle access to a circuit sampled from C_1.

ν-concentrated for some negligible function ν and function f. An obfuscator $\mathcal{O}$ is SIO for a class $\mathcal{C}$ of circuits if for any two (not necessarily efficiently samplable) distributions $\tilde{C}_0, \tilde{C}_1$ over circuits in $\mathcal{C}$ that are concentrated around the same function, it holds that $\mathcal{O}(\tilde{C}_0)$ and $\mathcal{O}(\tilde{C}_1)$ are computationally indistinguishable.
We show the following.

Theorem 1.1 (Informal) *An obfuscator is SIO for a class $\mathcal{C}$ of circuits if and only if it is worst-case VGB obfuscator for $\mathcal{C}$.*

Theorem 1.1 motivates the study of SIO as an independent notion, beyond its role in the construction of worst-case VGB obfuscation for NC^1 from semantically-secure graded encodings. Future results obtaining SIO for more functions, or based on different assumptions, will directly apply for VGB obfuscation as well. We note that existing candidate indistinguishability obfuscators *for all circuits* [1,2,8–10,19,21,28], may also be considered as candidates for SIO, and thus also for VGB obfuscation, for all circuits.

Ideas behind the proof of Theorem 1.1 Showing that VGB implies SIO is straightforward. In the other direction, we show how SIO implies the existence of a semi-bounded simulator $\mathcal{S}$ for any computationally bounded adversary $\mathcal{A}$.

Recall that, for *any* target circuit $C^* \in \mathcal{C}$ in the given collection $\mathcal{C}$, the all-powerful simulator $\mathcal{S}$ should simulate what $\mathcal{A}$ learns from an obfuscation $\mathcal{O}(C^*)$, given only polynomially many oracle queries to C^*. Following previous work in the context of worst-case obfuscation, the distinguishing gap between the outputs of $\mathcal{A}$ and $\mathcal{S}$, often called *the simulation accuracy*, can be bounded by an arbitrarily small (inverse) polynomial, when we allow the number of oracle queries that $\mathcal{S}$ makes to grow with this polynomial. (See Remark 2.2).

The high level idea is as follows: $\mathcal{S}$ will use its oracle to C^* in order to learn enough information about C^*. $\mathcal{S}$ does so by gradually reducing the set $\mathcal{K}$ of possible candidates for the circuit C^*, starting from $\mathcal{K}_0 = \mathcal{C}$, and continuing with progressively smaller sets of candidates:

$$\mathcal{K}_j \subsetneq \mathcal{K}_{j-1} \subsetneq \cdots \subsetneq \mathcal{K}_0 = \mathcal{C}.$$

$\mathcal{S}$ will continue this process until it obtains a set $\mathcal{K}^*$ where $\mathcal{A}$ cannot distinguish an obfuscation $\mathcal{O}(C^*)$ of the target circuit C^* from an obfuscation $\mathcal{O}(C)$ of a random circuit C in $\mathcal{K}^*$. $\mathcal{S}$ will then simulate the output of $\mathcal{A}$ by executing $\mathcal{A}$ on an obfuscation $\mathcal{O}(C)$ of a random circuit C in $\mathcal{K}^*$.

To carry out this plan, $\mathcal{S}^{C^*}$ iteratively performs two main steps: *concentration*, and *majority separation*. After each invocation of the two steps, the set of candidates for the circuit C^* shrinks significantly. This process stops when the set $\mathcal{K}^*$ is reached. We will be able to bound the number of iterations as well as the total number of queries made by $\mathcal{S}$ in the process.
Concentration In the concentration step, $\mathcal{S}$ tries to learn C^* in a straightforward way: it queries C^* on a point x_j that splits the current set of candidate circuits $\mathcal{K}_j$ as evenly as possible. Based on the value of $C^*(x_j)$, $\mathcal{S}$ rules out some of the candidates. This process is repeated until there is no point that shrinks the set of candidates by a factor

of at least $1-\varepsilon$. At the end of the concentration step, it is the case that for every point, all but perhaps an ε-fraction of the candidates in $\mathcal{K}_j$ agree on the point. Therefore, the set $\mathcal{K}_j$ is ε-concentrated (we say that a set of circuits is concentrated if the uniform distribution over this set is concentrated). This occurs after at most $\varepsilon^{-1}\log|\mathcal{C}|$ queries. Throughout, ε is a parameter of the simulation chosen such that $1/\varepsilon$ is a polynomial, depending only on $\mathcal{A}$ and on the required simulation accuracy.

Note that the concentration step alone essentially suffices to ensure *average-case* VGB simulation; indeed, it follows from SIO security that when the target circuit C^* is chosen at random from a concentrated set $\mathcal{K}_j$, $\mathcal{A}$ cannot compute any predicate $\pi(C^*)$, given $\mathcal{O}(C^*)$, better than it can given an obfuscation $\mathcal{O}(C)$ of an independent random circuit $C \leftarrow \mathcal{K}_j$.

Majority separation The concentration step alone does not guarantee *worst-case* simulation. In particular, $\mathcal{A}$ may have some hardwired information that allows it to distinguish C^* from a random circuit in $\mathcal{K}_j$. In this case, however, we show that $\mathcal{S}$ can further reduce the set of candidates $\mathcal{K}_j$ by making a query x that *separates* C^* from most of the circuits in $\mathcal{K}_j$; namely, $C^*(x) \neq \mathrm{maj}_{\mathcal{K}_j}(x)$ where $\mathrm{maj}_{\mathcal{K}_j}$ is the majority of all circuits in $\mathcal{K}_j$. We call such a point x a separating point. In fact, we show that there is a *small* set $\mathcal{L}_{\mathcal{A}}(\mathcal{K}_j)$ of *separating queries* such that C^* must disagree with the majority function in $\mathcal{K}_j$ on at least one $x \in \mathcal{L}_{\mathcal{A}}(\mathcal{K}_j)$.

In more detail, we define the set $\mathbb{D}_{\mathcal{A}}(\mathcal{K}_j)$ of *distinguishable circuits* in $\mathcal{K}_j$ as those circuits $C \in \mathcal{K}_j$ such that $\mathcal{A}$ can ε-distinguish $\mathcal{O}(C)$ from $\mathcal{O}(C')$ for a random $C' \leftarrow \mathcal{K}_j$. The set $\mathcal{L}_{\mathcal{A}}(\mathcal{K}_j)$ will consist of roughly $\varepsilon^{-1}\log|\mathcal{C}|$ points that will separate any distinguishable circuit in $\mathbb{D}_{\mathcal{A}}(\mathcal{K}_j)$ from the majority $\mathrm{maj}_{\mathcal{K}_j}(x)$, and our simulator will query the oracle C^* on all points in $\mathcal{L}_{\mathcal{A}}(\mathcal{K}_j)$. If the oracle C^* agrees with $\mathrm{maj}_{\mathcal{K}_j}$ on all points $x \in \mathcal{L}_{\mathcal{A}}(\mathcal{K}_j)$, then $\mathcal{A}$ cannot tell apart $\mathcal{O}(C^*)$ from $\mathcal{O}(C)$ for a random $C \leftarrow \mathcal{K}_j$, in which case, the simulation can be completed. Otherwise, if C^* disagrees with $\mathrm{maj}_{\mathcal{K}_j}$ on some point $x \in \mathcal{L}_{\mathcal{A}}(\mathcal{K}_j)$, $\mathcal{S}$ obtains a new set of candidates $\mathcal{K}_{j+1} \subsetneq \mathcal{K}_j$ which is necessarily smaller by an ε-factor, since $\mathcal{K}_j$ is ε-concentrated.

By iteratively applying the two steps, we either reach some $\mathcal{K}^*$ for which $\mathcal{A}$ cannot distinguish $\mathcal{O}(C^*)$ from $\mathcal{O}(C)$ for a random $C \leftarrow \mathcal{K}^*$, or we have completely exhausted the collection $\mathcal{C}$ and found exactly the circuit C^*. Since we reduce $\mathcal{K}_j$ at each step by a $(1-\varepsilon)$-factor at the least, the process must end after at most $\varepsilon^{-1}\log|\mathcal{C}|$ steps, and at most $\mathrm{poly}(\varepsilon^{-1}\log|\mathcal{C}|)$ queries.

The separating queries But how do we establish the existence of a small set $\mathcal{L}_{\mathcal{A}}(\mathcal{K}_j)$ that separates $\mathbb{D}_{\mathcal{A}}(\mathcal{K}_j)$ from the majority in $\mathcal{K}_j$? Here we rely on the SIO security of $\mathcal{O}$. Specifically, SIO implies that any subset S of the distinguishable circuits $\mathbb{D}_{\mathcal{A}}(\mathcal{K}_j)$ cannot be ε-concentrated around $\mathrm{maj}_{\mathcal{K}_j}$. Indeed, $\mathcal{A}$ distinguishes $\mathcal{O}(C)$, for $C \leftarrow \mathcal{K}_j$ from $\mathcal{O}(C')$ for $C' \leftarrow S \subseteq \mathbb{D}_{\mathcal{A}}(\mathcal{K}_j)$.[2]

Since no S as above is ε-concentrated around $\mathrm{maj}_{\mathcal{K}_j}$, we can show that it is possible to separate all of the circuits in $\mathbb{D}_{\mathcal{A}}(\mathcal{K}_j)$ from $\mathrm{maj}_{\mathcal{K}_j}$ with at most $\varepsilon^{-1}\log|\mathcal{C}|$ points, as required. Specifically, S constructs the set of separating points by iteratively selecting a point that separates as many circuits as possible from the remaining circuits in $\mathrm{maj}_{\mathcal{K}_j}$

[2] For simplicity of exposition, we assume here that the distinguishing gap is always of the same sign, and is thus preserved on any subset of $\mathbb{D}_{\mathcal{A}}(\mathcal{K}_j)$.

(that were not already separated by previously added points). Since in every iteration the set of remaining circuits is not ε-concentrated around $\mathrm{maj}_{\mathcal{K}_j}$, there must be a point that separates at least an ε-fraction of these circuits from $\mathrm{maj}_{\mathcal{K}_j}$. It follows that after adding at most $\varepsilon^{-1} \log |\mathcal{C}|$ points, no circuit in $\mathbb{D}_{\mathcal{A}}(\mathcal{K}_j)$ remains.

On the possibility of VBB obfuscation The simulation strategy described above requires only a polynomial number of queries, however, the overall running time of the simulator may not be bounded in general. Indeed, in the concentration step, finding a point x_j that significantly splits $\mathcal{K}_j$ may require super-polynomial time. Also, in the majority-separation step, while the set $\mathcal{L}_{\mathcal{A}}(\mathcal{K}_j)$ is small, computing it from $\mathcal{K}_j$ may also require super-polynomial time.

Nevertheless, we show that for certain classes of circuits, simulation can be done more efficiently, or even in polynomial time. Specifically, abstracting away from the above simulation process, we consider the notion of *learning via a majority-separation oracle*, where a given circuit C (or more generally a function) in a prescribed family is learned via oracle access to C and oracle access to the majority separation oracle $\mathbb{M}$, which takes as input (the description of) a concentrated sub-family $\mathcal{K}$ that includes C and outputs a point x that separates C from the majority in $\mathcal{K}$.

While the strategy described above shows that any class of circuits can be learned with polynomially many queries to C and $\mathbb{M}$, the learner itself may be inefficient, which results in inefficient simulation. We show that a more efficient learning procedure can sometimes be translated into a more efficient simulation strategy, depending on pattern of queries made by the learning algorithm. We identify several function classes, for which such efficient learning is possible, yielding new feasibility results for worst-case VBB obfuscation. Examples include fuzzy point functions, conjunctions, and constant-dimension linear subspaces.

Connection to previous worst-case VBB/VGB obfuscators The *majority separation technique* is rooted in the *sack of distinguishable points technique* of Canetti [11]. There, and in [27], it was used to get worst-case (simulation-based) VBB obfuscation from an indistinguishability-based notion of obfuscation, for the simple case of point functions. The technique was then extended to VBB obfuscation of constant-dimension hyperplanes [17] and VGB obfuscation of set functions [4]. The majority separation technique generalizes the above for arbitrary functions. Indeed, in the above works, the indistinguishability guarantee considered is equivalent to SIO (for the classes in question). Thus, we get a unified proof for all existing worst-case VBB obfuscation results.

Connection to VGB obfuscation of evasive functions *Evasive* collections are function collections concentrated around the all-zero function. Barak et al. [3] show that average-case VGB obfuscation for *all* evasive collections implies *weak average-case* VGB for all collections.[3] Here "weak" means that the simulator is allowed to make a slightly super-polynomial number of queries.

We show that an obfuscator is SIO for a collection $\mathcal{C}$ of circuits if and only if it is IO for $\mathcal{C}$ and in addition, it is average-case VGB for any evasive sub-collection of $\mathcal{C}$. In

[3] In fact, for concentrated, and in particular evasive, collections, average-case VGB and average-case VBB are equivalent.

particular, it follows that if an obfuscator is IO and average-case VGB for all evasive collections in $\mathcal{C}$, then it is a *worst-case* VGB obfuscator for $\mathcal{C}$. This is incomparable to the Barak et al. result: on the one hand, they do not need to assume that the obfuscator is IO; on the other hand they only show that it is average-case *weak* VGB, rather than worst-case standard VGB.

To better compare the two techniques, let us state the result we would get using our techniques, assuming only average-case VGB for all evasive collections, and without assuming IO (in particular, without assuming SIO). Roughly, we would get a weak kind of obfuscation where any adversary $\mathcal{A}$ has an $\mathcal{A}$-*designated* obfuscator $\mathcal{O}_{\mathcal{A}}$, which may be inefficient. The security guarantee is that $\mathcal{A}$ has a *worst-case* VGB simulator $\mathcal{S}$, so that for any circuit $C \in \mathcal{C}$, it holds that $\mathcal{A}(\mathcal{O}_{\mathcal{A}}(C)) \approx_{\varepsilon} \mathcal{A}(\mathcal{S}^{C})$; namely, $\mathcal{A}$ cannot tell an $\mathcal{A}$-designated obfuscation of C from a circuit sampled by the semi-bounded simulator, using only black-box access to C. The size of circuits output by $\mathcal{O}_{\mathcal{A}}(C)$ is a polynomial $p(|C|)$ that depends only on the class $\mathcal{C}$, but not on $\mathcal{A}$.

Assuming also IO allows us to "switch quantifiers", and show that there is *a single* efficient obfuscator $\mathcal{O}$ that works for *all* adversaries. This obfuscator would simply output an IO obfuscation of C (padded up to size $p(|C|)$). Security against all adversaries would then follow from the fact that IO is the "best-possible" obfuscator [7,22], and thus would achieve the same security as any adversary-designated obfuscator. [4]

1.2 Semantically Secure Graded Encoding Schemes: Background

Before describing how we get SIO from semantically secure graded encoding schemes, we provide some background on the latter.

A graded encoding scheme [18] defines encodings of ring elements R that support certain homomorphic operations. Each element is encoded relative to some control set. The scheme consists of the following algorithms: $\mathsf{InstGen}$ that given a universe set $[k]$, outputs public parameters pp and secret parameters sp, where pp contains a description of a ring R; Encode that given sp, a set $S \subseteq [k]$ and $\alpha \in R$, generates an encoding $[\alpha]_S$; Add and Sub that, given encodings $[\alpha_1]_S$ and $[\alpha_2]_S$, generate encodings $[\alpha_1 + \alpha_2]_S$ and $[\alpha_1 - \alpha_2]_S$ respectively; Mult that, given encodings $[\alpha_1]_{S_1}$ and $[\alpha_2]_{S_2}$ such that $S_1 \cap S_2 = \emptyset$, generates an encoding $[\alpha_1 \cdot \alpha_2]_{S_1 \cup S_2}$; and isZero that given an encoding $[\alpha]_{[k]}$ outputs 1 if and only if $\alpha = 0$ (all the algorithms above also take as input pp).

Given a sequence of encodings, the operations above can be used to test if certain arithmetic expressions vanish on the encoded elements. The high-level approach of Pass et al. [25] is to devise a security property that hides the value of any arithmetic expression that cannot be evaluated using the permitted operations. In other words, the encoding scheme should amount to an "ideal encoding scheme", where encodings are truly accessed only through permitted algebraic operations. This may, in particular, allow leveraging the existing proofs of VBB security in the ideal graded encoding model [1,8,9,28].

[4] We note that, in the body, our actual proof relies directly on SIO, which we show to follow from average-case VGB for evasive collections and standard IO.

More specifically, Pass et al. take the following approach (described first in an oversimplified manner). Consider a *message sampler* $\mathbb{M}([k], R)$ that samples a tuple $(S_1, m_1), \ldots, (S_\ell, m_\ell)$ from one of two distributions $\mathcal{D}_0$ or $\mathcal{D}_1$, where each $S_i \subseteq [k]$, each $m_i \in R$, and ℓ is polynomial in the security parameter. We say that the sampler is *admissible* if no polynomially-bounded "algebraic adversary" that is given $\vec{S} = (S_1, \ldots, S_\ell)$, and can access the ring elements $\vec{m} = (m_1, \ldots, m_\ell)$ only via *an ideal encoding oracle*, is able to tell whether $(\vec{S}, \vec{m})$ were taken from $\mathcal{D}_0$ or $\mathcal{D}_1$. The ideal encoding oracle only allows the same algebraic manipulations allowed by the graded encoding interface, or put abstractly, it allows the adversary to choose any arithmetic circuit C that respects the set structure given by $\vec{S}$, and test whether $C(\vec{m}) = 0$. The requirement is that, for such an admissible sampler, an efficient adversary that obtains actual encodings $\left\{[m_i]_{S_i} : i \in [\ell]\right\}$, along with the corresponding public parameters pp, also cannot tell whether $(\vec{S}, \vec{m})$ was sampled from $\mathcal{D}_0$ or $\mathcal{D}_1$.

As noticed by Pass et al., the assumption formulated above is actually false—it is susceptible to a diagonalization attack in the spirit of the [7] impossibility result for general VBB obfuscation. To get around this caveat, Pass et al. strengthen the admissibility requirement to require that $\mathcal{D}_0, \mathcal{D}_1$ are indistinguishable even to a *semi-bounded* algebraic adversary, namely an algebraic adversary that is computationally unbounded, but makes only a polynomial number of queries to the ideal graded encoding oracle. Furthermore, even this relaxed assumption suffices for obtaining IO in the plain model.

To get IO in the plain model, the idea is to rely on a construction of VBB obfuscation for NC^1 in the ideal graded encoding model [8,9] and replace the ideal graded encoding with a concrete graded encoding scheme satisfying semantic security as stated above. To show that obfuscations of two equivalent circuits C_0, C_1 are indeed indistinguishable, consider a message sampler $\mathbb{M}$ that samples a pair of distributions $\mathcal{D}_0, \mathcal{D}_1$ such that the distribution $\mathcal{D}_i$ is an obfuscation of the circuit C_i in the ideal graded encoding model. The admissibility of this sampler follows from the fact that the [8] obfuscator is secure even against *semi-bounded* adversaries. Specifically, an algebraic adversary accessing $\mathcal{D}_i$ via an ideal encoding oracle essentially has black-box access to the circuit C_i. Since C_0, C_1 cannot be distinguished given only black-box access, their obfuscations are indistinguishable as well.

In this work we consider the relaxed semantic security requirement above. Existing graded encoding candidates [15,16,18] do not satisfy this requirement as demonstrated by recent attacks [13,14,18,24]. The eventual Pass et al. assumption is further relaxed in several ways, while still yielding their main application to IO. See [6,25] for a discussion on relaxations of the semantic security assumption.

1.3 SIO from Semantically Secure Graded Encoding, and Back Again

We sketch our variant of the semantic security assumption, and explain how we obtain SIO for NC^1 circuits from this variant. We also give evidence for the *necessity* of semantic security for obtaining SIO.

Essentially, the reason that semantic security of graded encoding schemes implies SIO is that semantic security considers *any* admissible distributions over encodings, not

only ones that come out of obfuscating a given program. In particular, distributions that consist of ideal-graded-encoding-VBB obfuscations of circuits that are concentrated around the same function are admissible, thus their instantiations via a semantically secure graded encoding scheme are guaranteed to be indistinguishable.

However, some care has to be taken here: note that SIO considers even distributions that are not necessarily efficiently samplable. (Indeed, this property is crucial in the proof that SIO implies worst-case VGB.) This means that we will need to somewhat modify the formulation of the semantic security assumption.

A naive attempt to formalize this variant of semantic security may simply allow the sampler to be computationally unbounded. However, recall that the message sampler is given the description of a ring R. (This is required in order to sample obfuscations in the ideal graded encoding model that consist, for example, of random elements in R.) A computationally unbounded sampler that sees R may be able to recover information that compromises the security of the encodings (for example, the secret parameters). The sampler can produce encodings that reveal this secret information. Note that such a sampler may still be admissible since learning the secret parameters gives no advantage to an algebraic adversary.

Instead we sample messages in two stages: first, an unbounded sampler S generates a polynomial-size auxiliary input string s; second, an *efficient* encoder $\mathbb{M}$ gets the ring R and the auxiliary input string s, and generates the final samples. We call this variant *strong-sampler semantic security.*

Theorem 1.2 (Informal) *Let $\mathcal{O}$ be any obfuscator for a class $\mathcal{C}$ of circuits, that is VBB against semi-bounded adversaries in the ideal graded encoding model. Then instantiating the graded encoding oracle with a strong-sampler semantically-secure graded encoding scheme results in an obfuscator $\mathcal{O}'$ that is SIO for $\mathcal{C}$ in the plain model.*

Then, relying on the Barak et al. obfuscation for NC^1 in the ideal graded encoding model [8] (which is indeed VBB against semi-bounded adversaries), we obtain the following corollary.

Corollary 1.1 (Informal) *Assume there exists a strong-sampler semantically-secure encoding scheme. Then there exists SIO for NC^1.*

We also give evidence for the *necessity* of semantically-secure graded encoding schemes for obtaining VGB. To this end, we focus on a version of graded encoding with restricted functionality called *multilinear jigsaw puzzles* [19]. Unlike graded encodings, in multilinear jigsaw puzzles, encodings can only be generated together with the system parameters. We refer to the public parameters, together with the set of initialized encodings, as a puzzle. Instead of performing individual permitted operations over the encodings, all the jigsaw puzzle user can do is to specify an arithmetic circuit C that respects the set structure of the initialized encodings, and test whether C evaluates to 0 on these encodings or not.

Semantically secure multilinear jigsaw puzzles are defined similarly to the graded encoding case and their existence follows from semantically secure graded encodings. Despite their restricted functionality, semantically secure multilinear jigsaw puzzles can replace graded encodings in our construction of SIO for NC^1. In the body of this work we present the construction of SIO based on multilinear jigsaw puzzles.

We observe that the existence of semantically secure jigsaw puzzles is implied by VGB obfuscation for all circuits. To see why this is the case, consider the circuit P that has a set of ring elements $\vec{m} = (m_1, \ldots, m_\ell)$ hardwired into it, together with the corresponding sets $\vec{S} = (S_1, \ldots, S_\ell)$. The circuit P takes as input an arithmetic circuit C that respects the set structure given by $\vec{S}$, and tests whether $C(\vec{m}) = 0$. To initialize a puzzle from a set of encodings $(\vec{S}, \vec{m})$, we simply VGB obfuscate the circuit P. The semantic security of this puzzle follows directly from the simulation security of the VGB obfuscation.

2 Obfuscation: VBB, VGB, Indistinguishability

2.1 Preliminaries

The cryptographic definitions in the paper follow the convention of modeling security against non-uniform adversaries. A function μ is said to be negligible if $\mu(n) = n^{-\omega(1)}$. We denote an arbitrary negligible function of n by $\mathrm{negl}(n)$. We consider collections of polynomial-size circuits $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$, such that all circuits $C \in \mathcal{C}_n$ are of the same polynomial size $n^{O(1)}$ with $n^{O(1)}$ input and output bits. An efficient adversary $\mathcal{A}$ is modeled as a sequence of (perhaps probabilistic) circuits $\mathcal{A} = \{\mathcal{A}_n\}_{n\in\mathbb{N}}$, such that each circuit $\mathcal{A}_n$ is of polynomial size $n^{O(1)}$ with $n^{O(1)}$ input and output bits. We often omit the subscript n when it is clear from the context and simply refer to such a collection or an adversary as *polynomial-size*. For a randomized algorithm $\mathcal{A}$ we denote by $\Pr_{\mathcal{A}}[E]$ the probability over the randomness of $\mathcal{A}$ that the event E occurs.

2.2 Obfuscation

An obfuscator $\mathcal{O}$ is a PPT algorithm that takes as input a circuit C and outputs an obfuscated circuit $\mathcal{O}(C)$. We next review three basic definitions of obfuscation that are used throughout the paper. We start by defining the functionality requirement, which all the notions share, and then define different security notions.

Definition 2.1 (*Functionality*) Let $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ be a collection of polynomial-size circuits such that any circuit $C \in \mathcal{C}_n$ takes inputs of length n. A PPT algorithm $\mathcal{O}$ is an obfuscator for the collection $\mathcal{C}$ if for any $C \in \mathcal{C}$,

$$\Pr_{\mathcal{O}}[\forall x : \mathcal{O}(C)(x) = C(x)] = 1.$$

Remark 2.1 (Obfuscation of NC^1) We say that $\mathcal{O}$ is an obfuscator for NC^1 if it is an obfuscator for every collection of equal-depth circuits in NC^1. A collection of polynomial-size circuits $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ is said to be in NC^1 if there exists a constant $c \in \mathbb{N}$ such that the depth of every circuit $C \in \mathcal{C}_n$ is at most $c \cdot \log n$. The collection is equal-depth if all circuits are of depth exactly $c \cdot \log n$.

2.3 VBB and VGB Obfuscation

Virtual Black Box (VBB) obfuscation [7] guarantees that an obfuscated circuit $\mathcal{O}(C)$ does not reveal any predicate $\pi(C)$ that cannot be learned by an efficient simulator that is given only black-box access to C. The basic definition is *worst-case* in the sense that the simulator needs to be successful for any circuit in a given circuit collection. We later also address an *average-case* notion. In the definition below we use a slightly weaker definition than the standard one, and allow the simulator to depend on the distinguishing probability p (See Remark 2.2).

Definition 2.2 (*Worst-case VBB Obfuscation*) An obfuscator $\mathcal{O}$ for a collection of polynomial-size circuits $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ is worst-case VBB if for every polynomial-size adversary $\mathcal{A}$, and polynomial p, there exists a polynomial-size simulator $\mathcal{S}$, such that for every $n \in \mathbb{N}$, every predicate $\pi : \mathcal{C}_n \to \{0,1\}$, and every $C \in \mathcal{C}_n$:

$$\left| \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C)) = \pi(C)] - \Pr_{\mathcal{S}}[\mathcal{S}^C(1^n) = \pi(C)] \right| \leq 1/p(n).$$

Virtual Grey Box (VGB) obfuscation [4] relaxes VBB by allowing the simulator to have unbounded computational power, but still only a bounded number of oracle queries to C.

Definition 2.3 (*Worst-case VGB Obfuscation*) An obfuscator $\mathcal{O}$ for a collection of polynomial-size circuits $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ is worst-case VGB if for every polynomial-size adversary $\mathcal{A}$, and polynomial p, there exists an unbounded simulator $\mathcal{S}$, and a polynomial q, such that for every $n \in \mathbb{N}$, every predicate $\pi : \mathcal{C}_n \to \{0,1\}$, and $C \in \mathcal{C}_n$:

$$\left| \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C)) = \pi(C)] - \Pr_{\mathcal{S}}[\mathcal{S}^{C[q(n)]}(1^n) = \pi(C)] \right| \leq 1/p(n),$$

where $C[q(n)]$ is an oracle that allows at most $q(n)$ queries.

We also consider relaxed versions of VBB and VGB, where the corresponding guarantee only holds for a random circuit sampled from a distribution, rather than for any circuit.

Definition 2.4 (*Average-case obfuscation*) Each of Definitions 2.2, 2.3 is said to hold in the average case, for a distribution ensemble $\tilde{\mathcal{C}} = \bigcup_{n\in\mathbb{N}} \tilde{\mathcal{C}}_n$ on the collection $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$, if each of the corresponding probability statements is over a random $C \leftarrow \tilde{\mathcal{C}}_n$, rather than required for every $C \in \mathcal{C}$.

Remark 2.2 (Simulation accuracy) In the above definitions (and throughout the paper), the simulator $\mathcal{S}$, and in VGB, also its number of queries q, are allowed to depend on the required simulation accuracy $1/p(n)$. This is the case in all previous works that have established worst-case VBB or VGB (for specific classes) [4,11,17,27]. This definition is implied by original definition of [7] where the same simulator $\mathcal{S}$ should be $1/p(n)$-accurate for all polynomials p (for large enough security parameter n).

Indistinguishability Obfuscation We next define the notion of indistinguishability obfuscation, introduced in [7].

Definition 2.5 (*Indistinguishability obfuscation* [7]) An obfuscator for a collection of polynomial-size circuits $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ is said to be an *indistinguishability obfuscator*, denoted by $i\mathcal{O}$, if for any polynomial-size distinguisher $\mathcal{D}$, there exists a negligible function μ such that for all $n \in \mathbb{N}$, and any two circuits $C_0, C_1 \in \mathcal{C}_n$ of the same size and functionality,

$$\Pr\left[b \leftarrow \{0,1\}; \mathcal{D}(C_0, C_1, i\mathcal{O}(C_b)) = b\right] \leq \frac{1}{2} + \mu(n).$$

It can be readily seen that if an obfuscator $\mathcal{O}$ is VBB for a function collection $\mathcal{C}$ then it is also VGB for $\mathcal{C}$. Furthermore, if $\mathcal{O}$ is VGB for $\mathcal{C}$ then it is also an indistinguishability obfuscator for $\mathcal{C}$.

3 Strong Indistinguishability Obfuscation

In this section we define the notion of strong indistinguishability obfuscation (SIO). We start by defining the notion of concentrated distributions over circuits.

3.1 Concentrated Circuit Distributions

At a high-level, a distribution ensemble $\tilde{\mathcal{C}}$, over a circuit collection $\mathcal{C}$, is *concentrated*, if given polynomially many oracle queries to a random circuit C from the distribution, it is information theoretically hard to find an input x such that C does not agree with $\mathrm{maj}_{\tilde{\mathcal{C}}}$ on the point x, where $\mathrm{maj}_{\tilde{\mathcal{C}}}$ is the common output of circuits distributed according to $\tilde{\mathcal{C}}$. If $\tilde{\mathcal{C}}$ corresponds to the uniform distribution on some collection $\mathcal{C}$, $\mathrm{maj}_{\tilde{\mathcal{C}}}$ is simply the majority vote. Concentrated distributions naturally generalize the concept of *evasive distributions* studied in [3], in which the majority is always the all-zero function, i.e. $\mathrm{maj}_{\tilde{\mathcal{C}}} \equiv 0$.

Definition 3.1 (*Concentrated circuit distributions*) Let $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ be a polynomial-size circuit collection, where $\mathcal{C}_n$ consists of circuits $C : \{0,1\}^n \to \{0,1\}$, and let $\tilde{\mathcal{C}}_n$ be a distribution on $\mathcal{C}_n$. Let $\mathrm{maj}_{\tilde{\mathcal{C}}_n(x)} := \lfloor \mathbb{E}_{C\leftarrow\tilde{\mathcal{C}}_n} C(x) \rceil$ be the common output at point x of circuits drawn from $\tilde{\mathcal{C}}_n$.

1. For any $\varepsilon \in [0,1]$, $\tilde{\mathcal{C}}_n$ is said to be ε-concentrated if

$$\max_{x\in\{0,1\}^n} \Pr_{C\leftarrow\tilde{\mathcal{C}}_n}\left[C(x) \neq \mathrm{maj}_{\mathcal{C}_n}(x)\right] \leq \varepsilon.$$

2. $\tilde{\mathcal{C}}$ is said to be concentrated if for some negligible $\mu(\cdot)$, and any $n \in \mathbb{N}$, $\tilde{\mathcal{C}}_n$ is $\mu(n)$-concentrated.
3. $\tilde{\mathcal{C}}$ is said to be evasive if it is concentrated, and for any $n \in \mathbb{N}$ and any $x \in \{0,1\}^n$, $\mathrm{maj}_{\tilde{\mathcal{C}}_n(x)} = 0$.

4. We say that the collection $\mathcal{C}$ itself is concentrated (evasive) if the uniform distribution ensemble on circuits in $\mathcal{C}$ is concentrated (evasive).

3.2 Strong Indistinguishability Obfuscation

Strong Indistinguishability Obfuscation requires that indistinguishability holds, even when C_0 and C_1 do not necessarily compute the exact same function, but are taken from two distributions $\tilde{\mathcal{C}}_n^0$ and $\tilde{\mathcal{C}}_n^1$ that are concentrated around the same function; namely, $\text{maj}_{\tilde{\mathcal{C}}_n^0} \equiv \text{maj}_{\tilde{\mathcal{C}}_n^1}$:

Definition 3.2 (*Strong indistinguishability obfuscation*) An obfuscator for $\mathcal{C}$ is said to be a **strong** *indistinguishability obfuscator* for $\mathcal{C}$, denoted by $i\mathcal{O}^*$, if for any two concentrated distribution ensembles $\tilde{\mathcal{C}}^0, \tilde{\mathcal{C}}^1$ on $\mathcal{C}$, such that $\forall n \in \mathbb{N} : \text{maj}_{\tilde{\mathcal{C}}_n^0} \equiv \text{maj}_{\tilde{\mathcal{C}}_n^1}$, and any polynomial-size distinguisher $\mathcal{D}$, there exists a negligible function μ such that for all $n \in \mathbb{N}$,

$$\Pr[b \leftarrow \{0,1\}; (C_0, C_1) \leftarrow (\tilde{\mathcal{C}}_n^0, \tilde{\mathcal{C}}_n^1); \mathcal{D}(i\mathcal{O}^*(C_b)) = b] \leq \frac{1}{2} + \mu(n).$$

We observe that any SIO obfuscator for $\mathcal{C}$ is also an IO obfuscator for $\mathcal{C}$. Indeed, for any two circuits C_0, C_1 of equivalent functionality, each of these circuits on its own is trivially concentrated around their common functionality.

4 SIO is Equivalent to Worst-Case VGB

In this section, we prove that the notion of strong indistinguishability obfuscation (SIO) is equivalent to VGB. Clearly, any VGB obfuscator for a class $\mathcal{C}$ is also a SIO for $\mathcal{C}$. We show that the converse is true as well. Namely, we show that any strong indistinguishability obfuscator $\mathcal{O}$ for a class $\mathcal{C}$ of circuits is a worst-case VGB obfuscator for $\mathcal{C}$. In addition, we show that for classes $\mathcal{C}$ with some additional properties, $\mathcal{O}$ is in fact worst-case VBB. We refer the reader to Sect. 1.1 for an overview.

4.1 Definitions and Statement of Main Theorem

4.1.1 Notation and Terminology

For a function $f : \{0,1\}^n \to \{0,1\}$, we say that a point $x \in \{0,1\}^n$ *separates* a circuit C from f if $C(x) \neq f(x)$. We say that a set $L \subseteq \{0,1\}^n$ separates C from f, if some $x \in L$ separates C from f. Given a circuit collection $\mathcal{K}$, we say that L separates $\mathcal{K}$ from f, if L separates any $C \in \mathcal{K}$ from f. Recall, that we say that a collection $\mathcal{K}$ is concentrated if the uniform distribution on $\mathcal{K}$ is concentrated around its majority function $\text{maj}_{\mathcal{K}}$.

Definition 4.1 (*Majority-separating oracle*) Let $\mathcal{C}$ be a collection of boolean circuits defined over $\{0,1\}^n$, let $C \in \mathcal{C}$, and let $\varepsilon > 0$. An oracle $\mathbb{M}$ is said to be $(\mathcal{C}, C, \varepsilon)$-separating if given any ε-concentrated sub-collection $\mathcal{K} \subseteq \mathcal{C}$, represented by a circuit

that samples uniform elements in $\mathcal{K}$, $\mathbb{M}(\mathcal{K})$ outputs a point $x \in \{0,1\}^n$ that separates C from $\mathrm{maj}_{\mathcal{K}}$, or $\perp$ if no such point exists.

Remark 4.1 In the above definition, and throughout this section, we often abuse notation and denote by $\mathcal{K}$ both the sub-collection and the circuit that samples uniform elements from the sub-collection.

Definition 4.2 (*Learnability by majority-separating oracles*) A collection $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ of boolean circuits is said to be $(t, \sigma, c, s, \varepsilon)$-learnable by a majority-separation oracle if there exists a deterministic oracle-aided machine $\mathcal{L}$ such that, given oracle access to $C \in \mathcal{C}_n$ and a $(\mathcal{C}_n, C, \varepsilon(n))$-separating oracle $\mathbb{M}$, $\mathcal{L}^{C,\mathbb{M}}(1^n)$ outputs $\hat{C} \in \mathcal{C}_n$ of equivalent functionality to C, in time $t(n)$, using at most $s(n)$ queries to $\mathbb{M}$, and most $\sigma(n)$ queries to C before the last call to $\mathbb{M}$, and at most $c(n)$ queries to C overall.

Our main technical theorem shows that any strong indistinguishability obfuscator for a circuit collection $\mathcal{C}$ that is learnable via a majority separation oracle is also a worst-case simulation-based obfuscator. The size and query complexity of the worst-case simulator, in particular whether it is a VBB or VGB simulator, is determined by the learnability parameters $(t, \sigma, c, s, \varepsilon)$.

Theorem 4.1 *Let $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ be a polynomial-size circuit collection, and let $\mathcal{O}$ be a strong indistinguishability obfuscator for $\mathcal{C}$. Let $\mathcal{A}$ be a boolean polynomial-size adversary, and let p be a polynomial. Then there exists a polynomial q, such that if $\mathcal{C}$ is $(t, \sigma, c, s, \frac{1}{q})$-learnable by a majority-separating oracle, and (t, σ, c, s) are polynomially bounded, then $(\mathcal{A}, p)$ has a simulator $\mathcal{S}$ of size $O(|\mathcal{A}| + t \cdot s \cdot q^s \cdot 2^\sigma)$ with $O(c + q \cdot s)$ oracle queries. The simulator works in the worst-case for any $C \in \mathcal{C}$.*

In Sect. 4.3 we show that for any $2 < q \leq n^{O(1)}$, any circuit collection $\mathcal{C}$ is indeed $(t, \sigma, c, s, \frac{1}{q})$-learnable, for some setting of parameters (t, σ, c, s) that are all polynomially bounded.

4.2 Proof of Theorem 4.1

Fix $\mathcal{C}, \mathcal{O}, \mathcal{A}, p$ satisfying the conditions of the theorem. The proof of the theorem will rely on the following key lemma that essentially shows that the set of circuits, whose obfuscation $\mathcal{A}$ can $1/p$-distinguish, can always be separated from the majority of circuits by a small separating set.

Lemma 4.1 *There exists a polynomial q, such that for any $n \in \mathbb{N}$, and any $\frac{1}{q(n)}$-concentrated sub-collection $\mathcal{K} \subseteq \mathcal{C}_n$, there exists a set $L(\mathcal{K}) \subseteq \{0,1\}^n$ of size at most $q(n)$, such that for any $C \in \mathcal{K}$ that is not separated from $\mathrm{maj}_{\mathcal{K}}$ by $L(\mathcal{K})$:*

$$\left| \Pr[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C' \leftarrow \mathcal{K}}[\mathcal{A}(\mathcal{O}(C')) = 1] \right| \leq 1/2p(n),$$

where the probability is also over the coins of $\mathcal{A}$ and $\mathcal{O}$.

Proof For any $n \in \mathbb{N}$, and sub-collection $\mathcal{K} \subseteq \mathcal{C}_n$, let us denote by $\mathbb{D}^b(\mathcal{K})$ the collection of all circuits $C \in \mathcal{K}$ such that

$$(-1)^b \left(\Pr\left[\mathcal{A}(\mathcal{O}(C)) = 1\right] - \Pr_{C' \leftarrow \mathcal{K}}\left[\mathcal{A}(\mathcal{O}(C')) = 1\right] \right) \geq 1/2p(n);$$

namely, $\mathbb{D}(\mathcal{K}) := \mathbb{D}^0(\mathcal{K}) \cup \mathbb{D}^1(\mathcal{K})$ consist of all the "distinguishable circuits" in $\mathcal{K}$.

Assume towards contradiction that the lemma does not hold. Then there exists a super-polynomial function $T(n) = n^{\omega(1)}$, such that for an infinite sequence $\mathbb{N}^* \subseteq \mathbb{N}$, and any $n \in \mathbb{N}^*$, there exists a $\frac{1}{T(n)}$-concentrated sub-collection $\mathcal{K}_n \subseteq \mathcal{C}_n$ such that any set $L(\mathcal{K}_n)$ separating the distinguishable circuits $\mathbb{D}^0(\mathcal{K}_n) \cup \mathbb{D}^1(\mathcal{K}_n)$ from $\mathrm{maj}_{\mathcal{K}_n}$ is of size greater than $T(n)$. In particular, for some $b_n \in \{0, 1\}$, any set separating $\mathbb{D}^{b_n}(\mathcal{K}_n)$ from $\mathrm{maj}_{\mathcal{K}_n}$ is of size greater than $T(n)/2$. For ease of notation, let us assume throughout that $b_n = 0$, and simply denote $\mathbb{D}(\mathcal{K}_n) = \mathbb{D}^0(\mathcal{K}_n)$. (This is indeed WLOG, by flipping the output of $\mathcal{A}$ if needed.) □

Claim 4.1 *For any $n \in \mathbb{N}^*$, there exists a non-empty concentrated sub-collection $\mathbb{D}^*(\mathcal{K}_n) \subseteq \mathbb{D}(\mathcal{K}_n)$. Specifically,*

$$\max_{x \in \{0,1\}^n} \left\{ \Pr_{C \leftarrow \mathbb{D}^*(\mathcal{K}_n)} \left[C(x) \neq \mathrm{maj}_{\mathcal{K}_n}(x) \right] \right\} \leq \alpha(n) := \frac{2 \log |\mathcal{C}_n|}{T(n)}.$$

Proof We describe an iterative process that results in the required $\mathbb{D}^*(\mathcal{K}_n)$. Let $\mathbb{D}_0 = \mathbb{D}(\mathcal{K}_n)$, and let $L_0 = \emptyset$. Given $\mathbb{D}_i$, we define $\mathbb{D}_{i+1}$ as follows. If $\mathbb{D}_i$ satisfies the property given by the claim, output $\mathbb{D}^*(\mathcal{K}_n) = \mathbb{D}_i$. Otherwise, there exists a point $x_i \in \{0, 1\}^n$ that separates an $\alpha(n)$-fraction of the circuits in $\mathbb{D}_i$ from $\mathrm{maj}_{\mathcal{K}_n}$. Then, add x_i to current partial separating set $L_{i+1} = L_i \cup \{x_i\}$, and let $\mathbb{D}_{i+1} \subseteq \mathbb{D}_i$ be the sub-collection that is not separated from $\mathrm{maj}_{\mathcal{K}_n}$ by x_i.

Note that this process ends after at most $T(n)/2$ steps (we do not require that it is efficient). Indeed, it holds that for $i \leq T(n)/2$ and for $\alpha = \frac{2 \log |\mathcal{C}_n|}{T(n)}$,

$$|\mathbb{D}_i| \leq (1 - \alpha(n)) \, |\mathbb{D}_{i-1}| \leq (1 - \alpha(n))^i \, |\mathbb{D}| \leq \left(1 - \frac{2 \log |\mathcal{C}_n|}{T}\right)^i |\mathcal{C}_n| \leq |\mathcal{C}_n|^{1 - \frac{2i}{T(n)}}.$$

Moreover, this process must end with a non-empty set. This is the case since otherwise after $T(n)/2$ steps we separated all of the original $\mathbb{D}(\mathcal{K}_n)$ from $\mathrm{maj}_{\mathcal{K}_n}$ with a set $L_{T(n)/2} \subseteq \{0, 1\}^n$ of size less than $T(n)/2$. This contradicts the fact that $\mathbb{D}(\mathcal{K}_n)$ cannot be separated from $\mathrm{maj}_{\mathcal{K}_n}$ by $T(n)/2$ elements or less.

We now show how to violate the fact that $\mathcal{O}$ is a SIO obfuscator for $\mathcal{C}$. Consider the concentrated sub-collections $\mathcal{D}^* = \bigcup_{n \in \mathbb{N}^*} \mathbb{D}^*(\mathcal{K}_n)$, and $\mathcal{K} = \bigcup_{n \in \mathbb{N}} \mathcal{K}_n$. (Formally, we need to also define these for $n \in \mathbb{N} \setminus \mathbb{N}^*$. We can do so in an arbitrary way that will keep them concentrated.) Since both $\mathbb{D}^*(\mathcal{K}_n)$ and $\mathcal{K}_n$ are concentrated around $\mathrm{maj}_{\mathcal{K}_n}$, it suffices to show that $\mathcal{A}$ distinguishes

$$\{\mathcal{O}(C) : C \leftarrow \mathcal{K}_n\}_{n \in \mathbb{N}} \text{ from } \left\{\mathcal{O}(C) : C \leftarrow \mathbb{D}^*(\mathcal{K}_n)\right\}_{n \in \mathbb{N}}.$$

Indeed, for any $n \in \mathbb{N}^*$

$$\Pr_{C \leftarrow \mathbb{D}^*(\mathcal{K}_n)}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \mathcal{K}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] \geq$$

$$\min_{C \in \mathbb{D}^*(\mathcal{K}_n)} \Pr[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \mathcal{K}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] \geq \frac{1}{2p(n)}.$$

□

To complete the proof of Theorem 4.1, let us fix q to be the polynomial given by Lemma 4.1 corresponding to $(\mathcal{A}, p)$, and assume that $\mathcal{C}$ is $(t, \sigma, c, s, \frac{1}{q})$-learnable by a majority-separating oracle. We next describe the simulator $\mathcal{S}$ for $(\mathcal{A}, p)$, argue its validity, and analyze its complexity.

Description of $\mathcal{S}$. Given oracle access to $C \in \mathcal{C}_n$, $\mathcal{S}$ runs the learner $\mathcal{L}^{C,\mathbb{M}}(1^n)$ given by Definition 4.2, and emulates for $\mathcal{L}$ the oracle C and the majority-separating oracle $\mathbb{M}$. Any call to C is answered by $\mathcal{S}$ using its own oracle to C. Oracle calls to $\mathbb{M}$ are handled as follows. Given a sub-collection $\mathcal{K} \subseteq \mathcal{C}_n$ that contains C (represented by a circuit that samples uniform elements in $\mathcal{K}$), $\mathcal{S}$ first retrieves the set $L(\mathcal{K})$ separating the distinguishable circuits $\mathbb{D}(\mathcal{K}) \subseteq \mathcal{K}$ from $\mathrm{maj}_{\mathcal{K}}$. Then, $\mathcal{S}$ queries its oracle C on all the points $x \in L(\mathcal{K})$, and tests whether $C(x) = \mathrm{maj}_{\mathcal{K}}(x)$, namely whether x separates C from $\mathrm{maj}_{\mathcal{K}}$. The computation of $\mathrm{maj}_{\mathcal{K}}(x)$ is done by computing $C_1(x), \ldots, C_n(x)$, where each $C_i \leftarrow \mathcal{K}$ is a random circuit from $\mathcal{K}$ and taking their majority.

If $\mathcal{S}$ found a separating point x, then it uses it to answer $\mathcal{L}$'s query, and continues its emulation. Otherwise, if no point in $L(\mathcal{K})$ separates C from $\mathrm{maj}_{\mathcal{K}}$, then $\mathcal{S}$ stops the emulation of $\mathcal{L}$, samples a random $C' \leftarrow \mathcal{K}$, and outputs the result of running $\mathcal{A}(\mathcal{O}(C'))$. In any case, after running $\mathcal{L}$ for at most $t(n)$ steps, $\mathcal{L}$ would output $\hat{C} \in \mathcal{C}_n$ of equivalent functionality to C, and $\mathcal{S}$ outputs the result of running $\mathcal{A}(\mathcal{O}(\hat{C}))$.

Validity The validity of $\mathcal{S}$ follows from Lemma 4.1, the guarantee on $\mathcal{L}$, and the correctness of computing $\mathrm{maj}_{\mathcal{K}}$. In more detail, we first condition on the event that the simulator $\mathcal{S}$ always computes $\mathrm{maj}_{\mathcal{K}}$ correctly. In this case, by Lemma 4.1, if at any point C agrees with $\mathrm{maj}_{\mathcal{K}}$ on all of $\mathcal{L}(\mathcal{K})$, then $\mathcal{A}$ distinguishes an obfuscation $\mathcal{O}(C)$ from an obfuscation $\mathcal{O}(C')$ for a random $C' \leftarrow \mathcal{K}$ with probability at most $\frac{1}{2p(n)}$. Otherwise, we successfully implement a $(\mathcal{C}_n, C, \frac{1}{q(n)})$-majority-separating oracle $\mathbb{M}$, and learn $\hat{C} \in \mathcal{C}_n$ of equivalent functionality to C. Since $\mathcal{O}$ is a strong indistinguishability obfuscator, and in particular an indistinguishability obfuscator, $\mathcal{A}$'s advantage is also bounded by $1/2p(n)$.

To complete the argument, we note that the condition that $\mathcal{S}$ always computes $\mathrm{maj}_{\mathcal{K}}$ holds except with negligible probability $2^{-\Omega(n)} \cdot (c+qs) \leq 1/2p(n)$. Indeed, each such computations is correct except with probability $2^{-\Omega(n)}$ (see Claim 5.1), and the bound follows by taking a union bound over the total number of queries $(c + qs) = n^{O(1)}$ made by $\mathcal{S}$ (calculated in detail below). Thus, the overall simulation error is bounded by $1/2p(n) + 1/2p(n) \leq 1/p(n)$, as required.

Complexity of $\mathcal{S}$. The queries of $\mathcal{S}$ to C include the $c(n)$ queries that $\mathcal{L}$ makes to C, and $q(n)$ queries for each of the $s(n)$ queries made by $\mathcal{L}$ to $\mathbb{M}$; indeed, remember that the size of each $\mathcal{L}(\mathcal{K})$ is bounded by $q(n)$. Thus the overall query complexity of $\mathcal{S}$ is $c(n) + s(n) \cdot q(n)$.

The total circuit size of $\mathcal{S}$ can be bounded by a fixed polynomial in

1. the size of the adversary $|\mathcal{A}|$,
2. the total size $t(n)$ of $\mathcal{L}$,
3. the number of sets $L(\mathcal{K})$ that the simulator may have to use throughout the simulation, times the size $q(n)$ of each $L(\mathcal{K})$,
4. the time it takes to compute the values $\mathrm{maj}_{\mathcal{K}}(x)$ throughout.

We now count the number of sets $L(\mathcal{K})$ necessary for $\mathcal{S}$. In what follows, for $1 \leq i \leq s+1$, we denote by $c_i(n)$ the number of queries made by $\mathcal{L}$ to the oracle C between the $i-1$-st and i-th queries to $\mathbb{M}$. Note $c(n) = \sum_{i=1}^{s+1} c_i(n)$ and that $\sigma(n) = \sum_{i=1}^{s} c_i(n)$. For ease of notation, from hereon we suppress the security parameter n.

Consider the (deterministic) learner $\mathcal{L}$, we consider its tree of possible executions. We view each node at level $0 \leq i \leq s$, as corresponding to the state of $\mathcal{L}$ before making c_{i+1} queries to C and the $i+1$-st query to $\mathbb{M}$. In this tree, a node at the i-th level has $2^{c_{i+1}} \cdot q$ sons. Indeed, there are at most $2^{c_{i+1}}$ possible sequences of queries and answers made by $\mathcal{L}$ to the oracle C, before the $i+1$-st query to $\mathbb{M}$; then, each such possible sequence determines a set $L(\mathcal{K})$ of q values that $\mathcal{L}$ will query $\mathbb{M}$ on. The overall number of sets $L(\mathcal{K})$ is thus

$$\sum_{i=0}^{s-1} q^i \cdot \prod_{j=1}^{i+1} 2^{c_j} \leq s \cdot q^{s-1} \prod_{i=1}^{s} 2^{c_i} = s \cdot q^{s-1} \cdot 2^{\sigma}.$$

Throughout the simulation, $\mathrm{maj}_{\mathcal{K}}(x)$ is computed at most $s(n) \cdot q(n)$ times. The computation itself is done by computing $C_1(x), \ldots, C_n(x)$, where each $C_i \leftarrow \mathcal{K}$ is a random circuit from $\mathcal{K}$ and taking their majority. Sampling $C_i \leftarrow \mathcal{K}$ and computing $C_i(x)$, can be done in time at most $t(n)$ (i.e., the size of $\mathcal{L}$) since we assume that the learner $\mathcal{L}$ represents each of its queries $\mathcal{K}$ to its majority-separating oracle $\mathbb{M}$ by a circuit that samples uniform elements in $\mathcal{K}$.

This completes the proof of Theorem 4.1.

4.3 VGB and VBB by Majority-Separation Learning

In this section, we show that any class of circuits is learnable by a majority-separating oracle, with parameters that yield VGB simulation. We then discuss additional classes that can be learned with better parameters, yielding VBB simulation. This includes previously obfuscated classes as well as new ones.

4.3.1 VGB Obfuscation for All Circuits

We show

Theorem 4.2 *Let $\mathcal{C}$ be any circuit collection and let $\mathcal{O}$ be a strong indistinguishability obfuscator for $\mathcal{C}$. Then $\mathcal{O}$ is also a worst-case VGB obfuscator for $\mathcal{C}$.*

To prove Theorem 4.2, we show that any circuit collection is learnable by a majority-separating oracle, where the learner is of unbounded size, but only performs a polynomial number of queries to its oracles. Theorem 4.2 then follows from Theorem 4.1.

Lemma 4.2 *For any $q > 2$, any circuit collection $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ is $(t, \sigma, c, s, \frac{1}{q})$-learnable by a majority-separating oracle for $t(n) = \infty$, $s(n) \leq \sigma(n) \leq c(n) \leq q(n) \cdot \log|\mathcal{C}_n|$.*

Proof We describe the required learner $\mathcal{L}$. $\mathcal{L}^{C,\mathbb{M}}$ works iteratively; starting from the entire collection $\mathcal{K}_0 = \mathcal{C}_n$, it each time reduces the current set of candidates $\mathcal{K}_i$ to a strict sub-collection $\mathcal{K}_{i+1} \subsetneq \mathcal{K}_i$, until it finds a circuit $\hat{C}$ computing the same function as C, or C itself. Specifically, as long as $\mathcal{K}_i$ contains some x that separates at least a $\frac{1}{q}$-fraction of the circuits in $\mathcal{K}_i$ from $\text{maj}_{\mathcal{K}_i}$, $\mathcal{L}$ queries C on x and defines $\mathcal{K}_{i+1}$ to be the subset of all circuits in $\mathcal{K}_i$ that agree with C on x. If no such x exists then $\mathcal{K}_i$ is $\frac{1}{q}$-concentrated, in which case $\mathcal{L}$ asks the majority-separating oracle $\mathbb{M}$ for a point x that separates C from $\text{maj}_{\mathcal{K}_i}$. If $\mathbb{M}$ returns $\perp$, then $C \equiv \text{maj}_{\mathcal{K}_i}$ is returned. Otherwise, $\mathcal{L}$ queries C on x and continues as before.

It is left to note that, with each query to C of the first type, $\mathcal{K}_i$ is reduced by a factor of $(1 - 1/q(n))$, and with each query of the second type it is reduced by a factor of $1/q(n) < 1 - 1/q(n)$, and thus:

$$|\mathcal{K}_i| \leq (1 - 1/q(n))^i \, |\mathcal{K}_0| \leq 2^{-i/q(n)}|\mathcal{C}_n| \leq |\mathcal{C}_n|^{\left(1-\frac{i}{q(n)\cdot\log|\mathcal{C}_n|}\right)}.$$

This implies that $\mathcal{L}$ learns some $\hat{C} \in \mathcal{C}_n$ of equivalent functionality to C after at most $q(n) \cdot \log|\mathcal{C}_n|$ iterations, and thus $s(n) \leq \sigma \leq c \leq q(n) \cdot \log|\mathcal{C}_n|$. □

4.3.2 VBB Obfuscation for Sets of Constant Size

A k-set circuit C_S is associated with a set S of k points, it accepts all points in S, and rejects all other points. A special well-studied case of set circuits is that of point circuits, where $k = 1$.

Definition 4.3 (*Set circuits*) For a set $S \subseteq \{0,1\}^n$ of size k, the set circuit C_S returns 1 for any $x \in S$, and 0 for all $x \notin S$. $\mathcal{S}^k = \bigcup_{n\in\mathbb{N}} \mathcal{S}^k_n$, where $\mathcal{S}^k_n = \{C_S : S \subseteq \{0,1\}^n, |S| = k(n)\}$, is the collection of k-set circuits.

Theorem 4.3 *Let $\mathcal{O}$ be a strong indistinguishability obfuscator for $\mathcal{S}^k$. Then $\mathcal{O}$ is also a worst-case VGB obfuscator for $\mathcal{S}^k$, with a simulator of size $n^{O(k)}$, and polynomially many queries. In particular, for $k = O(1)$ it is also a VBB obfuscator.*

As for Theorem 4.2, Theorem 4.3 is proven by showing how to learn set circuits via a majority-separating oracle with certain efficiency parameters, and plugging it in Theorem 4.1.

Lemma 4.3 *$\mathcal{S}^k$ is $(t, \sigma, c, s, \varepsilon)$-learnable by a majority-separating oracle for $t(n) = \text{poly}(n)$, $\sigma(n) = c(n) = 0$, $s(n) = k(n)$, and $\varepsilon(n) = 2^{-\Omega(n)}$.*

Proof We describe the required learner $\mathcal{L}$. $\mathcal{L}^{C_S,\mathbb{M}}$ works iteratively, revealing the points in the set S one by one, as follows. Having already revealed a subset $T \subsetneq S$, $\mathcal{L}$ queries $\mathbb{M}$ on the sub-collection $\mathcal{K}_T$ corresponding to all the set circuits $C_{S'}$ such that $T \subseteq S'$ and $|S'| = k$, where $\mathcal{K}_T$ is represented via a poly(n)-size circuit that samples a random element in $\mathcal{K}_T$.

Note that each such sub-collection $\mathcal{K}_T$ is $2^{-\Omega(n)}$-concentrated, since for a point $x \in T$ all circuits in $\mathcal{K}_T$ return 1, whereas for $x \notin T$, all but $2^{-\Omega(n)}$-fraction of the circuits in $\mathcal{K}_T$ return 0. Eventually, after at most $k(n)$ queries to $\mathbb{M}$, the entire set S is revealed. □

4.3.3 VBB Obfuscation for Linear Subspaces over Finite Fields

Let $\mathbb{F} = \{\mathbb{F}_n\}_{n\in\mathbb{N}}$ be a sequence of finite fields such that $\mathbb{F}_n$ is of size $2^{\Theta(n)}$ and representing elements and computing field operations can be done efficiently as a function of n. A subspace circuit tests whether a given $x \in \mathbb{F}^d$ is a member of some linear subspace $\mathcal{V} \subseteq \mathbb{F}^d$, or equivalently whether x belongs to the kernel of some given matrix $A \in \mathbb{F}^{d\times d}$.

Definition 4.4 (*Subspace circuits*) Let $d(n)$ be a polynomially bounded function. For a matrix $A \in \mathbb{F}^{d(n)\times d(n)}$ let C_A be a circuit that returns 1 if and only if $x \in \ker(A)$. Let $\mathcal{V}^{d,\mathbb{F}} = \bigcup_{n\in\mathbb{N}} \mathcal{L}_n^{d,\mathbb{F}}$, where $\mathcal{V}_n^{d,\mathbb{F}} = \left\{C_A : A \in \mathbb{F}_n^{d(n)\times d(n)}\right\}$ is the collection of subspace circuits.

A special case of subspace circuit obfuscation, studied by [17], is that of hyperplane circuits where $\text{rank}(A) = 1$. They show how to VBB obfuscate hyperplanes for dimension $d = O(1)$, under a strong variant of Decision Diffie Hellman.

We prove the following theorem.

Theorem 4.4 *Let $\mathcal{O}$ be a strong indistinguishability obfuscator for $\mathcal{V}^{d,\mathbb{F}}$. Then $\mathcal{O}$ is also a worst-case VGB obfuscator for $\mathcal{V}^{d,\mathbb{F}}$, with a simulator of size $n^{O(d)}$, and polynomially many queries. In particular, for $d = O(1)$ it is also a VBB obfuscator.*

As before, the theorem is proven by showing how to learn subspace circuits using a majority-separating oracle with ceratin efficiency features, and plugging it in Theorem 4.1.

Lemma 4.4 *$\mathcal{V}^{d,\mathbb{F}}$ is $(t, \sigma, c, s, \varepsilon)$-learnable by a majority-separating oracle for $t(n) = \text{poly}(n)$, $\sigma(n) = c(n) = 0$, $s(n) \le d(n)$, and $\varepsilon(n) = 2^{-\Omega(n)}$.*

Proof We describe the required learner $\mathcal{L}$. Let $d = d(n)$ and let $A \in \mathbb{F}_n^{d\times d}$. $\mathcal{L}^{C_A,\mathbb{M}}$ gradually constructs a basis for $\ker(A)$. Having found a matrix $B \in \mathbb{F}_n^{d\times i}$ of $\text{rank}(B) = i$, for $i < d$, and such that $AB = 0^i$, $\mathcal{L}$ queries the majority separating oracle $\mathbb{M}$ on the sub-collection $\mathcal{K}_B = \left\{C_{A'} : A'B = 0^i\right\}$, where $\mathcal{K}_B$ is represented by a poly(n)-size circuit that samples uniform elements from $\mathcal{K}_B$. The collection $\mathcal{K}_B$ is $2^{-\Omega(n)}$-concentrated; indeed, for any $x \in \text{span}(B)$, $x \in \ker(A')$ and $C_{A'}(x) = 1$ for all $C_{A'} \in \mathcal{K}_B$. For $x \notin B$, as long as $i < d$, $x \in \ker(A')$ with probability at most $|\mathbb{F}_n|^{-1} = 2^{-\Omega(n)}$. Thus, $\mathcal{L}$ obtains a separating vector $b_{i+1} \in \ker(A) \setminus \text{span}(B)$, and

can extend B to $B_{i+1} = (B_i|b_{i+1})$. If for some $i < d$, at the i-th step $\mathbb{M}$ returns $\perp$, it holds that $\ker(A) = \text{span}(B_i)$, and $\mathcal{L}$ outputs some A' such that $\ker(A') = \text{span}(B_i)$. Otherwise $A = 0^{d\times d}$. □

4.3.4 VBB Obfuscation for All-or-Nothing Learnable Circuits

A collection $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ of polynomial-size boolean circuits is said to be all-or-nothing learnable if:

- $\mathcal{C}$ is evasive. That is, given oracle access to a random $C \in \mathcal{C}_n$ is is hard to find a point x such that $C(x) = 1$.
- Given oracle access to any $C \in \mathcal{C}_n$ and given any point x such that $C(x) = 1$ the circuit C can be efficiently learned.

Examples of all-or-nothing learnable collections include point circuits (discussed above), conjunction circuits $C_{T,F}(x) = \bigwedge_{i\in T} x_i \bigwedge_{i\in F} \neg x_i$, for disjoint $T, F \subset [n]$, or Hamming ball circuits $C_{y,d}(x) = 1$ iff $|y - x| \leq d$, for $y \in \{0,1\}^n$ and $d \in [n]$ (these are also known as fuzzy point circuits).[5]

Definition 4.5 (*All-or-nothing learnable circuits*) An evasive circuit collection $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ is said to be all-or-nothing learnable, if there exists a polynomial-time learner $\mathcal{R}$ such that for every $C \in \mathcal{C}_n$ and $x \in C^{-1}(1)$, $\mathcal{R}^C(x) = \hat{C}$, for some $\hat{C} \in \mathcal{C}_n$ that is functionally equivalent to C.

We prove the following theorem.

Theorem 4.5 *Let $\mathcal{O}$ be a strong indistinguishability obfuscator for any collection $\mathcal{C}$ of all-or-nothing learnable circuits. Then $\mathcal{O}$ is also a worst-case VBB obfuscator for $\mathcal{C}$.*

Again, the theorem is proved by showing how to learn am all-or-nothing learnable circuit collection using a majority-separating oracle with ceratin efficiency features, and plugging it in Theorem 4.1.

Lemma 4.5 *Any collection $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ of all-or-nothing learnable circuits is $(t, \sigma, c, s, \varepsilon)$-learnable by a majority-separating oracle for $t(n) = \text{poly}(n)$, $s(n) = 1$, $\sigma(n) = 0$, $c(n) = \text{poly}(n)$, and $\varepsilon(n) = n^{-\omega(1)}$.*

Proof We describe the required learner $\mathcal{L}$. $\mathcal{L}^{C_S,\mathbb{M}}$ queries the majority-separating $\mathbb{M}$ once, on the entire collection $\mathcal{C}_n$ (which we assume to have an efficient circuit that samples uniform elements in $\mathcal{C}_n$), obtains a point x such that $C(x) = 1$, and uses the learner $\mathcal{R}$, given by Definition 4.5, to learn $\hat{C} \in \mathcal{C}_n$ with equivalent functionality to C. Recall that indeed, since $\mathcal{C}$ is evasive, $\text{maj}_{\mathcal{C}_n} \equiv 0$ and $\mathcal{C}_n$ is $\mu(n)$-concentrated for some $\mu(n) = n^{-\omega(1)}$. In particular, no calls are made to C before the query to $\mathbb{M}$, meaning that $\sigma(n) = 1$. All the c calls made to C, are made after the call to $\mathbb{M}$, when applying the learner $\mathcal{R}$, and thus $c(n) = \text{poly}(n)$ □

[5] Indeed, the first two examples are also evasive collections. The Hamming ball collection, for a given d, is evasive up to a certain threshold $d^* \in [n]$, and beyond that threshold, every function in the collection is exactly learnable.

4.3.5 Remarks

Having presented the above results, a few technical remarks are in place.

Remark 4.2 Barak et al. [3] show that if there exists an average-case VBB obfuscator for every evasive function collection, then for every collection of polynomial-size circuits, there exists a *weak* average-case VGB obfuscation, where the simulator is allowed some super-polynomial number of oracle queries. Their result can also be scaled down to speak of all collections in NC^1. The VGB obfuscators constructed here are stronger in two aspects: first they are worst-case, rather than average-case, and second, the obfuscation simulator is only allowed a polynomial number of queries.

Remark 4.3 (Non-boolean functions) Our results are stated for boolean functions. For our result on VGB obfuscation for all circuits, this is without loss of generality, since for non-boolean circuit $C(x)$, we can obfuscate the boolean circuit $C'(x, i) = C_i(x)$ that returns the i-th output bit, given additional input i. As for our VBB results on restricted classes, such as set circuits, subspace circuits, or all-or-nothing learnable circuits, our results can be rather directly generalized to also allow a given multi-bit output. Namely, the image of any circuit is still boolean, but rather than $\{0, 1\}$ it consists of $\{0, s\}$, for any given string s. These type of multi-bit output circuits were previously studied in [4,12,17], and proven useful for strong forms of encryption.

Remark 4.4 (Auxiliary input) The worst-case VBB and VGB definitions considered here allow a *non-universal simulator* [5]. In particular, the simulator is allowed to have non-uniform advice that arbitrarily (and inefficiently) depends on the adversary's non-uniform advice. As noted in [4], in the case of VGB this is without loss of generality. However, for VBB, universal simulation does not follow from non-universal simulation, and we do not know how to extend our results to this setting.

5 SIO is Equivalent to VBB Obfuscation for Concentrated Distributions

In this section we show that SIO for a given collection $\mathcal{C}$ is not only equivalent to VGB for $\mathcal{C}$, but is also equivalent to requiring average-case VBB for any concentrated distribution on $\mathcal{C}$.

Theorem 5.1 *Let $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ be a circuit collection, and let $\mathcal{O}$ be an obfuscation algorithm for $\mathcal{C}$. Then the following two conditions are equivalent:*

1. *$\mathcal{O}$ is a strong indistinguishability obfuscator for $\mathcal{C}$.*
2. *For any concentrated sub-collection $\mathcal{B} = \bigcup_{n\in\mathbb{N}} \mathcal{B}_n \subseteq \mathcal{C}$, $\mathcal{O}$ is average-case VBB for $\mathcal{B}$.*

Before proving the theorem, we first prove the following useful lemma regarding an alternative definition for average-case obfuscation for concentrated distributions. The lemma, implicitly proven in [3] for the special case of evasive distributions, shows that, for concentrated distributions, (average-case) VBB obfuscation admits a universal simulator that essentially runs the adversary on an obfuscation of a random circuit.

Lemma 5.1 *Let $\tilde{\mathcal{C}} = \bigcup_{n\in\mathbb{N}} \tilde{\mathcal{C}}_n$ be a concentrated distribution ensemble on a circuit collection $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$. Then $\mathcal{O}$ is an average-case VBB obfuscator for $\tilde{\mathcal{C}}$ if and only if for any polynomial-size $\mathcal{A}$ there exists a negligible $\mu(\cdot)$, such that for any $n \in \mathbb{N}$, and any predicate $\pi : \mathcal{C}_n \to \{0,1\}$,*

$$\left| \Pr_{C\leftarrow\tilde{\mathcal{C}}_n} [\mathcal{A}(\mathcal{O}(C)) = \pi(C)] - \Pr_{C,C'\leftarrow\tilde{\mathcal{C}}_n} \left[\mathcal{A}(\mathcal{O}(C)) = \pi(C')\right] \right| \leq \mu(n). \tag{1}$$

Proof For the first direction, assume that $\mathcal{O}$ is an average-case VBB obfuscator for $\tilde{\mathcal{C}}$. Fix any polynomial-size $\mathcal{A}$. Fix any polynomial $p(\cdot)$, and let $\mathcal{S}$ be an average-case VBB simulator for $(\mathcal{A}, p)$ according to Definitions 2.2,2.4. Then, for any $n \in \mathbb{N}$ and predicate $\pi : \mathcal{C}_n \to \{0,1\}$,

$$\begin{aligned}
&\left| \Pr_{C\leftarrow\tilde{\mathcal{C}}_n} [\mathcal{A}(\mathcal{O}(C)) = \pi(C)] - \Pr_{C,C'\leftarrow\tilde{\mathcal{C}}_n} \left[\mathcal{A}(\mathcal{O}(C')) = \pi(C)\right] \right| \\
&\leq \left| \Pr_{C\leftarrow\tilde{\mathcal{C}}_n} [\mathcal{A}(\mathcal{O}(C)) = \pi(C)] - \Pr_{C\leftarrow\tilde{\mathcal{C}}_n} \left[S^C(1^n) = \pi(C)\right] \right| \\
&\quad + \left| \Pr_{C,C'\leftarrow\tilde{\mathcal{C}}_n} \left[\mathcal{A}(\mathcal{O}(C)) = \pi(C')\right] - \Pr_{C,C'\leftarrow\tilde{\mathcal{C}}_n} \left[S^C(1^n) = \pi(C')\right] \right| \\
&\quad + \left| \Pr_{C\leftarrow\tilde{\mathcal{C}}_n} \left[S^C(1^n) = \pi(C)\right] - \Pr_{C,C'\leftarrow\tilde{\mathcal{C}}_n} \left[S^C(1^n) = \pi(C')\right] \right| \\
&\leq \frac{1}{p(n)} + \frac{1}{p(n)} + 2q(n) \max_{x\in\{0,1\}^n} \Pr_{C\leftarrow\tilde{\mathcal{C}}_n} \left[C(x) \neq \mathrm{maj}_{\tilde{\mathcal{C}}_n}(x)\right] \\
&\leq \frac{2}{p(n)} + 2q(n) \cdot \mu(n),
\end{aligned}$$

In the second inequality, the first two summands are bounded by $1/p(n)$ by the security of $\mathcal{O}$. The third summand is bounded by the probability that in the two executions of S, at least one query has a non-zero answer (otherwise the output of $\mathcal{S}$ is independent of its oracle). This probability is bounded by $2q(n) \cdot \mu(n)$, where $q(n)$ is the polynomial bounding the number of queries made by $\mathcal{S}$, and $\mu(n)$ is the negligible concentration of $\tilde{\mathcal{C}}_n$.

Note that the above holds for every polynomial p, which implies that

$$\left| \Pr_{C\leftarrow\tilde{\mathcal{C}}_n} [\mathcal{A}(\mathcal{O}(C)) = \pi(C)] - \Pr_{C,C'\leftarrow\tilde{\mathcal{C}}_n} \left[\mathcal{A}(\mathcal{O}(C')) = \pi(C)\right] \right| = \mathrm{negl}(n),$$

as desired.

For the second direction, let $\mathcal{S}_n$ be a simulator that outputs 1 with probability

$$p_n \triangleq \Pr_{\mathcal{A},\mathcal{O},C\leftarrow\tilde{\mathcal{C}}_n} [\mathcal{A}(\mathcal{O}(C)) = 1],$$

where p_n is non-uniformly hardwired into $\mathcal{S}_n$. By Equation (1), $\mathcal{S}$ is a valid simulator for $\tilde{\mathcal{C}}$. □

Proof of Theorem 5.1 We first prove that (2) implies (1). Specifically, assume that (1) does not hold, we show that (2) also does not hold. If (1) does not hold then there exist two concentrated ensembles $\tilde{\mathcal{C}}^0, \tilde{\mathcal{C}}^1$ such that $\text{maj}_{\tilde{\mathcal{C}}_n^0} \equiv \text{maj}_{\tilde{\mathcal{C}}_n^1}$, a polynomial-size adversary $\mathcal{A}$, and a noticeable function δ such that, for infinitely many $n \in \mathbb{N}^* \subseteq \mathbb{N}$,

$$\Pr_{C \leftarrow \tilde{\mathcal{C}}_n^0}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{C}}_n^1}[\mathcal{A}(\mathcal{O}(C)) = 1] \geq \delta(n)$$

(the absolute value is discarded WLOG, by flipping $\mathcal{A}$'s output if necessary).

For any circuit $C \in \mathcal{C}_n$, let $p(C) = \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C)) = 1]$, and for a distribution $\tilde{D}$ on $\mathcal{C}_n$, let $p(\tilde{D}) = \mathbb{E}_{C \leftarrow \tilde{D}}[p(C)]$. Then, for any $n \in \mathbb{N}^*$,

$$p(\tilde{\mathcal{C}}_n^0) - p(\tilde{\mathcal{C}}_n^1) \geq \delta(n).$$

Next, denote

$$S_n^0 = \left\{C : p(C) \geq p(\tilde{\mathcal{C}}_n^0) - \delta(n)/4\right\}$$
$$S_n^1 = \left\{C : p(C) \leq p(\tilde{\mathcal{C}}_n^1) + \delta(n)/4\right\}.$$

Note that

$$\begin{aligned}
p(\tilde{\mathcal{C}}_n^0) &\leq \Pr_{C \leftarrow \tilde{\mathcal{C}}_n^0}[C \in S_n^0] + p(\tilde{\mathcal{C}}_n^0) - \delta(n)/4 \\
&\Rightarrow \boxed{\Pr_{C \leftarrow \tilde{\mathcal{C}}_n^0}[C \in S_n^0] \geq \delta(n)/4}, \\
p(\tilde{\mathcal{C}}_n^1) &\geq \Pr_{C \leftarrow \tilde{\mathcal{C}}_n^1}[C \notin S_n^1] \cdot \left(p(\tilde{\mathcal{C}}_n^1) + \delta(n)/4\right) \\
&\Rightarrow \boxed{\Pr_{C \leftarrow \tilde{\mathcal{C}}_n^1}[C \in S_n^1] \geq 1 - \frac{p(\tilde{\mathcal{C}}_n^1)}{p(\tilde{\mathcal{C}}_n^1) + \delta(n)/4} \geq \frac{\delta(n)/4}{p(\tilde{\mathcal{C}}_n^0)} \geq \delta(n)/4}.
\end{aligned}$$

We next consider the following two distributions conditioned on the above events

$$\tilde{\mathcal{D}}_n^b := \tilde{\mathcal{C}}_n^b | S_n^b \text{, for } b \in \{0, 1\}.$$

Then

$$\begin{aligned}
p(\tilde{\mathcal{D}}_n^0) - p(\tilde{\mathcal{D}}_n^1) \geq \\
(p(\tilde{\mathcal{C}}_n^0) - \delta(n)/4) - (p(\tilde{\mathcal{C}}_n^1) + \delta(n)/4) \geq \\
p(\tilde{\mathcal{C}}_n^0) - p(\tilde{\mathcal{C}}_n^1) - \delta(n)/2 \geq \delta(n)/2.
\end{aligned}$$

We now consider the distribution $\tilde{\mathcal{D}}_n = \frac{\tilde{\mathcal{D}}_n^0 + \tilde{\mathcal{D}}_n^1}{2}$ that samples from $\tilde{D}_n^b$ for a uniform $b \in \{0, 1\}$. We first claim that the corresponding ensemble $\tilde{\mathcal{D}} = \bigcup_{n \in \mathbb{N}^*} \tilde{\mathcal{D}}_n$ is concentrated. Indeed, since each $\tilde{\mathcal{D}}_n^b$ is distributed like $\tilde{\mathcal{C}}_n^b$, conditioned on S_n^b, and since S_n^b has noticeable density $\delta(n)/4$, it holds that $\tilde{\mathcal{D}}_n^b$ is concentrated around $\mathrm{maj}_{\tilde{\mathcal{C}}_n^b}$. Thus,

$$\mathrm{maj}_{\tilde{\mathcal{D}}_n^0} \equiv \mathrm{maj}_{\tilde{\mathcal{C}}_n^0} \equiv \mathrm{maj}_{\tilde{\mathcal{C}}_n^1} \equiv \mathrm{maj}_{\tilde{\mathcal{D}}_n^1},$$

and since $\mathcal{D}$ is the average of $\mathcal{D}_n^0, \mathcal{D}_n^1$, it is also concentrated and

$$\mathrm{maj}_{\tilde{\mathcal{D}}_n} \equiv \mathrm{maj}_{\tilde{\mathcal{D}}_n^0} \equiv \mathrm{maj}_{\tilde{\mathcal{D}}_n^1}.$$

Next, define a predicate π_n on the support of $\mathcal{D}_n$ such that $\pi_n(C) = b$ if and only if $C \in S_n^b$. Then, it holds that

$$\begin{aligned}
&\Pr_{C,C' \leftarrow \tilde{\mathcal{D}}_n}\left[\mathcal{A}(\mathcal{O}(C')) = \pi_n(C)\right] - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}\left[\mathcal{A}(\mathcal{O}(C)) = \pi_n(C)\right] \\
&= \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\pi_n(C) = 0] \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\mathcal{A}(\mathcal{O}(C)) = 0] \\
&\quad + \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\pi_n(C) = 1] \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] \\
&\quad - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\pi_n(C) = 1] \Pr_{C \leftarrow \tilde{\mathcal{D}}_n^1}[\mathcal{A}(\mathcal{O}(C)) = 1] \\
&\quad - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\pi_n(C) = 0] \Pr_{C \leftarrow \tilde{\mathcal{D}}_n^0}[\mathcal{A}(\mathcal{O}(C)) = 0] \\
&= \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\pi_n(C) = 0]\left(\Pr_{C \leftarrow \tilde{\mathcal{D}}_n^0}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1]\right) \\
&\quad + \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\pi_n(C) = 1]\left(\Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n^1}[\mathcal{A}(\mathcal{O}(C)) = 1]\right) \\
&= \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[C \in S_n^0]\left(\Pr_{C \leftarrow \tilde{\mathcal{D}}_n^0}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1]\right) \\
&\quad + \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[C \in S_n^1]\left(\Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n^1}[\mathcal{A}(\mathcal{O}(C)) = 1]\right) \\
&= \frac{1}{2}\left(\Pr_{C \leftarrow \tilde{\mathcal{D}}_n^0}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1]\right) \\
&\quad + \frac{1}{2}\left(\Pr_{C \leftarrow \tilde{\mathcal{D}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n^1}[\mathcal{A}(\mathcal{O}(C)) = 1]\right) \\
&= \frac{1}{2}\left(\Pr_{C \leftarrow \tilde{\mathcal{D}}_n^0}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{D}}_n^1}[\mathcal{A}(\mathcal{O}(C)) = 1]\right)
\end{aligned}$$

$$= \frac{1}{2}\left(p(\tilde{\mathcal{D}}_n^0) - p(\tilde{\mathcal{D}}_n^1)\right)$$
$$\geq \delta(n)/4.$$

By Lemma 5.1, this contradicts the fact that $\mathcal{O}$ is average-case VBB for the concentrated ensemble $\tilde{\mathcal{D}}$.

We next prove that (1) implies (2). Fix any concentrated ensemble $\tilde{\mathcal{C}}$ on the collection $\mathcal{C}$, and assume that (2) does not hold, we show that (1) also does not hold. By Lemma 5.1, if (1) does not hold, there exists a polynomial-size $\mathcal{A}$ and noticeable $\delta(\cdot)$ such that for infinitely many $n \in \mathbb{N}^* \subseteq \mathbb{N}$ and predicates $\pi_n : \mathcal{C}_n \to \{0,1\}$, it holds that

$$\delta(n) \leq \left| \Pr_{C \leftarrow \tilde{\mathcal{C}}_n}[\mathcal{A}(\mathcal{O}(C)) = \pi_n(C)] - \Pr_{C,C' \leftarrow \tilde{\mathcal{C}}_n}\left[\mathcal{A}(\mathcal{O}(C')) = \pi_n(C)\right] \right|$$
$$\leq \Pr_{C \leftarrow \tilde{\mathcal{C}}_n}[\pi_n(C) = 0] \left| \Pr_{C \leftarrow \tilde{\mathcal{C}}_n : \pi_n(C)=0}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{C}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] \right|$$
$$+ \Pr_{C \leftarrow \tilde{\mathcal{C}}_n}[\pi_n(C) = 1] \left| \Pr_{C \leftarrow \tilde{\mathcal{C}}_n : \pi_n(C)=1}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{C}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] \right|.$$

Then, for infinitely many $n \in \mathbb{N}^*$, one of the two above summands is at least $\delta(n)/2$, let us assume WLOG that it is the first (the proof is similar in the second case). Now consider the distribution $\tilde{\mathcal{C}}_n^0 = \left\{C \in \tilde{\mathcal{C}}_n : \pi_n(C) = 0\right\}$. Since $\Pr_{C \leftarrow \tilde{\mathcal{C}}_n}[\pi_n(C) = 0] \geq \frac{\delta(n)}{2}$, it holds that $\tilde{\mathcal{C}}_n^0$, like $\tilde{\mathcal{C}}_n$ is concentrated around $\mathrm{maj}_{\tilde{\mathcal{C}}_n}$. Moreover,

$$\delta(n)/2 \leq \left| \Pr_{C \leftarrow \mathcal{C}_n^0}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{C \leftarrow \tilde{\mathcal{C}}_n}[\mathcal{A}(\mathcal{O}(C)) = 1] \right|,$$

implying that strong indistinguishability, required in (2), does not hold. □

5.1 Equivalence of VBB Obfuscation for Concentrated and Evasive Distributions

We complete this section by showing that indistinguishability obfuscation, plus (average-case) VBB obfuscation for evasive distributions implies (average-case) VBB obfuscation for *concentrated* distributions.

We start by noting the following fact.

Claim 5.1 *Let $\tilde{S}$ be a $\frac{1}{3}$-concentrated distribution over boolean circuits where each boolean circuit $C \in \mathrm{supp}(\tilde{S})$ is defined over $\{0,1\}^n$ and is of depth at most d and size at most ℓ. Then the majority function $\mathrm{maj}_{\tilde{S}}$ can be computed by a (non-uniform) circuit of size $O(n \cdot \ell)$ and depth $O(\log n + d)$. Also, if $\tilde{S}$ is samplable by a circuit of size s, such a majority circuit can be sampled, with overwhelming probability $1 - 2^{-\Omega(n)}$, by a circuit of size $O(n \cdot s)$.*

Proof Since $\tilde{S}$ is $\frac{1}{3}$-concentrated, by a Chernoff bound and a union bound on 2^n inputs, the majority of $O(n)$ random circuits from $\tilde{S}$ computes $\mathrm{maj}_{\tilde{S}}$ with probability $1 - 2^{-\Omega(n)}$. □

We next state the equivalence lemma. For a circuit C of size at most ℓ, we denote by $[C]_\ell$ a canonically zero-padded version of C of size ℓ. For a collection $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$, and functions $\ell(\cdot), d(\cdot)$, we denote by $[\mathcal{C}]_\ell^d$ the class of circuits where each $C \in \mathcal{C}$ computes the same function as some $C \in \mathcal{C}_n$ and is of size $\ell(n)$, and depth $d(n)$. Let M_n be a polynomial-size circuit computing the majority $\mathrm{maj}_{\tilde{\mathcal{C}}_n}$ over $\tilde{\mathcal{C}}_n$. We denote by $\tilde{\mathcal{C}} \oplus M_n$ the distribution ensemble $\bigcup_{n\in\mathbb{N}} \left\{C \oplus M_n : C \leftarrow \tilde{\mathcal{C}}_n\right\}$. Observe that $\tilde{\mathcal{C}} \oplus M_n$ is evasive.

Lemma 5.2 *Let $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$ be a circuit collection where each $C \in \mathcal{C}_n$ is of polynomial-size $\ell(n)$ and depth at most d, and let $\tilde{\mathcal{C}}$ be a concentrated distribution ensemble on $\mathcal{C}$. For any $n \in \mathbb{N}$ let M_n be a circuit of size $O(\ell(n) \cdot n)$ and depth $O(\log n + d(n))$ that computes $\mathrm{maj}_{\tilde{\mathcal{C}}}$. Assume that there exists an average-case VBB obfuscator for the evasive distribution ensemble $\tilde{\mathcal{C}} \oplus M_n$ that blows up the size of any circuit C by some polynomial $B(\cdot)$. Then there exist a polynomial $\ell'(n)$ and a function $d'(n) = O(d(n) + \log n)$, depending only on (ℓ, B, d'), such that if $i\mathcal{O}$ is an indistinguishability obfuscator for $[\mathcal{C}]_{\ell'}^{d'}$, then the obfuscator $c\mathcal{O}$, given by*

$$c\mathcal{O}(C) \leftarrow i\mathcal{O}([C]_{\ell'(n)})$$

is an average-case VBB obfuscator for $\tilde{\mathcal{C}}$.

Proof Let M_n be the circuit of size $O(\ell(n) \cdot n)$ that computes $\mathrm{maj}_{\tilde{\mathcal{C}}_n}$. Let $e\mathcal{O}$ be an average-case VBB obfuscator for the evasive distribution ensemble $\tilde{\mathcal{C}} \oplus M_n$ such that $|e\mathcal{O}(C)| = B(|C|)$. For $C \in \mathcal{C}_n$, consider the circuit

$$C_r := M_n \oplus e\mathcal{O}(C \oplus M_n; r)),$$

which computes the same function as C, but in a different way—it has hardwired an obfuscation $e\mathcal{O}(C \oplus M_n; r)$, using some randomness r, that computes its difference from the majority; it first runs this obfuscation on the given input, and then computes again the difference from the majority, resulting back in $C(x)$. We let $\ell'(n)$ be the size of C_r and $d'(n)$ be its depth. Note that these, indeed, only depend on ℓ, b, d and $d'(n) = O(\mathsf{depth}(C) + \mathsf{depth}(M_n)) = O(d(n) + \log n)$.

Next, applying first the IO guarantee and then the guarantee that $e\mathcal{O}$ is an average-case VBB obfuscator for the evasive ensemble $\bigcup_{n\in\mathbb{N}} M_n \oplus \tilde{\mathcal{C}}_n$, as given by Lemma 5.1, we have

$$
\begin{aligned}
&\Pr_{\mathcal{A},c\mathcal{O},C\leftarrow\tilde{\mathcal{C}}_n}[\mathcal{A}(c\mathcal{O}(C))=\pi(C)]\\
&=\Pr_{\mathcal{A},i\mathcal{O},C\leftarrow\tilde{\mathcal{C}}_n}\left[\mathcal{A}(i\mathcal{O}([C]_{\ell'(n)}))=\pi(C)\right]\\
&=\Pr_{\mathcal{A},i\mathcal{O},r,C\leftarrow\tilde{\mathcal{C}}_n}[\mathcal{A}(i\mathcal{O}(C_r))=\pi(C)]\pm n^{-\omega(1)}\\
&=\Pr_{\mathcal{A},i\mathcal{O},r,C\leftarrow\tilde{\mathcal{C}}_n}[\mathcal{A}(i\mathcal{O}(M_n\oplus e\mathcal{O}(C\oplus M_n;r)))=\pi(C)]\pm n^{-\omega(1)}\\
&=\Pr_{\mathcal{A},i\mathcal{O},r,C,C'\leftarrow\tilde{\mathcal{C}}_n}\left[\mathcal{A}(i\mathcal{O}(M_n\oplus e\mathcal{O}(C\oplus M_n;r)))=\pi(C')\right]\pm n^{-\omega(1)}\\
&=\Pr_{\mathcal{A},i\mathcal{O},r,C,C'\leftarrow\tilde{\mathcal{C}}_n}\left[\mathcal{A}(i\mathcal{O}(C_r))=\pi(C')\right]\pm n^{-\omega(1)}\\
&=\Pr_{\mathcal{A},i\mathcal{O},C,C'\leftarrow\tilde{\mathcal{C}}_n}\left[\mathcal{A}(i\mathcal{O}([C]_{\ell'(n)}))=\pi(C')\right]\pm n^{-\omega(1)}\\
&=\Pr_{\mathcal{A},c\mathcal{O},C,C'\leftarrow\tilde{\mathcal{C}}_n}\left[\mathcal{A}(c\mathcal{O}(C))=\pi(C')\right]\pm n^{-\omega(1)}.
\end{aligned}
$$

Thus, again by Lemma 5.1, we deduce that $c\mathcal{O}$ is an average-case VBB obfuscator for the concentrated ensemble $\tilde{\mathcal{C}}$. □

6 From Semantically Secure Multilinear Jigsaw Puzzles to SIO for NC^1

The obfuscation scheme of [19] is based on the notion of *multilinear jigsaw puzzles* that captures a restriction of the multilinear maps functionality. In this section, we define semantically-secure multilinear jigsaw puzzles, which are a variant of the Pass et al. [25] definition. We show that *any* obfuscation scheme for a class $\mathcal{C}$ of circuits that is virtual-black-box secure in the ideal multilinear jigsaw puzzle model against semi-bounded adversaries, together with semantically-secure multilinear jigsaw puzzles, implies *strong indistinguishability obfuscation* for $\mathcal{C}$. Combined with the recent ideal-model obfuscators for NC^1 [1,8,9,28], we obtain SIO for NC^1. We also show that VGB obfuscation for all polynomial-size circuits implies semantically-secure multilinear jigsaw puzzles.

6.1 Multilinear Jigsaw Puzzles

Multilinear jigsaw puzzles were introduced by Garg et al. [19] to capture the restricted graded encoding functionality (described in Sect. 1.2 of the introduction) used in their obfuscation construction. In graded encoding schemes, elements are encoded one by one, and any encoding can be combined with any other encoding in a homomorphic computation. In contrast, multilinear jigsaw puzzles encode, once and for all, a sequence of elements (with their respective control sets). The puzzle supports the same homomorphic operations as graded encodings, but only over the elements encoded in the puzzle. There is no way to compute on elements encoded in different puzzles.

Once the sequence of encoded elements is fixed and puzzles is initialized, all the evaluator can do is homomorphically evaluate some arithmetic expression over these encodings and test if the polynomial evaluates to zero or not. In contrast to graded encodings, in multilinear jigsaw puzzles, the homomorphic operations and zero-test operation are not performed one by one on individual encodings, and the evaluator does not obtain encoding of intermediate steps of the homomorphic computation. Instead, the puzzle evaluator specifies a *solution* in the form of an arithmetic circuit, and tests if it "solves" the puzzle. If the arithmetic circuit respects the control sets the the evaluator learns if evaluation of the circuit on the encoded elements is zero or not. Formally, a multilinear jigsaw puzzle scheme is a given by a tuple of PPT algorithms

$$(\mathsf{KeyGen}, \mathsf{PuzzleGen}, \mathsf{Test}) ,$$

with the following syntax

- KeyGen is a randomized algorithm generating a secret key for the scheme. It takes is input the security parameter 1^n and a bound $B \in \mathbb{N}$ on the solution size. It outputs a secret key sk including the description of a ring R.
- $\mathsf{PuzzleGen}$ is a randomized algorithm for generating puzzles. It takes as input the secret key sk, a level $k \in \mathbb{N}$, a vector of sets $\vec{S} \in (2^{[k]})^{\ell}$ and a vector of elements $\vec{m} \in R^{\ell}$. It outputs a puzzle Z.
- Test is a deterministic algorithm for testing solutions to puzzles. It takes as input a puzzle Z and an arithmetic circuit C of size at most B. It output a bit.

Next, we recall the definition of set-respecting arithmetic circuits from [25]. An arithmetic circuit *respects* a sequence of control sets if it is possible to test if the circuit evaluates to zero on a sequence of encodings with the given control sets via the supported graded encoding operations.

Definition 6.1 (*Set-respecting arithmetic circuits* [25]) Given a level $k \in \mathbb{N}$, and a vector of sets $\vec{S} \in (2^{[k]})^{\ell}$, we say that an arithmetic circuit C, taking ℓ inputs, is $\vec{S}$-respecting if there exists a function Tag from the wires of C to $2^{[k]}$ such that the following holds:

- For every $i \in [\ell]$, the i-th input wire w_{in}^{i} satisfies $\mathsf{Tag}(w_{in}^{i}) = \vec{S}[i]$.
- Every $+$ or $-$ gate in C connecting input wires u and v to an output wire w, satisfies $\mathsf{Tag}(u) = \mathsf{Tag}(v) = \mathsf{Tag}(w)$.
- Every $\times$ gate in C connecting input wires u and v to output wire w, satisfies $\mathsf{Tag}(u) \cap \mathsf{Tag}(v) = \emptyset$ and $\mathsf{Tag}(u) \cup \mathsf{Tag}(v) = \mathsf{Tag}(w)$.
- The output wire w_{out} satisfies $\mathsf{Tag}(w_{out}) = [k]$.

Next, we define the functionality of multilinear jigsaw puzzles.

Definition 6.2 (*Multilinear jigsaw puzzle functionality*) A multilinear jigsaw puzzle scheme $(\mathsf{KeyGen}, \mathsf{PuzzleGen}, \mathsf{Test})$ satisfies the following functionality requirement. For every security parameter $n \in \mathbb{N}$, every size bound $B \in \mathbb{N}$, every secret key sk in the support of $\mathsf{KeyGen}(1^n, B)$ including the description of a ring R, every level $k \in \mathbb{N}$, every a vector of sets $\vec{S} \in (2^{[k]})^{\ell}$, every vector of elements $\vec{m} \in R$ and every arithmetic circuit C of size at most B the following holds.

- If C is not $\vec{S}$-respecting then

$$\Pr[\mathsf{Test}(\mathsf{PuzzleGen}(\mathsf{sk}, k, \vec{S}, \vec{m}), C) = \perp] = 1.$$

- If C is $\vec{S}$-respecting and $C(\vec{m}) = 0$

$$\Pr[\mathsf{Test}(\mathsf{PuzzleGen}(\mathsf{sk}, k, \vec{S}, \vec{m}), C) = 1] = 1.$$

- If C is $\vec{S}$-respecting and $C(\vec{m}) \neq 0$

$$\Pr[\mathsf{Test}(\mathsf{PuzzleGen}(\mathsf{sk}, k, \vec{S}, \vec{m}), C) = 0] = 1.$$

Garg et al. [19] suggested a simple construction of multilinear jigsaw puzzles based on graded encodings. Their puzzle contains the public parameters of the graded encodings scheme and individual encodings of the puzzle's elements.

6.2 Semantic Security

We now define semantically-secure multilinear jigsaw puzzles. The definition follows that of Pass et al. [25] for graded encodings. (Indeed, semantically-secure multilinear jigsaw puzzles will be a restriction that is implied by semantically-secure graded encodings.) We then formulate a variant of semantic security with inefficient massage samplers (See Remark 6.2). Although it appears to be somewhat stronger than the notion considered by Pass et al., we find it natural and appealing. See [6] for a number of relaxations of this basic notion and their relative security.
We first define the ideal oracle implementing the multilinear jigsaw puzzle functionality.

Definition 6.3 (*Ideal multilinear jigsaw puzzle oracle* $\mathcal{M}$ ([25])) For a ring R, size bound $B \in \mathbb{N}$, a level $k \in \mathbb{N}$, a vector of sets $\vec{S} \in (2^{[k]})^{\ell}$ and a vector of elements $\vec{m} \in R^{\ell}$, the oracle $\mathcal{M}(R, B, k, \vec{S}, \vec{m})$ is defined as follows: for every query C, if $|C| > B$ or if C is not a description of an $\vec{S}$-respecting arithmetic circuit, $\mathcal{M}$ outputs $\perp$. Otherwise, $\mathcal{M}$ evaluates C on $\vec{m}$ and outputs 1 if C evaluates to 0, and outputs 0 otherwise.

Central to the definition of semantic security is the notion of an *admissible message sampler* (analogous to *respecting message samplers* in [25]). A message sampler is admissible if it samples two vectors of elements $\vec{m}_0, \vec{m}_1$ that cannot be distinguished by a semi-bounded adversary that has unbounded in size, but can only access $\vec{m}_0$ or $\vec{m}_1$ by making a polynomial number of queries to an ideal multilinear jigsaw puzzle oracle $\mathcal{M}$.

Definition 6.4 (*Efficient message sampler*) Let $(\mathsf{KeyGen}, \mathsf{PuzzleGen}, \mathsf{Test})$ be a multilinear jigsaw puzzle scheme and let $k = k(n)$, $\ell = \ell(n)$, and $B = B(n)$ be polynomials. A PPT algorithm $\mathbb{M}$ is an efficient (k, ℓ, B)-message sampler if for every $n \in \mathbb{N}$, and every secret key sk in the support of $\mathsf{KeyGen}(1^n, B)$ including the

description of a ring R, $\mathbb{M}(1^n, R)$ samples a vector of sets $\vec{S} \in (2^{[k]})^\ell$ and two vectors of elements $\vec{m}_0, \vec{m}_1 \in R^\ell$. We require that the vector of sets $\vec{S}$ depends only on the security parameter n an not on randomness of $\mathbb{M}$.

Remark 6.1 In the above definition, we do not allow $\mathbb{M}$ to choose the vector of sets $\vec{S}$ depending on its randomness or on the circuit C. This restriction simplifies the presentation of our results. However, all our results hold also with respect to the more general definition of message samplers where $\vec{S}$ can be sampled from some distribution.

Definition 6.5 (*Admissible message sampler*) Let (KeyGen, PuzzleGen, Test) be a multilinear jigsaw puzzle scheme and let $k = k(n)$, $\ell = \ell(n)$, and $B = B(n)$ be polynomials. A (k, ℓ, B)-message sampler $\mathbb{M}$ (Definition 6.4) is admissible if for every polynomial q and for every (unbounded) oracle machine $\mathcal{A}$ (called the ideal adversary) making at most $q(n)$ oracle queries, where every query describes an arithmetic circuit of size at most $B(n)$, there exists a negligible function μ such that for every $n \in \mathbb{N}$ and every secret key sk in the support of $\mathsf{KeyGen}(1^n, B)$ including the description of a ring R,

$$\Pr\left[\begin{array}{c} b \leftarrow \{0,1\} \\ (\vec{S}, \vec{m}_0, \vec{m}_1) \leftarrow \mathbb{M}(1^n, R) \end{array} ; \mathcal{A}^{\mathcal{M}(R,B,k,\vec{S},\vec{m}_b)}(n) = b\right] \leq \frac{1}{2} + \mu(n),$$

where the probability is also over the coins of $\mathcal{A}$.

Loosely speaking, a multilinear jigsaw puzzle scheme is semantically secure if for a tuple $(\vec{S}, \vec{m}_0, \vec{m}_1)$ generated by an admissible message sampler, given a puzzle encoding the elements $\vec{m}_b$ with the sets $\vec{S}$, it is hard to predict b with non-negligible advantage in polynomial time.

Remark 6.2 (Inefficient message samplers) For most of our results, we need to rely on a stronger notion of semantic security that allows for computationally unbounded admissible message samplers. Since the message sampler $\mathbb{M}$ in Definition 6.4 takes as input the description of a ring R associated with the secret key of multilinear jigsaw puzzle, simply allowing $\mathbb{M}$ to be unbounded may result in an unachievable definition. Specifically, consider a multilinear jigsaw puzzle scheme where it is possible to recover the secret key sk from the description of the ring R in unbounded time.[6] An inefficient $\mathbb{M}$ may recover sk and sample elements that reveal it. $\mathbb{M}$ may still be admissible since knowing the secret key gives no advantage to the ideal adversary, however given sk it may be possible to distinguish the sampled puzzles (for any non-trivial $\mathbb{M}$).

Instead, in Definition 6.6 we keep the sampler $\mathbb{M}$ efficient, but we give it auxiliary input that is sampled by an inefficient algorithm. Importantly, we do not give the auxiliary-input sampler the description of the ring.

Definition 6.6 (*Inefficient message sampler*) Let (KeyGen, PuzzleGen, Test) be a multilinear jigsaw puzzle scheme and let $k = k(n)$, $\ell = \ell(n)$, and $B = B(n)$ be polynomials. An unbounded (k, ℓ, B)-message sampler is defined by a PPT algorithm $\mathbb{M}$

[6] We note that this is not the case for existing candidate construction [15,18].

and an unbounded auxiliary input sampler $\mathbb{Z}$. We require that there exist a polynomial q such that for every $n \in \mathbb{N}$, and every secret key sk in the support of $\mathsf{KeyGen}(1^n, B)$ including the description of a ring R, $|\mathbb{Z}(n)| < q(n)$, and $\mathbb{M}(1^n, \mathbb{Z}(n), R)$ outputs a vector of sets $\vec{S} \in (2^{[k]})^\ell$ and two vectors of elements $\vec{m}_0, \vec{m}_1 \in R^\ell$. We require that the vector of sets $\vec{S}$ depends only on the security parameter n and not on randomness of $\mathbb{M}$ or $\mathbb{Z}$.

The definition of admissability remains as in Definition 6.5. In what follows, we only consider semantic security with respect to inefficient admissible message samplers (unless we explicitly state otherwise).

Definition 6.7 (*Semantically-secure multilinear jigsaw puzzles*) A multilinear jigsaw puzzle scheme (KeyGen, $\mathsf{PuzzleGen}$, Test) is semantically secure if for every polynomials $k = k(n)$, $\ell = \ell(n)$, $B = B(n)$, every inefficient admissible (k, ℓ, B)-message sampler $(\mathbb{Z}, \mathbb{M})$, and every polynomial-size adversary $\mathcal{A}$, there exist a negligible function μ such that for every $n \in \mathbb{N}$,

$$\Pr\left[\begin{array}{c} b \leftarrow \{0,1\} \\ \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^n, B) \\ (\vec{S}, \vec{m}_0, \vec{m}_1) \leftarrow \mathbb{M}(1^n, \mathbb{Z}(n), R) \end{array} ; \mathcal{A}\left(\mathsf{PuzzleGen}(\mathsf{sk}, k, \vec{S}, \vec{m}_b)\right) = b\right] \leq \frac{1}{2} + \mu(n),$$

where R is described in the secret key sk and the probability is also over the coins of $\mathcal{A}$.

6.3 Ideal Multilinear Jigsaw Puzzle Obfuscation

Barak et al. [8] construct a virtual black-box obfuscator for NC^1 in the ideal multilinear jigsaw puzzle model where algorithms have access to an ideal oracle $\mathcal{M}$ (See Definition 6.3). Next, we define virtual-black-box obfuscation in this model. Here an obfuscation of a circuit consists of a vector of sets $\vec{S}$, and a vector of elements $\vec{m}$ used to initialize a multilinear jigsaw puzzle oracle $\mathcal{M}$. The honest evaluator and the attacker have oracle access to $\mathcal{M}$.

Definition 6.8 (*Ideal multilinear jigsaw puzzle obfuscation*) Let (KeyGen, $\mathsf{PuzzleGen}$, Test) be multilinear jigsaw puzzle scheme. A PPT algorithm $\mathcal{O}$ is a virtual black-box (VBB) obfuscator in the ideal multilinear jigsaw puzzle model, for a family of polynomial-size circuits $\mathcal{C} = \{C_n\}_{n\in\mathbb{N}}$, if it satisfies the following requirements:

- Functionality: There exist polynomials $k = k(n)$, $\ell = \ell(n)$ and $B = B(n)$ and there exists a polynomial-time oracle machine $\mathsf{Eval}_{\mathcal{O}}$ that satisfy the following: for every $n \in \mathbb{N}$, every secret key sk in the support of $\mathsf{KeyGen}(1^n, B)$ including the description of a ring R, and for every $C \in \mathcal{C}_n$, the obfuscator $\mathcal{O}(1^n, R, C)$ outputs a vector of sets $\vec{S} \in (2^{[k]})^\ell$ and a vector of elements $\vec{m} \in R^\ell$ such that for every input x to C:

$$\mathsf{Eval}_{\mathcal{O}}^{\mathcal{M}(R,B,k,\vec{S},\vec{m})}(x) = C(x).$$

We require that the vector of sets $\vec{S}$ depends only on the security parameter n an not on the circuit C or the randomness of $\mathcal{O}$.

- Virtual black-box: For every polynomial function q and for every (unbounded) oracle machine $\mathcal{A}$ (called the ideal adversary) making at most $q(n)$ queries, where every query describes an arithmetic circuit of size at most $B(n)$, there exist a PPT oracle machine $\mathcal{S}$ (called the simulator), and negligible function μ, such that for every $n \in \mathbb{N}$, every secret key sk in the support of $\mathsf{KeyGen}(1^n, B)$ including the description of a ring R, and for every $C \in \mathcal{C}_n$:

$$\left| \begin{array}{l} \Pr[(\vec{S}, \vec{m}) \leftarrow \mathcal{O}(1^n, R, C); \mathcal{A}^{\mathcal{M}(R,B,k,\vec{S},\vec{m})}(n) = 1] - \\ \Pr[\mathcal{S}^{\mathcal{A},C}(1^n) = 1] \end{array} \right| \leq \mu(n),$$

where the probabilities are over the coins of the obfuscator $\mathcal{O}$, the adversary $\mathcal{A}$ and the simulator $\mathcal{S}$. The notation $\mathcal{S}^{\mathcal{A}}$ means that $\mathcal{S}$ gets oracle access to $\mathcal{A}$, when answering $\mathcal{A}$'s oracle queries.

Remark 6.3 In the above definition we do not allow $\mathcal{O}$ to choose the vector of sets $\vec{S}$ depending on its randomness or on the circuit C. This restriction simplifies the presentation of our results. Specifically, we rely on the fact that obfuscation of different circuits of the same size use the same vector $\vec{S}$. However, all our results hold also with respect to the more general definition of ideal multilinear jigsaw puzzle obfuscation where the choice of $\vec{S}$ is not restricted. We note that all existing constructions of ideal obfuscation [8,9,25] meet the functionality requirement in its restricted form.

Remark 6.4 (Ideal obfuscation in [8].) The syntax for the obfuscator defined above is different from the one in [8], and is adapted to the way that ideal multilinear jigsaw puzzles are dealt with in this work. A more essential difference is that Barak et al. state their final result for bounded ideal adversaries, whereas we consider semi-bounded ones. Against bounded ideal adversaries, Barak et al. achieve ideal obfuscation for all polynomial-size circuits. For semi-bounded ideal adversaries considered in this work, their result only holds for NC^1.

Theorem 6.1 ([8]) *There exists an ideal multilinear jigsaw puzzle obfuscation for every circuit family in NC^1.*

6.4 Obfuscation from Semantically-Secure Multilinear Jigsaw Puzzles

We show that semantically-secure multilinear jigsaw puzzles imply SIO, and, as a corollary of our result from Sect. 4, also other forms of obfuscation.

Proposition 6.1 *Assume there exists a semantically-secure multilinear jigsaw puzzle scheme (Definition 6.7), and assume there exists an ideal multilinear jigsaw puzzle obfuscation (Definition 6.8) for a circuit class $\mathcal{C}$. Then there exists a strong indistinguishability obfuscator for the circuit class $\mathcal{C}$, in the plain model (Definition 3.2).*

As a corollary of Theorem 6.1, Proposition 6.1, and the transformation from (standard) IO for NC^1 to (standard) IO for all polynomial-size circuit classes [19], we obtain the following theorem.

Theorem 6.2 *Assume there exists a semantically secure multilinear jigsaw puzzle scheme. Then there exist:*

1. *SIO, for any circuit class in* NC^1,
2. *(standard) IO, for any polynomial-size circuit class, assuming also fully-homomorphic encryption with decryption in* NC^1.

As a corollary of the above theorem and of our results from Sect. 4, we obtain the following theorem.

Theorem 6.3 *Assume there exists a semantically-secure multilinear jigsaw puzzle scheme. Then there exist:*

1. *worst-case VGB for any collection in* NC^1,
2. *worst-case VGB obfuscation for the class of set circuits* $\mathcal{S}^k$ *for any* $k = \mathrm{poly}(n)$, *and VBB obfuscation for* $k = O(1)$,
3. *worst-case VGB obfuscation for the class of linear subspaces* $\mathcal{V}^{d,\mathbb{F}}$ *for any* $d = \mathrm{poly}(n)$, *and VBB obfuscation for* $d = O(1)$,
4. *worst-case VBB for any efficiently samplable collection of all-or-nothing learnable circuits in* NC^1, *in particular, for Hamming balls and conjunctions.*

We now turn to give a proof sketch of Proposition 6.1.

Proof of Proposition 6.1 Let $\mathcal{C}$ be a class of polynomial-size circuits, let $(\mathsf{KeyGen}, \mathsf{PuzzleGen}, \mathsf{Test})$ be a semantically secure multilinear jigsaw puzzle scheme. Let $\mathcal{O}$ be an ideal multilinear jigsaw puzzle obfuscator for $\mathcal{C}$.

The obfuscator $\tilde{\mathcal{O}}$ for $\mathcal{C}$: Let $\ell = \ell(n)$, $k = k(n)$ and $B = B(n)$ be the polynomials given by the ideal multilinear jigsaw puzzle obfuscator $\mathcal{O}$ (Definition 6.8). Given $C \in \mathcal{C}_n$, $\tilde{\mathcal{O}}$ samples a secret key $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^n, B)$ for the multilinear jigsaw puzzle scheme, including the description of a ring R. $\tilde{\mathcal{O}}$ then runs $\mathcal{O}(1^n, R, C)$ which outputs a vector of sets $\vec{S} \in (2^{[k]})^\ell$, and a vector of elements $\vec{m} \in R^\ell$, such that for every input x, $\mathsf{Eval}_{\mathcal{O}}^{\mathcal{M}(R,B,k,\vec{S},\vec{m})}(x) = C(x)$. Finally, $\tilde{\mathcal{O}}$ outputs an obfuscated circuit that contains, hardcoded into it, a puzzle

$$Z = \mathsf{PuzzleGen}(\mathsf{sk}, k, \vec{S}, \vec{m}).$$

The obfuscated circuit emulates the evaluation procedure of the ideal multilinear jigsaw puzzle obfuscation $\mathsf{Eval}_{\mathcal{O}}$, where any query C to the ideal oracle $\mathcal{M}$ is answered by $\mathsf{Test}(Z, C)$.

Functionality and indistinguishability The functionality of $\tilde{\mathcal{O}}$ follows readily from that of the multilinear jigsaw puzzle scheme and of $\mathcal{O}$. We now argue strong indistinguishability based on the semantic security of the multilinear jigsaw puzzle scheme. Let $\tilde{\mathcal{C}}^0, \tilde{\mathcal{C}}^1$ be two concentrated distribution ensembles on $\mathcal{C}$ such that $\mathrm{maj}_{\tilde{\mathcal{C}}_n^0} \equiv \mathrm{maj}_{\tilde{\mathcal{C}}_n^1}$, and let $\mathcal{D}$ be any polynomial-size distinguisher. We show that $\mathcal{D}$ cannot distinguish

whether it is given an obfuscation $\tilde{\mathcal{O}}(C_0)$ or $\tilde{\mathcal{O}}(C_1)$, for $(C_0, C_1) \leftarrow (\tilde{\mathcal{C}}_n^0, \tilde{\mathcal{C}}_n^1)$, with non-negligible advantage.

Assume towards contradiction that $\mathcal{D}$ can predict whether it is given $\tilde{\mathcal{O}}(C_0)$ or $\tilde{\mathcal{O}}(C_1)$ with a noticeable advantage over $\frac{1}{2}$. We construct an inefficient admissible (k, ℓ, B)-message sampler $(\mathbb{Z}, \mathbb{M})$, and show that together with $\mathcal{D}$ they violate the semantic security of the multilinear jigsaw puzzle scheme. The auxiliary input sampler $\mathbb{Z}$ samples $(C_0, C_1) \leftarrow (\tilde{\mathcal{C}}_n^0, \tilde{\mathcal{C}}_n^1)$ (note that if $\tilde{\mathcal{C}}_n^0, \tilde{\mathcal{C}}_n^1$ are not efficiently samplable, $\mathbb{Z}$ is inefficient). The message sampler $\mathbb{M}$, given a ring R, and the auxiliary input (C_0, C_1) executes the ideal multilinear jigsaw puzzle obfuscator $\mathcal{O}(1^n, R, \cdot)$ on each of the circuits, and obtains a vector of sets $\vec{S}$ and two vectors of messages $\vec{m}_0, \vec{m}_1$, which it then outputs. (Here we use the fact that $\mathcal{O}$ outputs the same the vector of sets $\vec{S}$ in both executions. See Remark 6.3.)

To argue that the sampler $(\mathbb{Z}, \mathbb{M})$ is admissible, we need to show that the following holds for every (unbounded) oracle machine $\mathcal{A}$ (the ideal adversary) making only polynomially many oracle queries, where every query describes an arithmetic circuit of size at most $B(n)$, for every $n \in \mathbb{N}$, and for every secret key sk in the support of $\mathsf{KeyGen}(1^n, B)$ including the description of a ring R,

$$\Pr\left[\begin{array}{c} b \leftarrow \{0,1\} \\ (\vec{S}, \vec{m}_0, \vec{m}_1) \leftarrow \mathbb{M}(1^n, \mathbb{Z}(n), R) \end{array}; \mathcal{A}^{\mathcal{M}(R,B,k,\vec{S},\vec{m}_b)}(n) = b\right] \leq \frac{1}{2} + \mathrm{negl}(n),$$

where the probability is also over the coins of $\mathcal{A}$.

This follows from the ideal virtual-black-box security of $\mathcal{O}$. Indeed, for any ideal adversary $\mathcal{A}$ as above, let $q(n)$ be the polynomial bound on its number of queries, and let $\mathcal{S}$ be its virtual-black-box simulator (according to Definition 6.8). Therefore,

$$\begin{aligned} &\Pr\left[\begin{array}{c} b \leftarrow \{0,1\} \\ (\vec{S}, \vec{m}_0, \vec{m}_1) \leftarrow \mathbb{M}(1^n, \mathbb{Z}(n), R) \end{array}; \mathcal{A}^{\mathcal{M}(R,B,k,\vec{S},\vec{m}_b)}(n) = b\right] \\ &\leq \Pr\left[b \leftarrow \{0,1\}; (C_0, C_1) \leftarrow (\tilde{\mathcal{C}}_n^0, \tilde{\mathcal{C}}_n^1); \mathcal{S}^{\mathcal{A}, C_b}(1^n) = b\right] + \mathrm{negl}(n) \\ &\leq \frac{1}{2} + q(n) \cdot \nu(n) + \mathrm{negl}(n) = \frac{1}{2} + \mathrm{negl}(n), \end{aligned}$$

where $\nu(n) = \max(\nu_0(n), \nu_1(n))$, and $\nu_b(n) = \mathrm{negl}(n)$ is the negligible concentration measure of $\mathcal{C}_n^b$ around $\mathrm{maj}_{\tilde{\mathcal{C}}_n^0} \equiv \mathrm{maj}_{\tilde{\mathcal{C}}_n^1}$.

It is left to note that, by the construction of $(\mathbb{Z}, \mathbb{M})$ and the assumption that $\mathcal{D}$ predicts b with noticeable advantage (given $\tilde{\mathcal{O}}(C_b)$ for a random b), $\mathcal{D}$ breaks the semantic security of the multilinear jigsaw puzzle scheme (Definition 6.7). □

6.5 Semantically-Secure Multilinear Jigsaw Puzzles from VGB Obfuscation

We show that VGB obfuscation for all polynomial-size circuits imply semantically secure multilinear jigsaw puzzles.

Theorem 6.4 *Assume there exists worst-case VGB obfuscation for the class of all polynomial-size circuits. Then there exists a semantically secure multilinear jigsaw puzzle scheme.*

Proof Let $\mathcal{O}$ be a worst-case VGB obfuscation for the class of all polynomial-size circuits. We construct a semantically secure multilinear jigsaw puzzle scheme (KeyGen, $\mathsf{PuzzleGen}$, Test) as follows.

Given the security parameter 1^n and a bound B, the key generation algorithm KeyGen output the secret key $\mathsf{sk} = (1^n, B, R)$, where R is a ring. (Our construction puts no restriction on the way R is chosen.)

Given the secret key sk, a level $k \in \mathbb{N}$, a vector of sets $\vec{S} \in (2^{[k]})^\ell$ and a vector of elements $\vec{m} \in R^\ell$, the puzzle generation algorithm $\mathsf{PuzzleGen}$ is defined as follows. Let $Z_{R,B,k,\vec{S},\vec{m}}$ be a circuit that takes as input a description of an arithmetic circuit C of size B (if $|C| < B$ the circuit is padded) and answers the similarly to the ideal oracle $\mathcal{M}(R, B, k, \vec{S}, \vec{m})$. $\mathsf{PuzzleGen}$ VGB obfuscates the circuit

$$\tilde{Z} = \mathcal{O}(Z_{R,B,k,\vec{S},\vec{m}}),$$

and outputs the obfuscated circuit $\tilde{Z}$ as the puzzle.

Given a puzzle $\tilde{Z}$, and an arithmetic circuit C of size at most B, the solution testing algorithm Test executes the circuit $\tilde{Z}$ on the description of C and outputs the result.

Functionality and semantic security The functionality of the multilinear jigsaw puzzle scheme follows readily from that of the $\mathcal{O}$. We now argue that the construction satisfies semantic security.

Let $k = k(n), \ell = \ell(n), B = B(n)$ be polynomial and let $(\mathbb{Z}, \mathbb{M})$ be an inefficient admissible (k, ℓ, B)-message sampler. Assume thete exists a polynomial-size adversary $\mathcal{A}$, and a polynomial p such that for infinitely many values of $n \in \mathbb{N}$

$$\Pr\left[\begin{array}{c} b \leftarrow \{0,1\} \\ (1^n, B, R) \leftarrow \mathsf{KeyGen}(1^n, B) \\ (\vec{S}, \vec{m}_0, \vec{m}_1) \leftarrow \mathbb{M}(1^n, \mathbb{Z}(n), R) \end{array} ; \mathcal{A}\left(\mathcal{O}(Z_{R,B,k,\vec{S},\vec{m}_b})\right) = b\right] \geq \frac{1}{2} + \frac{1}{p(n)},$$

Let $\pi_{R,B,k,\vec{S},\vec{m}_0,\vec{m}_0}$ be a predicate that outputs b given the circuit $Z_{R,B,k,\vec{S},\vec{m}_b}$. By the VGB security of $\mathcal{O}$ there exists a polynomial q and an unbounded simulator $\mathcal{S}$ making at most $q(n)$ queries, each describing an arithmetic circuit of size at most $B(n)$ such that for every n as above, for every sk in the support of $\mathsf{KeyGen}(1^n, B)$ including the description of a ring R, for every vectors $(\vec{S}, \vec{m}_0, \vec{m}_1)$ in the support of $\mathbb{M}(1^n, \mathbb{Z}(n), R)$ and for every bit $b \in \{0, 1\}$,

$$\left|\Pr_{\mathcal{A},\mathcal{O}}\left[\mathcal{A}\left(\mathcal{O}(Z_{R,B,k,\vec{S},\vec{m}_b})\right) = b\right] - \Pr_{\mathcal{S}}[\mathcal{S}^{\mathcal{M}(R,B,k,\vec{S},\vec{m}_b)}(1^n) = b]\right| \leq \frac{1}{2p(n)}.$$

We therefore have that for infinitely many values of $n \in \mathbb{N}$ there exists a secret key sk in the support of support of $\mathsf{KeyGen}(1^n, B)$ including the description of a ring R such that

$$\Pr\left[\begin{array}{c} b \leftarrow \{0,1\} \\ (\vec{S}, \vec{m}_0, \vec{m}_1) \leftarrow \mathbb{M}(1^n, \mathbb{Z}(n), R) \end{array} ; \mathcal{S}^{\mathcal{M}(R,B,k,\vec{S},\vec{m}_b)}(1^n) = b\right] \geq \frac{1}{2} + \frac{1}{2p(n)}.$$

This contradicts the admissibility of the inefficient message sampler $(\mathbb{Z}, \mathbb{M})$. □

Acknowledgements We are grateful to Rafael Pass for enlightening discussions and valuable comments. We also thank Vincenzo Iovino for carefully reading our manuscript and for providing useful comments.

References

1. Applebaum, B., Brakerski, Z.: Obfuscating circuits via composite-order graded encoding. In: TCC (2015)
2. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. Cryptology ePrint Archive, Report 2015/173. http://eprint.iacr.org/ (2015)
3. Barak, B., Bitansky, N., Canetti, R., Kalai, Y.T., Paneth, O., Sahai, A.: Obfuscation for evasive functions. In: TCC, pp. 26–51 (2014)
4. Bitansky, N., Canetti, R.: On strong simulation and composable point obfuscation. In: CRYPTO, pp. 520–537 (2010)
5. Bitansky, N., Canetti, R., Cohn, H., Goldwasser, S., Kalai, Y.T., Paneth, O., Rosen, A.: The impossibility of obfuscation with auxiliary input or a universal simulator. CoRR, abs/1401.0348, (2014)
6. Bitansky, N., Canetti, R., Kalai, Y.T., Paneth, O.: On virtual grey box obfuscation for general circuits. In: IACR Cryptology ePrint Archive, p. 554 (2014). The EPRINT version of our work
7. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: CRYPTO, pp. 1–18 (2001)
8. Barak, B., Garg, S., Kalai, Y.T., Paneth, O., Sahai, A.: Protecting obfuscation against algebraic attacks. Cryptology ePrint Archive, Report 2013/631. http://eprint.iacr.org/ (2013)
9. Brakerski, Z., Rothblum, G.N.: Virtual black-box obfuscation for all circuits via generic graded encoding. Cryptology ePrint Archive, Report 2013/563. http://eprint.iacr.org/ (2013)
10. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. Cryptology ePrint Archive, Report 2015/163. http://eprint.iacr.org/ (2015)
11. Canetti, R.: Towards realizing random oracles: hash functions that hide all partial information. In: CRYPTO, pp. 455–469 (1997)
12. Canetti, R., Dakdouk, R.R.: Obfuscating point functions with multibit output. In: Proceedings of Advances in Cryptology—EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp 489–508. Istanbul, 13–17 Apr 2008
13. Coron, J., Gentry, C., Halevi, S., de Lepoint, T., Maji, H.K., Miles, E., Raykova, M., Sahai, A., Tibouchi, M.: Zeroizing without low-level zeroes: new MMAP attacks and their limitations. In: Proceedings of Advances in Cryptology—CRYPTO 2015—35th Annual Cryptology Conference, Part I, pp. 247–266. Santa Barbara, 16–20 Aug 2015
14. Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé, D.: Cryptanalysis of the multilinear map over the integers. In: Proceedings of Advances in Cryptology—EUROCRYPT 2015—34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Part I, pp. 3–12. Sofia, Bulgaria, 26–30 Apr 2015
15. Coron, J.S., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. CRYPTO **1**, 476–493 (2013)
16. Coron, J.S., de Lepoint, T., Tibouchi, M.: New multilinear maps over the integers. In: CRYPTO (2015)
17. Canetti, R., Rothblum, G.N., Varia, M.: Obfuscation of hyperplane membership. In: TCC, pp. 72–89 (2010)
18. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: EUROCRYPT, pp. 1–17(2013)
19. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: FOCS (2013)
20. Goldwasser, S., Kalai, Y.T.: On the impossibility of obfuscation with auxiliary input. In: FOCS, pp. 553–562 (2005)

21. Gentry, C., Lewko, A., Sahai, A., Waters, B.: Indistinguishability obfuscation from the multilinear subgroup elimination assumption. Cryptology ePrint Archive, Report 2014/309. http://eprint.iacr.org/ (2014)
22. Goldwasser, S., Rothblum, G.N.: On best-possible obfuscation. In: TCC, pp. 194–213 (2007)
23. Hada, S.: Zero-knowledge and code obfuscation. In: ASIACRYPT, pp. 443–457 (2000)
24. Hu, Y., Jia, H.: Cryptanalysis of GGH map. In: Proceedings of Advances in Cryptology—EUROCRYPT 2016—35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Part I, pp. 537–565. Vienna, 8–12 May 2016
25. Pass, R., Telang, S., Seth, K.: Obfuscation from semantically-secure multi-linear encodings. Cryptology ePrint Archive, Report 2013/781. http://eprint.iacr.org/ (2013)
26. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. IACR Cryptol ePrint Arch **2013**, 454 (2013)
27. Wee, H.: On obfuscating point functions. IACR Cryptol ePrint Arch **2005**, 1 (2005)
28. Zimmerman, J.: How to obfuscate programs directly. In: Eurocrypt (2015)