Electronic Colloquium on Computational Complexity, Report No. 40 (2016)

Affine Relativization: Unifying the Algebrization and Relativization Barriers

Barış Aydınlıoğlu [*] Eric Bach [†]

March 15, 2016

Abstract

We strengthen existing evidence for the so-called "algebrization barrier". Algebrization — short for algebraic relativization — was introduced by Aaronson and Wigderson (AW) in order to characterize proofs involving arithmetization, simulation, and other "current techniques". However, unlike relativization, eligible statements under this notion do not seem to have basic closure properties, making it conceivable to take two proofs, both with algebrizing conclusions, and combine them to get a proof without. Further, the notion is undefined for most types of statements, and does not seem to yield a general criterion by which we can tell, given a proof, whether it algebrizes. In fact the very notion of an algebrizing proof is never made explicit, and casual attempts to define it are problematic. All these issues raise the question of what evidence, if any, is obtained by knowing whether some statement does or does not algebrize.

We reformulate algebrization to handle these shortcomings. We first define a statement as *relativizing* if, intuitively, it is insensitive to the choice of a Boolean basis, and then as *relativizing affinely* if, roughly, it relativizes with respect to every affine extension — here an affine extension is the result of a particular error correcting code applied to the characteristic string of a language. We also define the notion of a *proof* to relativize (affinely), while ensuring closure under inference. We show that all statements that AW declare as algebrizing can be derived via an affinely relativizing proof, and that no such proof exists for any of the statements shown not-algebrizing by AW in the classical computation model.

Our work complements, and goes beyond, the subsequent work by Impagliazzo, Kabanets, and Kolokolova (IKK), which also proposes a reformulation of algebrization, but falls short of recovering some key results of AW, most notably regarding the NEXP versus $\mathrm{P/poly}$ question.

One consequence of our definitions is a demystified perspective on the extent to which relativizing techniques view computation as a "black box" and current uses of arithmetization do not. As a bonus, we give new streamlined proofs of $\mathrm{PSPACE} \subset \mathrm{IP}$ and $\mathrm{NEXP} \subset \mathrm{MIP}$.

[*] University of Wisconsin-Madison; baris@cs.wisc.edu.
[†] University of Wisconsin-Madison; bach@cs.wisc.edu

ISSN 1433-8092

Contents

1 Introduction

The algebrization notion — short for algebraic relativization — was put forth by Aaronson and Wigderson [1] (AW henceforth) to give evidence that certain complexity-theoretic conjectures are beyond the reach of "current proof techniques". Although the name suggests some type of relativization, algebrization lacks two essential properties of relativization:

Closure under inference. What exactly constitutes a "current technique" may be inherently unclear, but at a minimum it seems logical inference rules should be included. However, as pointed out in [1, 24, 19], statements that algebrize in the AW formulation are not known to be closed under inference.

For example, AW show that the statement $\psi := \mathrm{NEXP} \not\subset \mathrm{P/poly}$ does not algebrize, and interpret this to mean that a certain class of proof techniques, say "algebrizing techniques", cannot prove ψ. Yet, this does not rule out an approach where, say, one comes up with a class $\mathcal{C}$ and, shows $\mathcal{C} \subset \mathrm{NEXP}$ via algebrizing techniques, then shows $\mathcal{C} \not\subset \mathrm{P/poly}$ via algebrizing techniques, and thus derive the very same ψ.

Lack of closure under inference thus significantly thins any evidence imparted by a negative algebrization result — as AW obtained for NEXP versus P/poly and for other questions of structural complexity — since the class of proofs ruled out by such a result might be much smaller than intended.

This precludes algebrization from having one of the two key virtues of relativization, namely delineating those conjectures within possible reach of a robust family of techniques, from those that are not. Indeed, some major results in complexity are suggested to have been found using relativization as such a guide [5, 14].

Universality. A main appeal of relativization is being a universal notion, in the sense that it applies to every statement in one generic way. Intuitively, a statement relativizes if its truth is insensitive to broadening the definition of computer, from an ordinary Turing Machine, to one with oracle access to an arbitrary language $\mathcal{O}$. (We provide an alternate intuition later in Section 1.1.)

This intuition is so natural that it enables the second key virtue of relativization, namely being a "litmus test" for weeding out futile endeavours. The idea is that if ψ is already known to not relativize, then any strategy for proving ψ, in order to be viable, must somehow be unable to handle arbitrary extensions of the computer notion, or else it would be a strategy for proving not just ψ, but that ψ relativizes. Given the scarcity of such proof strategies in structural complexity — at least for those ψ involving P-based classes — this idea makes relativization a practical tool for guiding research. (Alas, we do not have a count on the number of fruitless research hours saved this way.)

For algebrization, however, we have no comparable intuition. This is mainly because algebrization is a selective notion, in the sense that it is defined only for containments $\mathcal{C} \subset \mathcal{D}$ and separations $\mathcal{C} \not\subset \mathcal{D}$, and moreover, it is applied differently to each side of the containment / separation. Supposing we have a strategy to prove ψ — and assuming, to begin with, ψ is of compatible syntax — there is no universal criterion we can apply, to check if our ideas can be extended to show that ψ algebrizes. This calls into question how relevant it is to know that ψ is non-algebrizing in the first place.

Besides the above problems, algebrization brings back some longstanding ones that are as old as the relativization notion itself:

Controversial relativizations. A pair of theorems might be derived using seemingly same techniques, yet only one might be relativizing / algebrizing. For example, $\mathrm{PSPACE} \subset \mathrm{IP}$, as AW show, algebrizes, yet its cousin, $\mathrm{NEXP} \subset \mathrm{MIP}$, does *not*, as observed by Impagliazzo, Kabanets, and Kolokolova [19] — except it *does*, as AW show, if we restrict oracle access for NEXP to be of polynomial-length.

It is not clear how to interpret such results without further work. Can we justify restricting oracle access, say by showing that it yields a natural subclass not tied to the Turing machine model? If so, then which "current technique" eliminates the difference between the two classes, the subclass and the original, thereby overcoming the limits of algebrizing techniques (whatever they are)?

Relativizing statements vs. proofs. A generally accepted (though not uncontested [21]) convention is to remark that some proof, say of ψ, relativizes or algebrizes, with no clear consensus on what that exactly means.

The typical intent behind such remarks seems to be that the said proof can be transformed into a proof *that* ψ *relativizes* (or algebrizes). However, as anything can be transformed into anything when there is no constraint, it is not clear which proofs do *not* relativize under such a definition. And even if some commonsense transformations are tacitly agreed upon — e.g., "give every Turing machine an oracle for $\mathcal{O}$," or "bring each statement to its relativized form" — it is unclear whether the transformed object would always be a valid proof, let alone a valid proof that ψ relativizes.

Although an early manuscript by Arora, Impagliazzo, and Vazirani [3] (AIV) succeeds in giving a precise definition of a relativizing proof, the approach taken there is recursion-theoretic and makes no reference to devices such as Turing machines. Consequently, it seems difficult to tell whether an everyday statement / proof — involving circuits, Turing machines, etc. — relativizes using their formulation. For example, it is not clear if "Satisfiability is NP-complete" relativizes in their framework, or has such a proof, even though "oracle gates" for $\mathcal{O}$ can be easily incorporated into a circuit / formula.

Naturally thus the question arises, of whether a simple definition can be given for what constitutes a relativizing / algebrizing proof, without having to do any hand-tailoring for the computational model being used, circuits versus machines. Ideally, such a definition should also test the folk belief that relativizing techniques use computation as a "black box", and relate that to algebrization.

In this paper, we reformulate relativization and algebrization, in a way that addresses the above problems.

First, we give a simple, combinatorially flavored definition of what it means for a statement / proof to relativize, that yields the following intuition: a statement / proof relativizes if it is insensitive to enlarging the standard Boolean basis. This part of our work can be considered an alternative to the recursion-theoretic approach of Arora, Impagliazzo, and Vazirani [3] mentioned above (see Section 1.2 for a comparison).

Our main contribution is to the algebrization notion. We define a statement / proof as relativizing *affinely* if, intuitively, it is insensitive to enlarging the standard Boolean basis *with any affine extension* — here affine extension is the result of a particular error correcting code applied to the characteristic string of a language. With this definition, we show that every statement that AW declare as relativizing algebraically does relativize affinely — in fact has a *proof* that relativizes affinely — and that the opposite holds for statements declared non-algebrizing by AW in the classical model.[1] (Both require new ideas.) Our formulation in this sense gives rigorous support to the "algebrization barrier" idea of AW, which can thus be viewed as a refinement of the classic "relativization barrier" of Baker, Gill, and Solovay [9].

This part of our work complements, and goes beyond, the prior work by Impagliazzo, Kabanets, and Kolokolova [19] (IKK), which also proposes a relativization-based characterization for algebrization, but falls short of recovering some of the key results of AW, most notably regarding the NEXP versus $\mathrm{P/poly}$ question (see Section 1.2 for details).

Affine relativization is a refinement of relativization so as to capture the known uses of *arithmetization*, a technique for interpolating Boolean formulas into polynomials. Famously used in early 90's for obtaining $\mathrm{PSPACE} \subset \mathrm{IP}$ and related results, which are false relative to some choices of an oracle $\mathcal{O}$ [17, 16, 11],

[1] AW state some non-algebrization results for quantum-based complexity classes as well; we do not pursue these.

arithmetization is widely regarded as a counterexample — maybe *the* counterexample — to the rule-of-thumb that "most known proof techniques relativize" in structural complexity theory. Affine relativization, to the extent that it captures the known uses of arithmetization — and it does so fairly well, as we argue in the rest of this section — can be viewed as a step towards reinstating that rule-of-thumb (albeit only a step, as the PCP theorem is out of scope of this and related work; see open question in Section 5).

As one conceptual consequence, our formulations yield a demystified perspective on the extent to which relativizing techniques are "black-box" and arithmetization-based techniques are not; see Section 5.

Our formulations also tell something about those "few known proof techniques" that do not seem to relativize affinely, in particular, about *locality of computation*. It is a longstanding debate whether locality — that the next step of a computation depends on only a "small" fragment of its current state — plays any role in current results of complexity, particularly in interactive proofs [15, 3, 14, 19]. On one hand, $\mathrm{NEXP} \subset \mathrm{MIP}$ can be explained away as relativizing algebraically with a convenient, but questionable, alteration of the oracle access mechanism as mentioned above; on the other hand, locality could provide an honest explanation of this theorem, as argued by Arora, Impagliazzo, and Vazirani [3], but an incongruent one to its algebraic nature, especially when its cousin, $\mathrm{PSPACE} \subset \mathrm{IP}$, needs no such explanation.

Our results shed some light onto this matter. As we explain in Section 1.3, it is fruitful to put a particular class between PSPACE and IP, and another one between NEXP and MIP, so that each theorem reads as two containments. The second containment, we argue, captures the real content in each theorem, namely "gap amplification"; affine relativization can derive every containment except the first one for NEXP versus MIP. We conclude that whether or not $\mathrm{NEXP} \subset \mathrm{MIP}$ algebrizes is just a matter of definition, because there is no application of this theorem (as far as we know) that is sensitive to how it is viewed, gap amplification versus the common view. Therefore affine relativization can be viewed as a robust proxy, or a candidate thereof, for the current state of the art.

This is mere interpretation, however, and is not to be confused with the main message of the paper:

Summary of Results. *Affinely relativizing proofs, as defined in Section 1.1, have the following properties.*

- *Each of the following has an affinely relativizing proof*
  - $\mathrm{PSPACE} \subset \mathrm{IP}$, *viewed as gap amplification* *(Corollary 15)*
  - $\mathrm{NEXP} \subset \mathrm{MIP}$, *viewed as gap amplification* *(Theorem 17)*
  - $\mathrm{MA_{EXP}} \not\subset \mathrm{SIZE}(2^{\log^d n}), \forall d$ *(Theorem 25)*
  - $\mathrm{prMA} \not\subset \mathrm{SIZE}(n^d), \forall d$ *(Theorem 25)*
  - $\mathrm{NP} \subset \mathrm{ZKIP}$ *if one-way-functions exist* *(Theorem 30)*
- *None of the following has an affinely relativizing proof*
  - $\mathrm{NP} \not\subset \mathrm{P}$, *in fact* $\mathrm{PSPACE} \not\subset \mathrm{P}$ *(Proposition 32)*
  - $\mathrm{NP} \subset \mathrm{P}$, *in fact* $\mathrm{RP} \subset \mathrm{SUBEXP}$ *(Corollary 40)*
  - $\mathrm{NP} \subset \mathrm{BPP}$, *in fact* $\mathrm{coNP} \subset \mathrm{MA}$ *(Corollary 39)*
  - $\mathrm{P}^{\mathrm{NP}} \subset \mathrm{PP}$ *(Corollary 39)*
  - $\mathrm{NEXP} \not\subset \mathrm{P/poly}$, *in fact* $\mathrm{NEXP} \not\subset \mathrm{SIZE}(n^d), \forall d$ *(Theorem 35)*

Further, affinely relativizing proofs are closed under inference, and if a statement has an affinely relativizing proof, then it "affinely relativizes," i.e., it holds relative to each language that is an affine extension, as defined in Section 1.1.

1.1 Relativization and Affine Relativization

We now describe our formulation of the relativization and affine relativization notion. We caution that the notion of an affinely relativizing *statement* does not depend on any peculiarity in the definitions given here. Readers who are already at ease with some notion of relativization (vague though it may be) can skip this section and still understand most of the paper — except what affinely relativizing *proofs* are and why they are closed under inference.

Relativization without oracles. Let the standard Boolean basis refer to the set $\{0, 1, \wedge, \oplus\}$ comprising four languages (with 0 denoting the empty language, viewed as the function mapping all binary strings to zero, 1 its negation, and with $\wedge$,$\oplus$ denoting the AND,XOR function on binary strings respectively). We say that the statement ψ holds relative to the language $\mathcal{O}$ iff ψ is true when the standard Boolean basis is extended with $\mathcal{O}$. We say ψ relativizes to mean that ψ holds relative to every $\mathcal{O}$.

Some remarks are in order.

- Let us momentarily be more precise. Take the axioms of everyday mathematics, say, the Zermelo-Fraenkel set theory. Add two new axioms: (i) that B_{std} equals $\{0, 1, \wedge, \oplus, \mathcal{O}\}$, and (ii) that $\mathcal{O}$ is a language. Here, B_{std} and $\mathcal{O}$ are variables that are not previously used, and $0, 1, \wedge, \oplus$ refer to corresponding languages. Name the new collection of axioms $\mathcal{RCT}$, for relativized complexity theory.

  Now take $\mathcal{RCT}$, and add the axiom: $\mathcal{O}$ is empty (i.e., $\mathcal{O}$ equals the language 0 of B_{std}). Call this set of axioms $\mathcal{CT}$, for real-world complexity theory.

  Our framework of mathematics is $\mathcal{CT}$. Given a statement ψ (in the language of $\mathcal{CT}$, which is the same as that of $\mathcal{RCT}$), we call ψ relativizing iff it is true in every standard model of $\mathcal{RCT}$.[2]
- ψ being nonrelativizing per se does not make it interesting or hard to prove; e.g. let ψ be "the standard Boolean basis is $\{0, 1, \wedge, \oplus\}$".

We agree to take the uniform-circuit-based definition of P, and P-based definitions of NP, NEXP, etc., so that extending the standard Boolean basis with $\mathcal{O}$ automatically gives us the $\mathrm{P}^{\mathcal{O}}$, $\mathrm{NP}^{\mathcal{O}}$, etc., without having to mention oracle access to $\mathcal{O}$ — though we do mention it anyway, for emphasis. An algorithm thus means to us a uniform family of circuits. So if we "let V be a time-$t(n)$ algorithm with oracle access to a proof string π", for example, then we mean to "let $V := \{V_n\}$ be a polynomial-time-uniform circuit family of size $t(n)$ (or of size $t(n)\,\mathrm{polylog}\,t(n)$ — does not matter in this paper) over the basis $B_{std} \cup \pi$" for some language π defined appropriate to the context. The poly-time uniformity can be specified using any notion of classical computer (Turing machines, pointer machines, etc.) or via a recursion-theoretic approach.[3]

It is out of the scope of this paper whether our framework can handle classes "below" P, or those classes not definable from P. (And it is one of the observations of this paper that NEXP with poly-length queries *can* be defined from P, as 0-gap-MIP. Similarly PSPACE can be defined as 0-gap-IP. See Section 1.3.)

ψ relativizes vs. ψ has a proof that relativizes. We call a proof of ψ relativizing iff after extending the standard Boolean basis with an arbitrary language $\mathcal{O}$, it remains a proof of ψ.

Further remarks are in order.

- Let us again be more precise for a moment. Recall that our framework of mathematics is $\mathcal{CT}$. Given a proof Π in $\mathcal{CT}$ — i.e., a sequence $\phi_1..\phi_n$ where ϕ_i either is in $\mathcal{CT}$ or can be inferred from $\phi_1, .., \phi_{i-1}$ — we say that Π relativizes iff it is also a proof in $\mathcal{RCT}$.

[2] Assuming, as we may here and throughout the paper, that everyday mathematics is consistent, a standard model of set theory is one where the symbol for set membership is interpreted as the actual "is an element of" relation.

[3] As mentioned, we do not exploit any peculiarity in this way of defining P. The reader who prefers the Turing machine model can stick to it, provided the machine is defined to have oracle access to every element of the standard Boolean basis.

In other words, *all* proofs in $\mathcal{RCT}$ are relativizing, and no other.

It is immediate that relativizing proofs are closed under inference, since by definition they are exactly those proofs derivable from a certain set of axioms. Also, if a proof relativizes then so does its conclusion, as does each intermediate statement; this is just the soundness theorem from logic.[4]

- A relativizing proof of ψ proves more than ψ. Indeed, such a proof can be viewed as a family of proofs, parameterized by every language $\mathcal{O}$, where trivial settings of $\mathcal{O}$ correspond to proofs of ψ in real-world complexity theory. The family is *uniform*, in fact: one proof works for *every* extension of the standard Boolean basis. One might say the proof treats $\mathcal{O}$ as a "black box".
- If we prove (i.e., if $\mathcal{CT}$ proves) that ψ relativizes, then it does not follow that there is a relativizing proof of ψ, e.g., take ψ to be "Zermelo-Fraenkel set theory is consistent".

  Even if we can, in addition, prove ψ itself — so now we have ψ *and* that ψ relativizes — it is not clear, to us at least, whether a relativizing proof of ψ exists, and unclear still, if furthermore each step in the proof of ψ relativizes. (See open question in Section 5.)

In Section 3, some of the theorems we derive are of the form "ψ, and ψ (affinely) relativizes", or something to that effect. By the above remarks, such a theorem does not, by itself, imply that ψ has a relativizing proof. Nonetheless, in the process of deriving each of these theorems we end up giving a relativizing proof of the corresponding ψ. (Of course we do not provide a formal proof of ψ in first order logic, just as we do not specify algorithms by implementing them as Boolean circuits.)

So those theorems in Section 3 can be read — and should be, by those who have not skipped this section — as "ψ has an (affinely) relativizing proof" for some ψ. To be convinced of such claims, the salient point that should be checked in their proofs is whether they are uniform in the choice of a basis extension, i.e., whether they treat the nonstandard basis elements as (the affine extension of) a black box — and they do.

Affine relativization. Take $\mathcal{RCT}$ and add the axiom: $\mathcal{O}$ is the affine extension of some language. That is, there is a language f, with f_n denoting its restriction to length-n inputs, and with $\widehat{f_n}$ denoting the unique n-variate polynomial of individual degree-≤ 1 extending f_n, such that $\mathcal{O}$ represents the evaluation of $\widehat{f_n}$ over $\mathrm{GF}(2^k)$, for all k and n. (See Section 2 for a precise definition, and Section 1.3 for a discussion.)

Call the resulting set $\mathcal{ACT}$, for affinely relativized complexity theory. Define the notion of a statement / proof affinely relativizing similarly to $\mathcal{RCT}$. It is immediate that affinely relativizing proofs are closed under inference, as they are exactly those proofs derivable from $\mathcal{ACT}$.

The empty language is the affine extension of itself. Thus it does not make any difference to add the same axiom, that $\mathcal{O}$ is an affine extension, to $\mathcal{CT}$ as well. Now we have three theories $\mathcal{RCT} \subset \mathcal{ACT} \subset \mathcal{CT}$, each strictly more powerful than the one before, as the results in this paper imply.

Multiple oracles. If ψ is (affinely) relativizing, or has such a proof, then what happens if the Boolean basis is extended twice — say with $\mathcal{O}_0$ and $\mathcal{O}_1$? For plain relativization the answer is easy; just set $\mathcal{O}$ to be their disjoint union, $\mathcal{O}_0 \coprod \mathcal{O}_1 : bx \mapsto \mathcal{O}_b(x)$, and proceed as before.

For affine relativization, however, a bit more care is needed since we want $\mathcal{O}$ to be an affine extension. If $\mathcal{O}_0$ is the affine extension of L_0, and $\mathcal{O}_1$ of L_1, the key observation is that the disjoint union $\mathcal{O}_0 \coprod \mathcal{O}_1$ of the affine extensions is equivalent, under Cook reductions, to the affine extension of the disjoint union $L_0 \coprod L_1$. This is spelled out in Proposition 38, but intuitively is true because the disjoint union merely adds an extra dimension — the "b-axis" — and the affine extension acts on each dimension independently (see equation (†) on page 9). So we set $\mathcal{O}$ to the affine extension of $L_0 \coprod L_1$ and proceed as before, the upshot being that one affine oracle is just as good as k of them for $k \in \mathbb{N}^+$.

[4] Viewed in the contrapositive, this vindicates the use of casual remarks of the form "this proof does not relativize because this step of it does not" — provided the said step indeed does not relativize, which by itself is a hairy claim in a casual context.

1.2 Comparison with Prior Work

Four past works have a direct relation to ours. The main effort in all of them, and in ours, can be viewed as trying to: (i) formalize the notion of relativizing *proof*, and / or (ii) refine the relativization notion so as to capture $\mathrm{PSPACE} \subset \mathrm{IP}$ and related results. We now do a comparison with past work, first with respect to (i) and then (ii).

1.2.1 Efforts to Formalize Relativization

In an unpublished but well-known manuscript, **Arora, Impagliazzo, Vazirani** [3] (AIV henceforth) build on Cobham's axioms for polynomial-time computation [12] to define $\mathcal{RCT}$, relativized complexity theory, and argue that the standard models of this theory contain, as a sub-model, "relativized P" in the sense of Baker-Gill-Solovay [9], i.e., $\mathrm{P}^{\mathcal{O}}$ for arbitrary $\mathcal{O}$. They then define relativizing proofs as those expressible in $\mathcal{RCT}$.

A common feature — or flaw, if the reader is logically inclined — of both our definition of $\mathcal{RCT}$ and AIV's is that the axioms for capturing relativization go on top of an existing collection of axioms governing everyday mathematics. On one hand, this is a feature because relativization is meant to be a guide for the everyday researcher, who has everyday mathematics at disposal. On the other hand, this is a flaw because statements such as "P versus NP is independent of $\mathcal{RCT}$" can be easily misunderstood, as the so-called independence concerns only *one* natural way of defining P, NP out of at least two — another one being to just ignore the extra axioms (they do not interfere with everyday math). This is inevitable unless one *removes* axioms from mathematics and not add to it, and the quest then becomes to find the "weakest" version of math that can prove statements such as $\mathrm{PSPACE} \subset \mathrm{IP}$, as opposed to finding, like we and AIV essentially set out to do, for the "strongest" version of these statements that can be proven by everyday math.

One difference of our version of $\mathcal{RCT}$ from AIV's is its accessibility. AIV use recursion-theoretic axioms à la Cobham [12] to create a universe of polynomial-time computable functions that goes beside the universe of everyday mathematics. While this approach allows AIV to give an elegant and self-contained axiomatization of P (with only a dozen or so axioms), it also makes cumbersome expressing everyday ideas such as circuits, oracle access to a proof, etc. Our approach, in contrast, is "naive" in the sense that it does not attempt at a minimal set of axioms (nor does it spell out every axiom) but in return, it gives a formalism that is arguably closer to the everyday uses of relativization — e.g., "Satisfiability is NP-complete" is easily seen to relativize in our framework.

1.2.2 Efforts to Refine Relativization

Although relativization succeeds at explaining the failures of structural complexity until the 90s, it fails at explaining the successes after, especially those regarding interactive proofs. We now discuss four past proposals to refine relativization. The overarching goal in these is (or so will be our view here) to provide some model for "known techniques", which involves meeting two competing objectives: (a) derive all relevant theorems in the model, and (b) provably fail to derive in the model all relevant conjectures that are evidently beyond current reach.

We will use Figure 1 to roughly illustrate how each proposal fares with respect to these two objectives (a) and (b). The take-away message from this micro-survey is that although parts of (a), (b) have been attained by prior work, ours is the first successful attempt that yields all the critical pieces under one framework.

Figure 1: Attempts at refining relativization

	$(\exists \mathcal{C}{:}\mathcal{C}\subset \mathrm{NEXP} \wedge \mathcal{C}\not\subset \mathrm{P/poly}) \implies \mathrm{NEXP}\not\subset \mathrm{P/poly}$	$\mathrm{PSPACE}\subset \mathrm{IP}$	PCP thm	$\mathrm{NEXP}\not\subset \mathrm{P/poly}$	$\mathrm{NP}\not\subset \mathrm{P}$, $\mathrm{EXP}\not\subset$i.o.-$\mathrm{P/poly}$,..
AIV	✓	✓	✓	?	?
For	✓	✓	?	?	✓̸
AW	?	✓	?	✓	✓
IKK	✓	✓	?	?	✓
this work	✓	✓	?	✓	✓

Although the table is less precise than the discussion that follows, it does illustrate some key differences among prior work. The vertical line in the table is a caricature of the state of the art; to the left of the line are known theorems / facts, and to the right are conjectures evidently out-of-reach.

The first proposal is from the same paper discussed above, by **AIV** [3]. Besides $\mathcal{RCT}$, there the authors propose "local checkability" as the key non-relativizing ingredient underlying $\mathrm{PSPACE} \subset \mathrm{IP}$ as well as other results including the PCP theorem. The idea is that a polynomial-time computation should be verifiable by inspecting all bits of its transcript in parallel, where each bit depends on only a logarithmic number of bits elsewhere. For computations with oracle access, however, this property may not hold, although it will if the oracle itself is checkable. So their approach can be viewed very roughly in terms of ours, as taking our version of $\mathcal{RCT}$ and adding the constraint "$\mathcal{O}$ is locally checkable".

The authors call their refined theory $\mathcal{LCT}$, and point out that although $\mathcal{LCT}$ implies many known non-relativizing results, whether it can settle questions such as P versus NP is very hard to know. In fact, they observe that if P versus NP were shown beyond reach of $\mathcal{LCT}$ in the manner of Baker, Gill, Solovay — by giving contradictory relativizations with oracles satisfying the theory — then P would actually be separated from NP. In this sense, $\mathcal{LCT}$ is an unsatisfactory candidate for "current techniques". (Notice that if all we want is a theory that can derive the current theorems then we can just let $\mathcal{O}$ be empty.)

In a counterview to the AIV proposal dated around the same time, **Fortnow** [14] argues that the nonrelativizing ingredient in the proof of $\mathrm{PSPACE} \subset \mathrm{IP}$ is of an algebraic nature. We can interpret his key insight as follows. Although $\mathrm{PSPACE} \subset \mathrm{IP}$ does not relativize, it does in a weaker sense: Let $\widehat{\mathcal{O}}$ denote the affine extension of $\mathcal{O}$, as defined on page 5. (Strictly speaking Fortnow works over $\mathbb{Z}$ instead of $\mathrm{GF}(2^k)$.) Then $\mathrm{PSPACE}^{\mathcal{O}} \subset \mathrm{IP}^{\widehat{\mathcal{O}}}$, and consequently, $\mathrm{PSPACE}^{\mathcal{O}} \subseteq \mathrm{IP}^{\mathcal{O}}$ whenever $\widehat{\mathcal{O}}$ Cook-reduces to $\mathcal{O}$. Effectively, then, he defines a theory $\mathcal{ACT}$ by taking our version of $\mathcal{RCT}$ and adding the constraint $\widehat{\mathcal{O}} \in \mathrm{P}^{\mathcal{O}}$.

Although Fortnow does not prove any unprovability results for his theory, we can show that his version of $\mathcal{ACT}$ yields most of AW's classification of what algebrizes and what does not (hence the '✓̸' symbol) — but not all, as we explain later below.

A decade-and-half after the above two papers, **AW** [1] introduce algebrization. Their paper finesses the question of how relativization should be refined, by simply declaring that a statement $\mathrm{A} \subset \mathrm{B}$ relativizes algebraically if $\mathrm{A}^{\mathcal{O}} \subset \mathrm{B}^{\widehat{\mathcal{O}}}$ for every $\mathcal{O}$ (for a notion of $\widehat{\mathcal{O}}$ similar to our notion of affine extension), and that $\mathrm{A} \not\subset \mathrm{B}$ algebrizes if $\mathrm{A}^{\widehat{\mathcal{O}}} \not\subset \mathrm{B}^{\mathcal{O}}$. No definition is given for other types of statements, or for proofs.

Since we ultimately care about containments and their negations, the AW approach seems appealing. There are problems with it, however (page 1), chief among which is that not everything that relativizes can be said to algebrize. For example, the statement $(\exists \mathcal{C} : \mathcal{C} \subset \mathrm{NEXP} \wedge \mathcal{C} \not\subset \mathrm{P/poly}) \implies \mathrm{NEXP} \not\subset \mathrm{P/poly}$ is true no matter what NEXP or $\mathrm{P/poly}$ means — it is even true no matter what "is an element of" means — hence is relativizing, but it cannot be declared as algebrizing by building on the original definitions.

On the positive side, AW succeed in giving containments $\mathrm{A} \subset \mathrm{B}$ that do not algebrize, by showing that

an oracle $\mathcal{O}$ exists for which $\mathrm{A}^{\mathcal{O}} \not\subset \mathrm{B}^{\widehat{\mathcal{O}}}$. (There are similar examples for negations of containments.) This is a critical idea upon which subsequent work expands, including ours; we say more about this below.

Soon after the AW paper, **Impagliazzo, Kabanets, Kolokolova** [19] (IKK henceforth) resume the approach of AIV, and propose an intermediate theory between $\mathcal{RCT}$ and $\mathcal{LCT}$ that they call $\mathcal{ACT}$, short for arithmetic checkability theory. (They also define a variant, $\mathcal{ACT}^*$, but we blur the distinction here.)

We can view IKK's approach as being along the same line of Fortnow's, by considering the following task. Given ϕ and α, evaluate $\Phi(\alpha)$; here ϕ is a Boolean formula, Φ is any fixed low-degree polynomial interpolating ϕ (such as its arithmetization), and α are inputs from $\mathrm{GF}(2^{O(n)})$. (Like Fortnow, IKK work over $\mathbb{Z}$, but both approaches can be adapted to $\mathrm{GF}(2^k)$.) Call the decision version of this task — given i return the i^{th} bit of the result, for example — the language AF, short for arithmetized formula evaluation.

Clearly $\mathrm{AF} \in \mathrm{P}$. Indeed, this seems to be an essential feature of arithmetization: it would seem pointless to interpolate Boolean formulas into polynomials that we cannot evaluate efficiently. But if ϕ is over an arbitrary basis $\{\wedge, \oplus, \mathcal{O}\}$, then it does not seem that $\mathrm{AF}^{\mathcal{O}} \in \mathrm{P}^{\mathcal{O}}$ since the $\mathcal{O}$-gates within ϕ need to be extended somehow as well.

Now, in both IKK's approach and Fortnow's, we can interpret the starting point as restricting the oracle $\mathcal{O}$ so that $\mathrm{AF}^{\mathcal{O}} \in \mathrm{P}^{\mathcal{O}}$ becomes a true statement — in Fortnow's case via the constraint $\widehat{\mathcal{O}} \in \mathrm{P}^{\mathcal{O}}$, and in IKK's case, directly via $\mathrm{AF}^{\mathcal{O}} \in \mathrm{P}^{\mathcal{O}}$. The IKK constraint (rather, our interpretation of it) is implied by Fortnow's; this will be clear in Section 1.3 once we generalize arithmetization. Hence anything provable from the IKK constraint is automatically provable from Fortnow's. Although the converse is not known to hold, it does hold in the following sense. Anything *IKK show to be* unprovable from the IKK constraint, we can show is unprovable from Fortnow's constraint as well; we can do this using our observation on page 5, that affine extensions respect disjoint unions. So the two approaches currently seem to have the same power.

The key advantage of our approach over IKK's and Fortnow's is its avoidance of computational notions in restricting the oracle $\mathcal{O}$. By giving a direct algebraic restriction, namely that $\mathcal{O}$ equals $\widehat{f}$ for some language f, our approach allows us to expand on one of AW's critical ideas: using interpolation to show that certain statements ψ do not relativize algebraically (in our case, affinely). In contrast, neither IKK's approach nor Fortnow's is known to allow interpolation. Consequently, although IKK leave it as an open question for their framework, we can capture a key result of AW, that $\mathrm{NEXP} \not\subset \mathrm{P/poly}$ does not relativize algebraically.[5]

Another place where IKK diverges from AW concerns the theorem $\mathrm{NEXP} \subset \mathrm{MIP}$. As mentioned, AW showed that this theorem algebrizes under a machine-specific restriction of the class NEXP. While IKK did show an analogous result for their framework in the model-theoretic sense, they did not show it in the proof-theoretic sense; in fact they use a machine-free characterization of NEXP and cannot directly express the query restriction of AW, so it is not even clear a priori if the result itself can be expressed in their approach. Instead, IKK observe that under the proper, unrestricted definition of NEXP, the theorem does not algebrize, and suggest that there are additional ingredients underlying this theorem besides arithmetization, and point this out as a point of divergence from the AW thesis that algebrization captures "current techniques" [19, p.15]. As mentioned in page 3, our formulation of this theorem substantially clarifies the discussion.

Whether our formulation implies IKK's or Fortnow's, or vice versa, is not clear; we do not know if algebrizing in one sense can be shown to imply the other. What we *can* say, however, is that every statement that IKK show as algebrizing has an affinely relativizing proof, and that the opposite holds for those shown non-algebrizing by IKK — just as the case for AW. In particular, IKK show various compound statements to be non-algebrizing; these follow as consequences of results on simpler statements and can be shown in our framework as well (via what we call the proof theoretic approach in Section 4.3).

[5] In fact IKK leave open a weaker question: whether $\mathrm{EXP} \not\subset \mathrm{P/poly}$ can be shown to not algebrize in their framework [19, p.15]. The same question automatically applies to Fortnow's framework, since his constraint implies IKK's.

1.3 Overview of Ideas and Techniques

Defining affine relativization, and proving that it works, involve a number of observations as well as some technical ingredients. This section highlights the main ones.

Generalizing arithmetization using affine extensions. Our first observation concerns how the arithmetization method should be generalized to handle formulas over a generic Boolean basis, say $\{\wedge, \oplus, \mathcal{O}\}$ where $\mathcal{O}$ is an arbitrary language. In its typical description, the method states that the formula $\neg\phi$ arithmetizes as $1 - \Phi$ where Φ is the arithmetization of ϕ; similarly, $\phi \wedge \psi$ arithmetizes as $\Phi \cdot \Psi$. Other cases, such as $\vee$ and $\oplus$, are handled by reducing to these two.

We observe that $x \cdot y$ is the unique polynomial over $\mathbb{Z}$, of (individual) degree ≤ 1, that extends the Boolean function $(x, y) \mapsto x \wedge y$; in other words, it extends an $\wedge$-gate of fan-in 2. Similarly $1 - x$ extends a $\neg$-gate. We thus make the following generalization: Arithmetization replaces a Boolean gate $\mathcal{O}$, of fan-in m, with the gate $\widehat{\mathcal{O}}$ denoting the unique degree-≤ 1 polynomial

$$\widehat{\mathcal{O}}(x) := \sum_{b \in \{0,1\}^m} \mathcal{O}(b) \cdot \textstyle\prod_{i=1}^{m} (1 - x_i) \cdot (1 - b_i) + x_i \cdot b_i \qquad (\dagger)$$

that extends $\mathcal{O}$ from the Boolean domain to $\mathbb{Z}$. We call $\widehat{\mathcal{O}}$ *the (multi-)affine extension* of $\mathcal{O}$, and caution that the notation has nothing to do with Fourier analysis.

For our results we view (†) in fields of the form $\mathrm{GF}(2^k)$ only. There are several benefits to this, and we point them out as we explain our approach in this section. To begin with, we note that extension to $\mathrm{GF}(2^k)$ is conceptually cleaner, as it turns a function on n bits into a function on n vectors of k bits each. Also, in $\mathrm{GF}(2^k)$, the arithmetization of $\phi \oplus \psi$ becomes the natural $\Phi + \Psi$, whereas in other fields, neither $\oplus$, nor any other Boolean operator, gets arithmetized to $+$. Further, the equation (†) gets simplified, as the product inside becomes $\prod_{i=1}^{m}(1 + x_i + b_i)$. In fact by taking $\oplus \mapsto +$ and $\wedge \mapsto \times$ as the defining rules of arithmetization, its generalization to (†) can be arrived in another natural way, by first writing $\mathcal{O}$ in its "truth table form", i.e., as the exponential-size formula

$$\mathcal{O}(x) = (\mathcal{O}(0^m) \wedge x \equiv 0^m) \oplus \cdots \oplus (\mathcal{O}(1^m) \wedge x \equiv 1^m)$$

where $x \equiv b$ stands for $\bigwedge_i (1 \oplus x_i \oplus b_i)$, then arithmetizing, and then gathering the summands using an $\mathcal{O}$-gate.

Affine Relativization — capturing known uses of arithmetization. Consider a functional view of an $\widehat{\mathcal{O}}$-gate, as returning k bits when each of its inputs come from $\mathrm{GF}(2^k)$. In this view, arithmetizing a formula ϕ creates a family of formulas $\{\Phi_k\}$, with each Φ_k redundantly describing the behavior of ϕ on the Boolean domain — the larger k, the higher the redundancy (with $k = 1$ corresponding to ϕ itself).

Now if ϕ is over an arbitrary basis that includes $\mathcal{O}$-gates, then unlike the case for the standard basis, its arithmetization Φ does not seem to allow efficient evaluation, over say $\mathrm{GF}(2^{O(n)})$. Interpreting this to be the non-relativizing ingredient in proofs of $\mathrm{PSPACE} \subset \mathrm{IP}$, etc., we take the following approach to refine relativization.

The formula Φ, which redundantly encodes ϕ, is obtained from ϕ via a "local" transformation acting on its gates, namely by adding redundancy at the gates. Based on this, our idea is to have the oracle gates of ϕ compute not some arbitrary $\mathcal{O}$, but something that contains redundancy already, namely $\widehat{\mathcal{O}}$ for an arbitrary $\mathcal{O}$. The plan being then to show that arithmetization — rather, current uses of it — need not introduce redundancy at those gates, or, at least do so in a feasible way.

We arrive at our formulation thus: whereas a statement relativizes if it holds relative to every language $\mathcal{O}$, a statement relativizes *affinely*, if it holds relative to every language $\mathcal{A}$ of the form $\widehat{\mathcal{O}}$ for some $\mathcal{O}$. More

precisely, $\mathcal{A}$ encodes the family of polynomials $\{\widehat{\mathcal{O}}_m\}$ evaluated over $\mathrm{GF}(2^k)$ for all k, where $\mathcal{O}$ is an arbitrary language and $\mathcal{O}_m$ is its restriction to $\{0,1\}^m$. We also call $\mathcal{A}$ the *(multi-)affine extension* of $\mathcal{O}$.

Why was this notion not invented in 1994? Natural though it may seem, affine relativization poses the following difficulty: the very theorems that it is intended for, e.g. $\mathrm{PSPACE} \subset \mathrm{IP}$, do not appear to relativize affinely, at least not via a superficial examination of their proofs.

To see the issue, consider a property π of Boolean formulas — unsatisfiability, say. In proving $\pi \in \mathrm{IP}$ arithmetization is used as a *reduction*, from π to some property Π of arithmetic formulas — e.g., unsatisfiability of ϕ reduces, via arithmetization, to deciding if the product of $(1 + \Phi(\alpha))$, over all binary input vectors α, equals 1 in $\mathrm{GF}(2^k)$ for any k.

So each theorem of the form $\pi \in \mathrm{IP}$ is, in fact, a corollary of a more generic result of the form $\Pi \in \mathrm{IP}$, that gives an interactive protocol for an arithmetic property. It turns out those generic results can be further generalized, if we extend the arithmetic basis, from the standard $\times$-gates and $+$-gates — which are really $\widehat{\wedge}$- and $\widehat{\oplus}$-gates, respectively, per the first discussion above — by allowing $\widehat{\mathcal{O}}$-gates for an arbitrary $\mathcal{O}$. Then the same protocols that yield $\Pi \in \mathrm{IP}$ work just as well over this extended basis, given oracle access to the evaluation of $\widehat{\mathcal{O}}$. We may write $\Pi^{\widehat{\mathcal{O}}} \in \mathrm{IP}^{\widehat{\mathcal{O}}}$, where $\Pi^{\widehat{\mathcal{O}}}$ extends Π to formulas over the extended basis.

Now supposing we have a theorem $\pi \in \mathrm{IP}$, let us make a superficial attempt to extend its proof so that it yields $\pi^{\mathcal{A}} \in \mathrm{IP}^{\mathcal{A}}$ for some language $\mathcal{A}$; here π is a property of formulas, say over the basis $\{\wedge, \oplus\}$, and $\pi^{\mathcal{A}}$ is its extension to the basis $\{\wedge, \oplus, \mathcal{A}\}$. As just explained, the proof of $\pi \in \mathrm{IP}$ starts with a reduction, of the Boolean property π to an arithmetic property Π. Now here is the problem: what property do we reduce $\pi^{\mathcal{A}}$ to? By definition of arithmetization, it would be $\Pi^{\widehat{\mathcal{A}}}$, the extension of Π to formulas over the basis $\{\times, +, \widehat{\mathcal{A}}\}$. But then as just explained, we would be placing $\pi^{\mathcal{A}}$ in $\mathrm{IP}^{\widehat{\mathcal{A}}}$ — not in $\mathrm{IP}^{\mathcal{A}}$.

This seeming circularity — $\pi^{\mathcal{O}} \in \mathrm{IP}^{\widehat{\mathcal{O}}}$, $\pi^{\widehat{\mathcal{O}}} \in \mathrm{IP}^{\widehat{\widehat{\mathcal{O}}}}$, ... — can be interpreted as the main distraction from arriving at a natural notion such as ours. Indeed, all previous attempts to capture arithmetization [14, 1, 19], dating back to the 1994 article of Fortnow [14], can be interpreted as having to make compromises so as to break out of this circularity. For example, the AW notion of algebrization does this by declaring $\mathcal{C} \subset \mathcal{D}$ to algebrize if $\mathcal{C}^{\mathcal{O}} \subset \mathcal{D}^{\widehat{\mathcal{O}}}$ holds for every $\mathcal{O}$ (for a notion of $\widehat{\mathcal{O}}$ related to ours; there is a similar definition for $\mathcal{C} \not\subset \mathcal{D}$). We surveyed their approach and others in Section 1.2.

In contrast, our approach tackles circularity directly. The idea is to avoid the problematic reduction $\pi^{\mathcal{A}} \to \Pi^{\widehat{\mathcal{A}}}$, and to instead reduce $\pi^{\mathcal{A}}$ to $\pi^{\mathcal{O}}$ by somehow exploiting π whenever $\mathcal{A}$ is of the form $\widehat{\mathcal{O}}$ for some $\mathcal{O}$. Then the combined reduction $\pi^{\mathcal{A}} \to \pi^{\mathcal{O}} \to \Pi^{\mathcal{A}}$ breaks the circularity. This fulfils the plan of the previous discussion, namely to show that arithmetization, in its current uses, need not extend gates that are extensions of something already.

Relativizing $\oplus\mathrm{P} \subset \mathrm{IP}$. The idea of the previous discussion can be realized when π is the sum $\pi(\phi) := \oplus_x \phi(x)$, also known as the language $\oplus\mathrm{SAT}$. This is because when ϕ is a formula over the $\mathcal{A}$-extended Boolean basis, each occurrence of $\mathcal{A}$ evaluates the sum (†) over $\mathrm{GF}(2^k)$ for some k, and then returns, say, the i^{th} bit of the result given i. Therefore, if we step from $\mathrm{GF}(2^k)$ to $\mathrm{GF}(2)^k$, we can rewrite each occurrence of $\mathcal{A}$ as $\oplus_y \gamma(y)$, for some formula γ over the $\mathcal{O}$-extended Boolean basis. This becomes the reduction we want, once we show how to convert formulas involving sums to prenex form, i.e. such that all sums appear up front. It follows that $\oplus\mathrm{SAT} \in \mathrm{IP}$ — or equivalently, $\oplus\mathrm{P} \subset \mathrm{IP}$ — relativizes affinely.

Scaling to $\mathrm{PSPACE} \subset \mathrm{IP}$ — a proof sans degree reduction. Our approach for $\oplus\mathrm{P}$ can be adapted to show that $\mathrm{PSPACE} \subset \mathrm{IP}$ affinely relativizes as well. However, we find a more natural approach which yields another proof of this theorem; this may be of separate interest because current proofs, as far as we

know, employ syntactic tricks in order to control the degree of polynomials that arise from arithmetizing instances of a PSPACE-complete problem (e.g., [25, 6, 26, 2]).

In contrast we show, directly, that every downward-self-reducible language has an interactive protocol, by essentially bootstrapping the very fact that $\oplus\mathrm{P} \subset \mathrm{IP}$ relativizes affinely. In particular, we make no use of a specific PSPACE-complete problem; we do not even use any additional arithmetization beyond what is needed for $\oplus$SAT. (We emphasize that the new proof is sketched here because it might be of separate interest. The standard proofs of this theorem can also be adapted to our framework.)

The new proof goes as follows. If L is downward-self-reducible, then on inputs x of length n, it can be expressed as a $\mathrm{poly}(n)$-size circuit over the L-extended Boolean basis, of fan-in at most $n-1$. This circuit in turn can be expressed as the sum $\oplus_y \phi(x,y)$, where ϕ is a formula verifying that y represents the computation of the circuit on input x. In notation we may summarize this reduction as

$$L_n \rightarrow \oplus\mathrm{SAT}^{L_{n-1}} \qquad (*)$$

where $\oplus\mathrm{SAT}^{f_m}$ is the extension of $\oplus$SAT to formulas over the f-extended Boolean basis, of fan-in at most m. Repeating $(*)$ for L_{n-1} instead of L_n, we get

$$\oplus\mathrm{SAT}^{L_{n-1}} \rightarrow \oplus\mathrm{SAT}^{\oplus\mathrm{SAT}^{L_{n-2}}} \rightarrow \oplus\mathrm{SAT}^{L_{n-2}} \qquad (**)$$

where the first reduction is because extending the basis is functorial in the sense that $f \rightarrow g$ implies $\oplus\mathrm{SAT}^f \rightarrow \oplus\mathrm{SAT}^g$, and the second reduction follows by bringing sums to prenex form as mentioned in the previous discussion. Note that the reduced formula is now of size about n^{2d}, if the one in $(*)$ is of size n^d.

The idea is to tame the growth in the size of the reduced formulas, by using interaction. Building on the ideas of the previous discussion, it is easy to show a protocol yielding the *interactive* reduction

$$(\oplus\mathrm{SAT}^{f_m})_{n^d} \rightarrow (\oplus\mathrm{SAT}^{f_m})_{n^c}$$

that compresses instances to $\oplus\mathrm{SAT}^{f_m}$ of size n^d down to size n^c, for an arbitrarily large d and a *fixed* c, for every language f, in particular for $f = L$. (We sketch the protocol below on page 13.)

Thus we can keep repeating $(**)$ to get

$$L_n \rightarrow \oplus\mathrm{SAT}^{L_{n-1}} \rightarrow \oplus\mathrm{SAT}^{L_{n-2}} \rightarrow \cdots \rightarrow \oplus\mathrm{SAT}^{L_{O(1)}}$$

provided we interleave a compression phase whenever the formula size exceeds n^c. Since an L-gate of constant fan-in can be expressed as a constant-size formula, $\oplus\mathrm{SAT}^{L_{O(1)}}$ reduces to $\oplus$SAT. So $L \in \mathrm{IP}$ as desired.

That this proof affinely relativizes is straightforward to show; we enlarge the basis of the $\oplus$SAT-instances with an arbitrary affine extension $\mathcal{A}$, and employ the same ideas.

(Interestingly, just as this proof builds on the relativization of $\oplus\mathrm{P} \subset \mathrm{IP}$, we use the relativization of $\mathrm{PSPACE} \subset \mathrm{IP}$ in turn to give a streamlined proof of the $\mathrm{NEXP} \subset \mathrm{MIP}$ theorem, that uses no specific NEXP-complete problem nor any additional arithmetization; see Section 3.3.)

NEXP vs. MIP — the role of locality. As mentioned in the introduction, AW show that $\mathrm{NEXP} \subset \mathrm{MIP}$ algebrizes only under a restriction, and a questionable one at that, of the oracle access mechanism for NEXP.[6] Since we define complexity classes using P, it would be even more artificial to try to express this restriction in our framework. Instead, we find a natural approach that also sheds some light into the issues surrounding oracle access.

[6] We caution that neither AW, nor we, advocate or assume that NEXP be *always* relativized in this restricted way. It is only for the purpose of deriving this theorem that this restriction seems inevitable — and this discussion investigates why.

Consider generalizing the class IP, by replacing in its definition the popular constant $2/3$ with γ, so that if the input x is supposed to be rejected, then the verifier erroneously accepts x with probability $< 1 - \gamma$. (If x should be accepted, then, as before, it is.) Call this class γ-gap-IP.

It is easy to see, by the classical PSPACE-completeness result of Stockmeyer and Meyer [28], that 0-gap-IP is identical to PSPACE. Therefore PSPACE $\subset$ IP can be broken into the containments

$$\text{PSPACE} \subset \text{0-gap-IP} \subset \Omega(1)\text{-gap-IP}$$

with the second containment, "gap amplification", being the actual content of the theorem.

The corresponding case for NEXP $\subset$ MIP becomes revealing. Of the containments

$$\text{NEXP} \subset \text{0-gap-MIP} \subset \Omega(1)\text{-gap-MIP}$$

only the second one, gap amplification, affinely relativizes as we show in Section 3.3. So what "current technique" is it that yields the first containment, that affine relativization cannot capture?

It is locality, more specifically, *polylog*-locality, which yields the following variant of the Cook-Levin theorem: A language is in P iff it has circuits that are polylog-time uniform, i.e., iff it is computable by a family $\{C_n\}$ of circuits, such that given (n, i), the task to produce the type of the ith gate of C_n, as well as the indices of all gates connected to it, can be performed in $\text{poly} \log n$ time. Intuitively, this theorem does not relativize, even affinely, simply because it restricts the circuits to have polylogarithmic fan-in.

In our framework, we define P via *poly*-locality instead of polylog, and use P itself to express polylog-locality (by saying that the above function on pairs (n, i) is in FP) and call the class thus obtained P_{local}, the subclass of P satisfying the above locality theorem. We then use P_{local} to define NP_{local}, $\text{NEXP}_{\text{local}}$, etc. Immediately two things fall out of these definitions. First, that 0-gap-MIP is identical to $\text{NEXP}_{\text{local}}$, so locality does capture the first containment above. Second, that $\text{NEXP}_{\text{local}}$ is equivalent to the dubious version of NEXP with polynomial-length oracle queries (equivalent in the model-theoretic sense), making it not so dubious after all.

We do not know of any result using NEXP $\subset$ MIP, that would break down if $\text{NEXP}_{\text{local}} \subset$ MIP is used instead — in fact we do not know of any result using NEXP $\subset$ MIP, period. We conclude that locality arises in NEXP $\subset$ MIP only definitionally; it is an ingredient that has not been exploited beyond making definitions. (It would be interesting to know if the same reasoning could apply to the PCP theorem; see open problem in Section 5.)

NEXP vs. P/poly — a coding-theoretic interpolation lemma. One of the technical contributions of the paper is in showing that certain statements ψ do not relativize affinely. As usual (though not always), this entails constructing an eligible language — an affine extension $\mathcal{A}$ in our case — relative to which ψ is false.

For some ψ, this task turns out to be straightforward given prior work. Such ψ are of the form $\mathcal{C} \subset \mathcal{D}$, for which AW invented an approach based on communication complexity that also works in our setting.

For other ψ, however, in particular for NEXP $\not\subset$ P/poly, we need new ideas. While AW [1] did construct a $\mathcal{Q}$ such that $\text{NEXP}^{\mathcal{Q}} \subset \text{P}^{\mathcal{Q}}/\text{poly}$, they did this only for a multi-*quadratic* extension, i.e., for $\mathcal{Q}$ encoding a family of polynomials where each member has (individual) degree-≤ 2, instead of degree-≤ 1. It seemed "crucial" [1], in fact, to increase the degree for this purpose. While quadratic extensions suffice for the AW notion of algebrization, they do not for our notion of affine relativization.

As the key technical ingredient for this purpose, we derive a coding-theoretic ingredient (Lemma 33 and Theorem 34), stating that knowing t bits of a codeword exposes at most t bits of its information word, and this holds for every binary code, including the affine extension (over $\text{GF}(2^k)$).

AW implicitly proved a weaker form of this fact, involving quadratic polynomials. One of the ideas that enables us to do better, is to consider a different formulation for what it means to "expose" a bit of the information word. Whereas the AW approach (implicitly) considers each exposed bit as being completely

revealed, our approach gives a finer treatment: an exposed bit is one whose *location* is revealed, but whose contents may vary as a function of the unexposed bits.

The advantage of this refinement is that it allows us to show, given t bits of a codeword, that the set of all codewords agreeing on these t bits form an affine space, of dimension at most t less than the maximum possible. In contrast, the AW approach resorts to using indicator polynomials to surgically alter, bit-by-bit, the codeword whose t bits are revealed; this inevitably raises the degree to quadratic because each indicator polynomial must also vanish on the t points that are revealed, in addition to all-but-one point of the Boolean cube.

Compressing $\oplus$SAT. For the sake of completing the sketch of the alternate proof of $\mathrm{PSPACE} \subset \mathrm{IP}$ explained earlier, we now outline the compression protocol mentioned.

The protocol is based on the fact alluded to earlier, that $\oplus\mathrm{SAT}^f \in \mathrm{IP}^{\widehat{f}}$ for any language f. This fact follows from standard considerations: Given ϕ over the f-extended basis, in order to compute $\oplus_z \phi(z)$, the verifier: (i) arithmetizes ϕ to get Φ, a formula over the $\widehat{f}$-extended arithmetic basis, (ii) engages in a sumcheck protocol [5], thus reduces the original task to that of evaluating Φ over $\mathrm{GF}(2^k)$, with $k \in O(\log n)$ being sufficient for ϕ of size n, and (iii) evaluates Φ, by using the $\widehat{f}$-oracle for the $\widehat{f}$-gates.

The compression protocol also starts out as above. The difference begins in step (iii): instead of calling the $\widehat{f}$-oracle, the verifier engages the prover. By using standard interpolation techniques, the verifier reduces the task of computing the values of $\widehat{f}$ on up to n points, to doing the same on just m points or fewer, where m is the largest fan-in of any f-gate in the formula ϕ.

Thus the output of step (iii) is a list of at most m claims of the form "$\widehat{f}_{m'}(x) = v$" with $m' \leq m$ and $v, x_i \in \mathrm{GF}(2^k)$. Now because $\widehat{f}_{m'}$ is merely the sum (†) on page 9, which can be viewed in $\mathrm{GF}(2)^k$ rather than in $\mathrm{GF}(2^k)$, it follows that these claims can be expressed as a conjunction of $\oplus\mathrm{SAT}^{f_m}$-instances, of combined size $\mathrm{poly}(mk)$. This yields the compressed instance, since $\oplus\mathrm{SAT}$ is closed under conjunction.

2 Definitions, Notation and Conventions

$\mathcal{O}$ and $\mathcal{A}$. Unless stated otherwise, $\mathcal{O}$ stands for an arbitrary language, and $\mathcal{A}$ for its affine extension as defined below.

Well-behaved resource bound. We call a function $s : \mathbb{N} \to \mathbb{N}$ a *well-behaved resource bound* if it is increasing, computable in time polynomial in its input value, and satisfies $O(s(n)) \subset s(O(n)) \subset s(n)^{O(1)} \subset s(n^{O(1)})$ and $n \leq s(n)$. Functions of the form $n^d, n^d \log n, 2^{(\log n)^d}, 2^{dn}$ are well-behaved resource bounds.

The above generalizes to $s : \mathbb{N}^2 \to \mathbb{N}$ if fixing either of the inputs yields a well-behaved resource bound.

Languages as families. We view languages $L : \{0,1\}^* \to \{0,1\}$ as families of Boolean functions $\{L_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$, though we sometimes specify them as doubly-indexed families of the form $\{f_{m,k} : \{0,1\}^{s(m,k)} \to \{0,1\}\}_{m,k\in\mathbb{N}}$, where $s : \mathbb{N}^2 \to \mathbb{N}$ is a well-behaved resource bound that is polynomially bounded in mk.

It is an elementary fact that a family of the latter kind can be efficiently viewed as one of the former, by using a pairing function and padding. For the sake of concreteness, given $\{f_{m,k}\}$, let $m \diamond k$ denote the Cantor pairing of m and k, and define $\{L_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$ as $L_n(x_1..x_n) := f_{m,k}(x_1..x_{s(m,k)})$ for the largest $m \diamond k$ such that $s(m \diamond k, m \diamond k) \leq n$.

Representing $\mathbb{F}_{2^k}$. We represent each element of $\mathbb{F}_{2^k}$ by a k-bit Boolean vector, forming the coefficients of a polynomial in the ring $\mathbb{F}_2[x]$ mod some irreducible $p_k(x)$ of degree k. We fix a uniform collection $\{p_k\}_{k\in\mathbb{N}}$ so that a deterministic algorithm can produce p_k in time polynomial in k [27].

The Boolean version of $q : \mathbb{F}_{2^k}^m \to \mathbb{F}_{2^k}$ is, for concreteness, the function $\mathrm{bool}(q)$ mapping (x, i) to the i^{th} bit of $q(x)$. Our results do not depend on this definition; any other equivalent function (under Cook reductions) would work.

Affine extensions. (This definition uses all the definitions above.)

Given a total Boolean function $f_m : \{0,1\}^m \to \{0,1\}$, we define its *affine extension polynomial* as the unique m-variate polynomial over $\mathbb{F}_2$, with individual degree ≤ 1, that agrees with f_m over $\mathbb{F}_2$ (given explicitly in (†) on page 9 .) We denote this polynomial by $\widehat{f_m}$.

By the *affine extension* of f_m we mean the family

$$\left\{\mathrm{bool}\left(\widehat{f_m}^k\right)\right\}_{k\in\mathbb{N}}$$

where $\widehat{f_m}^k$ is the function that evaluates $\widehat{f_m}$ over $\mathbb{F}_{2^k}$.

Given a family F of functions f_m for various m, we define its affine extension (polynomial) as the family obtained by applying the above definitions to each member.

Boolean bases. We define a Boolean basis inductively, as either the set $B_{std} := \{0, 1, \oplus, \wedge\}$, or the set $B \cup \{f\}$ where B is a Boolean basis, and f is a (possibly partial) language. We refer to B_{std} as *the standard Boolean basis*, and to $B \cup \{f\}$ as the basis B *extended with* f.

The (partial) language $\oplus\mathrm{SAT}^f$. For every (partial) language f and Boolean basis B, we define $\oplus\mathrm{SAT}^f$ as the (partial) language mapping $\phi(\vec{x})$ to the evaluation of the mod-2 sum $\oplus_{\vec{\alpha}}\phi(\vec{\alpha})$, where ϕ denotes a formula over the basis B extended with f. By default B is the standard basis.

We index $\oplus\mathrm{SAT}^f$ by n, an upper bound on the number of gates of the formula ϕ.

$\oplus\mathrm{SAT}^f$ is undefined on those ϕ, and only on those, that are undefined on some $\vec{\beta}$ (due to some gate of ϕ receiving inputs out of its domain while evaluating $\phi(\vec{\beta})$). Notice that over the standard basis, if f is a language then so is $\oplus\mathrm{SAT}^f$, and the same holds if the standard basis is extended with any language.

(Interactive) Reductions. For (partial) languages f and g, we write

$$f \to g$$

if there is an interactive protocol Π such that given x and ε, the protocol runs in time $\mathrm{poly}(n/\varepsilon)$ and ends with the verifier outputting y, such that if $x \in \mathrm{dom}\, f$ then with probability at least $1-\varepsilon$: (i) $y \in \mathrm{dom}\, g$, and (ii) $f(x) = g(y)$.

We use the same notation even if Π uses no interaction, and even when it is deterministic.

3 Positive Relativization Results

This section shows that the famous results on interactive proofs admit affinely relativizing proofs, as do the circuit lower bounds that build on them. These are the IP theorem of Shamir ($\mathrm{PSPACE} \subset \mathrm{IP}$, Section 3.2), the MIP theorem of Babai, Fortnow, and Lund ($\mathrm{NEXP} \subset \mathrm{MIP}$, Section 3.3), the ZKIP theorem of Goldwasser, Micali, and Wigderson ($\mathrm{NP} \subset \mathrm{ZKIP}$ if one-way functions exist, Section 3.5), and the strongest lower bounds known-to-date against general Boolean circuits, by Buhrman, Fortnow, Thireauf, and by Santhanam (Section 3.4). All of these build on several properties of $\oplus\mathrm{SAT}$ developed in Section 3.1.

(As explained in Section 1.1, we do not state these results in proof-theoretic terms; e.g., Theorem 2 asserts, among other things, that $\oplus\mathrm{SAT} \subset \mathrm{IP}$ affinely relativizes, rather than that it has such a proof, even though the latter also holds.)

3.1 Checking and Compressing $\oplus\mathrm{SAT}$

This section develops two theorems and two propositions on $\oplus\mathrm{SAT}$ that enable most of the positive relativization results in the paper.

We first recall some notions from program checking [10].

Definition 1 (Same-length checkable)**.** We say a language $L := \{L_n\}_{n\in\mathbb{N}}$ is *same-length checkable* if there is an interactive protocol for computing L_n, i.e. for deciding the language $(x, b) \mapsto L_n(x) \equiv b$, wherein the prover acts as a purported oracle for L_n. We say L is *checkable* if the prover is allowed to answer queries also for $L_{\neq n}$. The verifier in these protocols is called a *checker* for L.

The first main result in this section shows (via an affinely relativizing proof) the existence of a $\oplus$P-complete language that is same-length checkable.

Theorem 2 (Checking $\oplus\mathrm{SAT}$)**.** *$\oplus\mathrm{SAT}$ is checkable. In fact, $\oplus\mathrm{SAT}$ is equivalent, under Karp reductions, to some language that is* same-length *checkable.*

This also holds if we extend the standard Boolean basis with $\mathcal{A}$, and give the checker access to $\mathcal{A}$.

Theorem 2 is used, from different aspects, in deriving Shamir's IP theorem (Section 3.2) and the circuit lower bounds of Buhrman et al. and of Santhanam (Section 3.4).

The second result gives an interactive compression scheme for $\oplus\mathrm{SAT}^L$, which cuts the size of a formula from n^d to n^c, for an arbitrary large d and a fixed c, as long as the L-gates have fan-in $O(n)$ in the original formula. (The runtime of the interaction depends on d.) The verifier in the interaction need not have oracle access to L; in fact L may even be undecidable as far as the verifier is concerned.

Theorem 3 (Compressing $\oplus\mathrm{SAT}$)**.** *For every language $L := \{L_m\}_{m\in\mathbb{N}}$, there is an interactive protocol that reduces instances of $\oplus\mathrm{SAT}^{L\leq m}$ of size n, to instances of $\oplus\mathrm{SAT}^{L\leq m}$ of size $\mathrm{poly}(m \log n)$ for every m, n.*

This also holds if we extend the standard Boolean basis with $\mathcal{A}$, and give the protocol access to $\mathcal{A}$.

Theorem 3 is used in deriving Shamir's IP theorem (Section 3.2) with a new proof of that result.

We now give two auxiliary facts that come in handy when proving Theorems 2-3 and the IP theorem. The first is a consequence of the fact that an arithmetic expression involving binary-valued sums can be written in prenex form, i.e., so that all the sums appear up front.

Proposition 4. $\oplus\mathrm{SAT}^{\oplus\mathrm{SAT}} \to \oplus\mathrm{SAT}$. *The reduction is universal in the choice of a Boolean basis.*

The second fact is that extending the Boolean basis is a functor, from the category of all (partial) languages f, to the category consisting of $\oplus\mathrm{SAT}^f$ for every f, where the morphisms are Karp reductions.

Proposition 5. *If $f \to g$ via a deterministic reduction, then $\oplus\mathrm{SAT}^f \to \oplus\mathrm{SAT}^g$, for all (partial) languages f and g. The implied reduction is universal in the choice of a Boolean basis if the assumed reduction is.*

Although Proposition 5 concerns $\oplus\mathrm{SAT}$, the fact holds more broadly, e.g., for the language $\mathrm{CircEval}$ that evaluates a given circuit (and the latter is what we essentially prove).

3.1.1 Proofs of Theorems 2-3 and Propositions 4-5

We now proceed to prove the claims made in the previous section — first the propositions then the theorems.

We begin by introducing arithmetic bases.

Definition 6 (Arithmetic basis)**.** For every Boolean basis B, define the *arithmetic basis* $\widehat{B}$ as the set comprising all constants in $\mathbb{F}_{2^k}$ for each k, and $\widehat{f}$ for each $f \in B$. Let the standard arithmetic basis be $\widehat{B}_{std}$.

Consider the class of formulas obtained inductively, by first letting in every arithmetic formula, and then inductively letting in $\sum_{y\in\{0,1\}} \psi$ for every ψ already let in and for every variable y. Let us refer to such formulas as arithmetic expressions involving binary sums.

We claim that an arithmetic expression $\Psi(x)$ involving binary sums can be rewritten in prenex form, i.e., as $\sum_y \Phi(x, y)$ where Φ is a summation free formula, in time polynomial in the size of the formula.

This is trivial to see if $\Psi(x)$ is of the form $\Psi_1 \cdot \Psi_2$, and if by recursion Ψ_1 is already brought to prenex form $\sum_y \Phi_1(x, y)$, and Ψ_2 to $\sum_z \Phi_2(x, z)$: then just make sure y and z refer to disjoint sets of variables by renaming as needed, and write $\Psi(x) = \sum_{y,z} \Phi_1(x, y) \cdot \Phi_2(x, z)$.

In case $\Psi = \Psi_1 + \Psi_2$, after recursing and renaming as before, write

$$\Psi(x) = \textstyle\sum_{b,y,z} \big(\Phi_1(x, y) \cdot b \cdot \Pi_i z_i \;+\; \Phi_2(x, z) \cdot (1 - b) \cdot \Pi_i y_i\big),$$

where b is a single variable.

This proves the claim for standard arithmetic operators. More generally, Ψ may be of the form $\widehat{\mathcal{O}}(\Psi_1,.., \Psi_m)$; in that case, use the definition of $\widehat{\mathcal{O}}(x_{1..m})$ (see (†) in Section 1.3) to rewrite Ψ as

$$\Psi(x) = \textstyle\sum_{b_1..b_m} \widehat{\mathcal{O}}(b_1,.., b_m) \cdot \prod_i (1 + \Psi_i(x) + b_i), \tag{1}$$

and then recurse into the product on the right side.

As a special case, we have Proposition 4:

Proof of Proposition 4. Let B be any Boolean basis, say, without loss of generality, the standard basis extended with $\mathcal{O}$. Given a formula ψ over the basis B extended with $\oplus$SAT, we want to reduce the task of computing $\oplus_x \psi(x)$ to that of computing $\oplus_z \phi(z)$, for some ϕ over the basis B.

Replace each occurrence of $\oplus$SAT in ψ with the actual sum to be computed. This gives an expression ψ' of the desired form, except for the occurrence of $\oplus$-quantifiers within.

Now observe that the above transformation involving arithmetic expressions applies just as well to ψ' (essentially because a Boolean expression can be viewed as an $\mathbb{F}_2$-arithmetic expression and vice versa, and because the transformation does not introduce any non-binary constants). This completes the reduction. □

We use Proposition 4 to prove the Proposition 5:

Proof of Proposition 5. Let R be an algorithm realizing the reduction $f \to g$, i.e., $R := \{R_n(x, i)\}$ is a uniform family of circuits such that if $y_i := R_{|x|}(x, i)$ then $f(x) = g(y)$, where $|y| \in \mathrm{poly}(n)$. Consider an f-gate, say of fan-in n. We can implement this gate with the circuit $g \circ R_n$, i.e. the circuit R_n composed with a g-gate, and this circuit can in turn be expressed as the sum $\oplus_y \phi(x, y)$ where ϕ is the formula verifying that y describes the computation of the circuit on x. This shows $f \to \oplus\mathrm{SAT}^g$, and the claim follows by Proposition 4. □

We now turn to Theorems 2 and 3. We start with defining the language claimed to exist in Theorem 2.

Definition 7 ($+$ASAT)**.** For every (partial) language f and Boolean basis B, define $+\mathrm{ASAT}^f$ as the Boolean version of the function mapping $\Phi(\vec{x})$ to the evaluation of the sum $\Sigma_{\vec{\alpha}} \Phi(\vec{\alpha})$; here, α_i ranges over $\{0, 1\}$, and Φ is a formula over the arithmetic basis $\widehat{B}$ extended with $\widehat{f}$. By default B is the standard basis.

We index $+\mathrm{ASAT}$ by n and k, and write the corresponding member as $+_k\mathrm{ASAT}_n$, where n upper bounds the size of the formula Φ as measured by the number of nodes, and k denotes the field $\mathbb{F}_{2^k}$ where the constants of Φ reside. For our purposes (to be made clear in the proof of Lemma 10) we require $k \geq 3 \log n$.

We break the proofs of Theorems 2-3 into three common lemmas. The first two give the equivalence of $\oplus\mathrm{SAT}$ and $+\mathrm{ASAT}$.

Lemma 8 (Arithmetization)**.** $\oplus\mathrm{SAT} \rightarrow +\mathrm{ASAT}$. *The reduction is universal in the choice of a Boolean basis. In addition, the same reduction gives* $\oplus\mathrm{SAT}_n \rightarrow +_k\mathrm{ASAT}$ *for some* $k \in O(\log n)$.

Lemma 9 (Booleanization)**.** $+\mathrm{ASAT} \rightarrow \oplus\mathrm{SAT}$. *The reduction is universal in the choice of a Boolean basis.*

We call Lemma 9 Booleanization, as it involves a switch of basis from arithmetic to Boolean.

Proof of Lemma 8. Given ϕ over any Boolean basis, let Φ be its arithmetization as defined in Section 1.3. By the way we represent $\mathbb{F}_{2^k}$ (Section 2), computing $\oplus_\alpha \phi(\alpha)$ reduces to computing the least significant bit of the sum $\Sigma_\alpha \Phi(\alpha)$ over $\mathbb{F}_{2^k}$ for any k, where each α_i ranges over $\{0, 1\}$ in both sums. □

Proof of Lemma 9. Given $\Phi(x)$ and ℓ, we want to express the ℓ^{th} bit of $\Sigma_x \Phi(x)$ as some mod-2 sum $\oplus_z \phi(z)$. We may assume that there is only one nonstandard basis element in Φ, say $\widehat{f}$; this is without loss of generality because whatever we do for $\widehat{f}$-gates will apply just as well to any other nonstandard element. Thus we want ϕ to be over the standard Boolean basis extended with f.

To begin with, let us assume that there is no $\widehat{f}$-gate in Φ, in other words, that Φ is a $\mathbb{F}_{2^k}$-polynomial for some k. By the way we represent $\mathbb{F}_{2^k}$ (Section 2), there is a Boolean circuit $C(X)$ that takes as input a k-bit vector X_j corresponding to each input x_j of $\Phi(x)$, and outputs k bits representing the value $\Phi(x)$. C is constructible in polynomial-time given Φ.

Because we want the ℓ^{th} bit of the sum $\Sigma_x \Phi(x)$, and because addition in $\mathbb{F}_{2^k}$ corresponds to componentwise addition in $\mathbb{F}_2^k$, we can ignore all output bits of C except the ℓ^{th} one. Further, because the summation variables x_i range over binary values, we can fix in each X_i all the bits to 0 except the least significant bit, which we can call x_i. So we now have a circuit $C(x)$ returning the ℓ^{th} bit of $\Phi(x)$ for every x from the Boolean domain.

It follows that the ℓ^{th} bit $\Sigma_x \Phi(x)$ equals $\oplus_{x,y} \phi(x, y)$, where ϕ is the formula verifying that y describes the computation of the circuit C on input x. This proves the theorem when $\Phi(x)$ is a polynomial.

Now suppose that Φ contains $\widehat{f}$-gates. Mimicking the above reasoning for the standard basis, we want to express the evaluation of Φ as a Boolean circuit with f-gates. Once this is done, the rest would follow as above.

Perform the process, explained before the proof of Proposition 4, of bringing Φ to prenex form — a seemingly useless thing to do as Φ does not involve sums. But notice that as a side effect, the process transforms the summation-free $\Phi(x)$ into the sum $\Sigma_B \Phi'(x, B)$, where each $\widehat{f}$-gate in Φ', say the i^{th} one, is "isolated" in the sense that its inputs now come from some $B_{i1}, .., B_{im_i}$ among the variables B, which all range over Boolean values.

It thus follows, similar to before, that the ℓ^{th} bit of $\Sigma_x \Phi(x)$ equals $\oplus_{x,B,y} \phi'(x, B, y)$, where ϕ' is over the standard basis extended with f. This finishes the proof. □

The last lemma we need for proving Theorems 2 and 3, essentially says that claims of the form '$\Sigma_x \Phi(x) = v$' are same-length checkable, provided that the checker has oracle access to each gate in Φ. If there is no

access for one type of gate, say the $\widehat{f}$-gates, then instead of a yes/no answer, it can output a small conjunction of claims of the form '$\widehat{f}(z) = w$' where "small" means no more than m conjuncts if the fan-in of the $\widehat{f}$-gates in Φ is at most m.

Lemma 10. *For every language $L := \{L_m\}_{m\in\mathbb{N}}$, there is a same-length checker V that reduces $+\mathrm{ASAT}^L$ to the task of verifying, in parallel, multiple claims regarding the affine extension of L.*

In particular, $+_k\mathrm{ASAT}^{L^{\leq m}}$ gets reduced to at most m claims of the form '$\widehat{L}_i^k(y) = v$' where $i \leq m$.

This also holds if we extend the standard Boolean basis with $\mathcal{O}$, and give the checker access to $\mathcal{A}$.

Proof of Lemma 10. The checker V is to verify that the ℓ^{th} bit of $\Sigma_x\Phi(x)$ equals b, given (Φ, ℓ, b); here Φ has constants in $\mathbb{F}_{2^k}$ and hence the sum is over $\mathbb{F}_{2^k}$. V works as follows:

First, it obtains the rest of the k bits for $\Sigma_x\Phi(x)$, so the claim is now '$\Sigma_x\Phi(x) = u$' for some $u \in \mathbb{F}_{2^k}$.

Second, it performs the sumcheck protocol [5] over $\mathbb{F}_{2^k}$ to get rid of the sum and update the claim to '$\Phi(y) = v$' for some y, v. Notice that the field size remains the same.

At this point, V obtains the value of each gate in the evaluation of $\Phi(y)$ — i.e., the evaluation of each subformula of Φ on y — and checks all of them except those involving $\widehat{L}$.

Finally, V uses the interpolation technique from the LFKN protocol [22], and combines multiple claims of the form '$\widehat{L}_i(z) = w$' into a single one, for each distinct i. This completes the description of the checker.

By representing each node of Φ with a string of $O(k)$ bits, we can ensure that the queries made by the checker remain the same length. The analysis of the protocol is standard, and the requirement that $k \geq 3\log n$ in the definition of $+\mathrm{ASAT}$ ensures soundness. The claim follows. □

Before we finally prove Theorems 2-3, let us note one consequence of Propositions 4-5 and Lemma 9:

Corollary 11. *Let $\mathcal{O}$ be a language, possibly partial, and let $\mathcal{A}$ be its affine extension. Then $\oplus\mathrm{SAT}^{\mathcal{A}} \to \oplus\mathrm{SAT}^{\mathcal{O}}$. The reduction is universal in the choice of a Boolean basis.*

Proof. Being the affine extension of $\mathcal{O}$, on input (x, ℓ), $\mathcal{A}$ gives the ℓ^{th} bit of the value $\widehat{\mathcal{O}}$ takes at x. In other words, $\mathcal{A}$ computes the $+\mathrm{ASAT}^{\mathcal{O}}$ instance (Φ, ℓ) where Φ is the formula '$\widehat{\mathcal{O}}(x)$'. Thus $\mathcal{A} \to +\mathrm{ASAT}^{\mathcal{O}}$, and by the Booleanization Lemma, $+\mathrm{ASAT}^{\mathcal{O}} \to \oplus\mathrm{SAT}^{\mathcal{O}}$. Then by Propositions 5 and 4,

$$\oplus\mathrm{SAT}^{\mathcal{A}} \to \oplus\mathrm{SAT}^{\oplus\mathrm{SAT}^{\mathcal{O}}} \to \oplus\mathrm{SAT}^{\mathcal{O}}$$

as claimed. □

Proof of Theorem 2. We are to show that there is a language K that is equivalent to $\oplus\mathrm{SAT}^{\mathcal{A}}$ under Karp reductions; K further must be same-length checkable given access to $\mathcal{A}$. We claim that $K := +\mathrm{ASAT}^{\mathcal{O}}$ would do.

We begin by showing that K and $\oplus\mathrm{SAT}^{\mathcal{A}}$ reduce to each other. In one direction we have

$$K \to \oplus\mathrm{SAT}^{\mathcal{O}} \to \oplus\mathrm{SAT}^{\mathcal{A}},$$

where the first reduction is given by the Booleanization Lemma, and the second by using Proposition 5 and the fact that $\mathcal{O}$ reduces to its affine extension $\mathcal{A}$. For the other direction, do the same sequence of reductions in reverse, by first using Corollary 11 and then the Arithmetization Lemma.

Next, the same-length checkability of K given access to $\mathcal{A}$, is immediate from Lemma 10 by setting L to the empty language $L : x \mapsto 0$.

Finally, that $\oplus\mathrm{SAT}^{\mathcal{A}}$ is checkable follows from its equivalence to K which itself is checkable: On input x, reduce it to an input x' for K, then simulate the checking protocol for $K(x')$, by reducing each query for K to one for $\oplus\mathrm{SAT}^{\mathcal{A}}$. □

Proof of Theorem 3. Extend the standard Boolean basis with $\mathcal{A}$. We are to show that there is an interactive protocol yielding the reduction

$$\oplus\mathrm{SAT}_n^{L_{\leq m}} \to \oplus\mathrm{SAT}_{\mathrm{poly}(m\log n)}^{L_{\leq m}},$$

assuming the protocol is given access to $\mathcal{A}$.

Let us make a notational convention, and use $\oplus\mathrm{SAT}^{f,g}$, to refer to either of $(\oplus\mathrm{SAT}^f)^g$ and $(\oplus\mathrm{SAT}^g)^f$ depending on context, as they are the same (partial) language by definition (Section 2).

We proceed with the proof. We have

$$\oplus\mathrm{SAT}^{\mathcal{A},L_{\leq m}} \to \oplus\mathrm{SAT}^{\mathcal{O},L_{\leq m}} \to +\mathrm{ASAT}^{\mathcal{O},L_{\leq m}},$$

where the first reduction is by Corollary 11 and the second by the Arithmetization Lemma. In fact, the same sequence yields

$$\oplus\mathrm{SAT}_n^{\mathcal{A},L_{\leq m}} \to +_k\mathrm{ASAT}^{\mathcal{O},L_{\leq m}},$$

for some $k \in O(\log n)$.

Now, Lemma 10 says that $+_k\mathrm{ASAT}^{\mathcal{O},L_{\leq m}}$ is reducible, via an interactive protocol that has access to $\mathcal{A}$, to the conjunction of at most m claims regarding $\widehat{L}_i^k$, $i \leq m$, and therefore to the conjunction of at most mk claims regarding the Boolean version of $\widehat{L}_i^k$. Altogether these claims can be expressed as one Boolean formula, hence as one $\oplus\mathrm{SAT}$ instance, of size $\mathrm{poly}(mk)$, over the standard basis extended with the affine extension of $L_{\leq m}$. In notation,

$$+_k\mathrm{ASAT}^{\mathcal{O},L_{\leq m}} \to \oplus\mathrm{SAT}_{\mathrm{poly}(mk)}^{\widehat{L}_{\leq m}}$$

where we momentarily overload the notation $\widehat{L}_{\leq m}$ to denote the affine extension of $L_{\leq m}$.

Finally, Corollary 11 says, with the setting $\mathcal{O} = L_{\leq m}$, that

$$\oplus\mathrm{SAT}^{\widehat{L}_{\leq m}} \to \oplus\mathrm{SAT}^{L_{\leq m}}$$

and the theorem follows. □

Remark — what is non-relativizing here? Fortnow and Sipser [17] show a language $\mathcal{O}$ relative to which $\mathrm{coNP} \subset \mathrm{IP}$ fails, and it is easy to extend their argument to $\oplus\mathrm{SAT} \subset \mathrm{IP}$. So at least Theorem 2 does not relativize. Yet, it seems that everything leading up to Theorem 2 above relativizes — Propositions 4-5, Lemmas 8-10, Corollary 11 — and that the theorem follows by just putting these ingredients together via inference. Since relativizing statements are closed under inference, it seems Theorem 2 does relativize.

Before clarifying, let us probe further. Can we not consider *every* true statement as relativizing, by simply removing all occurrences of "the standard Boolean basis" and explicitly listing all the elements?

In either case, the key point to remember is that we are really interested in a particular way of formalizing statements. We especially agree (Section 1.1) to define P in such a way that in a theorem of the form "ψ, and this holds when the Boolean basis extended with $\mathcal{O}$ [or $\mathcal{A}$]" it is superfluous to add "and the algorithms are given access to $\mathcal{O}$ [or $\mathcal{A}$, resp.]". If a statement does not follow this convention, then we are not really interested in whether it relativizes.

Now notice that Lemma 10 does not follow this convention: the basis is extended with $\mathcal{O}$, but the algorithm is given access to $\mathcal{A}$. We can restate this lemma so that it complies with the convention, and argue that in its revised form it does not relativize. But we prefer its current form for streamlining the exposition.

The quickest way to settle the issue, given the exposition, is to mentally replace Lemma 10 with its proof whenever it is invoked, namely in the proofs of Theorems 2-3, instead of viewing it as a separate result.

3.2 The IP Theorem

In this section we show that Shamir's IP theorem, PSPACE $\subset$ IP, admits an affinely relativizing proof. As a byproduct we obtain a new proof of this result; see Section 1.3 for an overview and comparison with previous proofs.

We begin by generalizing the self-reducibility notion to include oracles. Since an algorithm means to us a uniform-family of circuits (Section 1.2) we automatically have the following definition:

Definition 12 (Self-reducibility w.r.t. a basis)**.** Call a language $L := \{L_n\}_{n\in\mathbb{N}}$ *downward-self-reducible (d-s-r) with respect to the Boolean basis* B (standard by default) if there is a uniform family $C := \{C_n\}_{n\in\mathbb{N}}$ of $\mathrm{poly}(n)$-size circuits over the basis B extended with L, such that for every $n \geq 0$: (i) C_n computes L_n, and (ii) the occurrences of L-gates in C_n, if there are any, are for $L_{<n}$ only.

The proof of Shamir's theorem is a straightforward consequence of the results in Section 3.1, on checking and compressing $\oplus$SAT. We show:

Theorem 13. *Every downward-self-reducible language is computable by an interactive protocol.*

This also holds if the standard Boolean basis is extended with $\mathcal{A}$ and protocols are given access to $\mathcal{A}$.

Proof. Extend the standard Boolean basis with $\mathcal{A}$. Let $L := \{L_n\}_{n\in\mathbb{N}}$ be downward-self-reducible (with respect to the extended basis). Then for every n, the circuit C_n computing L_n can be (uniformly) expressed as the sum $\oplus_y \phi(x, y)$, where ϕ is the formula verifying that y describes the computation of C on the input x. So there is a reduction that yields for a large enough constant d

$$L_n \to (\oplus\mathrm{SAT}^{L_{n-1}})_{n^d}, \tag{2}$$

every $n \in \mathbb{N}$. Here and throughout the rest of the proof, $\oplus\mathrm{SAT}^{L_{-1}}$ denotes $\oplus$SAT, L_i denotes $L_{\leq i}$, and n^i denotes $n^i + i$.

Combining (2) with Proposition 5, then applying Proposition 4, and then Theorem 3, we get a sequence of reductions such that for every n,

$$(\oplus\mathrm{SAT}^{L_{n-1}})_{n^d} \to (\oplus\mathrm{SAT}^{\oplus\mathrm{SAT}^{L_{n-2}}})_{n^{d'}} \to (\oplus\mathrm{SAT}^{L_{n-2}})_{n^{d''}} \to (\oplus\mathrm{SAT}^{L_{n-2}})_{n^d}, \tag{3}$$

where d, d', d'' are large enough constants (in particular d must exceed the exponent hidden in the $\mathrm{poly}(\cdot)$ notation of Theorem 3).

Now consider the reduction that on input x to L_n, first applies the reduction in (2), and then for n iterations, applies the reduction sequence in (3). This compound reduction yields

$$L_n \to \oplus\mathrm{SAT},$$

for every $n \in \mathbb{N}$, in other words, it yields $L \to \oplus$SAT.

By Theorem 2 on checking $\oplus$SAT, it follows that L is computable by a protocol with access to $\mathcal{A}$. □

By the result of Stockmeyer and Meyer [28], PSPACE has a complete language that is d-s-r, and the same holds for PSPACE^f with respect to the f-extended basis, for every language f, in particular for $f = \mathcal{A}$. We have thus proved Shamir's theorem, which we now state as a "gap amplification" result, as discussed on page 11.

Definition 14 (γ-gap-IP)**.** Say that a language L is in γ-gap-IP iff there is an interactive protocol for L with completeness 1 and soundness $< 1 - \gamma$. Here γ can be any function from $\mathbb{N}$ to $[0, 1) \subset \mathbb{R}$.

The Stockmeyer-Meyer result can be viewed as saying PSPACE $\subset$ 0-gap-IP, and the latter class by its very definition has a complete language that is d-s-r (namely the language TQBF). Hence:

Corollary 15 (IP theorem)**.** 0-gap-IP $\subset \Omega(1)$-gap-IP. *This also holds if the standard Boolean basis is extended with $\mathcal{A}$ and protocols are given access to $\mathcal{A}$.*

3.3 The MIP Theorem

In this section we show that the NEXP $\subset$ MIP theorem of Babai, Fortnow, and Lund [7] (BFL) admits an affinely relativizing proof, if it is viewed as a gap amplification result as explained on page 11. We show:

Definition 16 (γ-gap-MIP)**.** Say that a language L is in γ-gap-MIP iff there is a multiple-prover interactive protocol for L with completeness 1 and soundness $< 1-\gamma$. Here γ can be any function from $\mathbb{N}$ to $[0,1) \subset \mathbb{R}$.

Theorem 17 (MIP theorem)**.** 0-gap-MIP $\subset \Omega(1)$-gap-MIP. *This also holds if the standard Boolean basis is extended with $\mathcal{A}$, and the protocols are given access to $\mathcal{A}$.*

The proof becomes a straightforward consequence of Section 3.2, that the IP theorem has an affinely relativizing proof, once two ingredients are introduced. First is a very useful characterization of MIP, and more generally of γ-gap-MIP, due to Fortnow, Rompel, and Sipser [14]. We paraphrase their result:

Fact 18. *$L \in$ MIP iff there is a language π such that $L \in \mathrm{IP}^{\pi}$, with a protocol that is robust to its oracle π in the following sense: if some other oracle π^* is used instead of π then no prover strategy can exploit this, i.e., the verifier cannot be convinced to accept x with probability $\geq 1/3$ whenever $L(x) = 0$, even if some other π^* is used as oracle instead of π.*

This equivalence more generally holds if "MIP" is replaced with "γ-gap-MIP", "IP" with "γ-gap-IP", and "$1/3$" with "$1-\gamma$".

The above also holds when all the protocols involved are given (additional) access to $\mathcal{O}$.

The second ingredient in proving Theorem 17, and the key one, is the seminal "multi-linearity test" of BFL [7]. We paraphrase again, by combining it with a "Booleanness test" from the same paper:

Fact 19. *There is an interactive protocol, $Decode$, satisfying the following: Given oracle access to a function π^*, and given inputs (X, ε), the protocol runs in time $\mathrm{poly}(N/\varepsilon)$ where $N = |X|$, and outputs:*

(i) $\pi^*(X)$, if π^* is the affine extension of some Boolean function on n bits,

(ii) $f(X)$, for some fixed affine extension f that is nearby π^*, if one exists nearby and if (i) fails

(iii) "fail", otherwise

where the outcome in case (i) happens with certainty and in cases (ii) - (iii) with probability $\geq 1-\varepsilon$, and where being nearby π^* means agreeing with π^* on some 0.99-fraction of inputs, and where the affine extensions are over $\mathbb{F}_{2^k}$ for some $k \geq c \log n$, with c being a universal constant.

With Facts 18-19 in hand, we are ready to derive Theorem 17. As mentioned, the proof will use the fact that Shamir's IP theorem affinely relativizes, so we begin with a preparatory remark about this. The way it is stated in Corollary 15, namely as 0-gap-IP $\subset$ IP, this fact is not strong enough for our purposes in this section. We need the following variant:

Corollary 20 (IP theorem, stronger version)**.** *For every 0-gap-IP protocol, there is a corresponding IP protocol that computes the same language. This also holds if the standard Boolean basis is extended with $\mathcal{A}$; in fact the corresponding protocol does not depend on the choice of $\mathcal{A}$.*

In contrast, Corollary 15 allows the corresponding protocol to depend on $\mathcal{A}$. To reinforce the difference, notice that it is not even true in general that an IP protocol remains one if its oracle is changed, because the gap condition can then be violated on some inputs. To be convinced of Corollary 20, one way is to skim back to Sections 3.1-3.2 and notice, in particular, that Theorems 2-3 regarding $\oplus$SAT assert that their claims carry over to an extended basis via the *same* protocols that work for the standard basis. (An alternate way is to observe that the IP protocol for the language TQBF is uniform in the choice of $\mathcal{A}$.)

Proof of Theorem 17. Let us make a notational convention, and use $[V(x)]$ to denote the maximum probability that a verifier V accepts its input x, over all possible prover strategies in a protocol.

Let $L \in 0\text{-gap-MIP}$. By Fact 18, $L \in 0\text{-gap-IP}^{\pi}$ for some language π; we may assume π is an affine extension since every language reduces to its affine extension. Pick any protocol realizing $L \in 0\text{-gap-IP}^{\pi}$, and let its verifier be V_0. We know that this protocol is robust to its oracle π in the sense of Fact 18.

By Corollary 15, $L \in \text{IP}^{\pi}$.[7] Therefore, by Fact 18, all that remains to show is that among the protocols realizing $L \in \text{IP}^{\pi}$, one is robust to π. So pick any protocol realizing $L \in \text{IP}^{\pi}$, with verifier V say. We may assume that the soundness error of this protocol is $< 1/6$ by amplification.

Consider modifying V, so that it performs each of its oracle queries, say to $\pi(X)$, via the protocol of Fact 23, as $Decode^{\pi}(X, \varepsilon)$, and rejects upon failure; the parameter ε will be determined later. By Fact 19-(i), this modification does not affect the outcome of the protocol when π is used as oracle, so we still have a IP^{π}-protocol for L.

We claim that this new protocol has the desired robustness; in the notation introduced up front,

$$[V \circ Dec^{\pi^*}(x)] < 1/3$$

for every language π^* and every x such that $L(x) = 0$, where $V \circ Dec$ denotes the modified verifier.

To prove this, let π^* be an arbitrary language and suppose $L(x) = 0$ for a given $x \in \{0,1\}^n$. For convenience, assume that on x, the verifier V makes oracle queries of one fixed length, so that the same holds for $V \circ Dec$ and we may focus on π^*_N for some N depending on x, instead of $\pi^*_{N_i}$ for various i.

The proof is trivial in case there is no affine extension nearby π^*_N in the sense of Fact 19. Indeed, we may assume that V always makes at least one oracle query, even if only to ignore the answer, so that $Decode$ is invoked at least once by $V \circ Dec$. Then $[V \circ Dec^{\pi^*}(x)] \leq \varepsilon$ by Fact 19-(iii).

So suppose there does exist an affine extension nearby π^*_N. Starting from the very first protocol for L, i.e. the one with the verifier V_0, and going toward the last one with the verifier $V \circ Dec$, we will now argue the validity of the implications

$$L(x) = 0 \implies [V_0^f(x)] < 1 \implies [V \circ Dec^f(x)] < 1/6 \implies [V \circ Dec^{\pi^*}(x)] < 1/3$$

for the affine extension f of some language, such that f_N is nearby π^*_N. This will prove the theorem.

The first two implications hold no matter what f is. Indeed, recall that V_0 is the verifier of a 0-gap-IP^{π}-protocol for L that is robust to π, and we already have the first implication.

For the second implication, we need the strength of Corollary 20. We know that V^{π} agrees with V_0^{π} on x by Corollary 15, but using Corollary 20 we can also say that V^f agrees with V_0^f on x, and hence accepts x with probability $< 1/6$. The same holds for $V \circ Dec^f$ in place of V^f, by Fact 19-(i).

For the last implication, let $t(n)$ be the running time of V on x. Because π^*_N is nearby an affine extension, Facts 19-(i)-(ii) imply that there is some fixed affine extension f_N for which $V \circ Dec^{f_N}$ behaves identical to $V \circ Dec^{\pi^*_N}$ on x except with probability $\leq t\varepsilon$. So set $\varepsilon = 1/(6t)$, and complete f_N to a language f that is itself an affine extension — a trivial thing to do as f can just be zero on the rest of the inputs. □

[7] When the Boolean basis is extended with $\mathcal{A}$, notice that 0-gap-IP^{π} really involves two affine oracles not just one. We are still justified in invoking Corollary 15, however; see the discussion in page 5 titled "multiple oracles".

3.3.1 Comparison with the standard view

We now make precise the discussion on page 11, that relates the gap amplification view of the MIP theorem, Theorem 17, to the standard view, $\mathrm{NEXP} \subset \mathrm{MIP}$:

Proposition 21. $\mathrm{NEXP} \subset 0\text{-gap-MIP}$. *This does* not always *hold if the standard Boolean basis is extended with $\mathcal{A}$.*

In order to make transparent what "current technique" yields it, we prove Proposition 21 in two steps. First, we characterize 0-gap-MIP as a subclass of NEXP, namely as those languages in NEXP with "strong locality". Then we show that the strong Cook-Levin theorem collapses NEXP to that subclass.

Definition 22 (strong uniformity)**.** Call a family of circuits $\{C_n\}_{n\in\mathbb{N}}$ strongly uniform iff the function that outputs, given input (n, i), the type of the i^{th} gate in C_n, as well as the indices of all gates connected to it, is in FP.

Definition 23 ($\mathrm{NEXP}_{\mathrm{local}}$)**.** Let $\mathrm{P}_{\mathrm{local}}$ be the class of all languages with strongly uniform polynomial-size circuits. Using this class define $\mathrm{NP}_{\mathrm{local}}$, and then by padding define $\mathrm{NEXP}_{\mathrm{local}}$; in other words let $\mathrm{NEXP}_{\mathrm{local}}$ be the class of all languages with strongly uniform exponential-size nondeterministic circuits.

Proposition 24. $0\text{-gap-MIP} = \mathrm{NEXP}_{\mathrm{local}}$. *This also holds if the standard Boolean basis is extended with $\mathcal{O}$, and the protocols are given access to $\mathcal{O}$.*

Remark. Over an arbitrary basis extension $\mathcal{O}$, the class $\mathrm{NEXP}_{\mathrm{local}}$ corresponds exactly to what we may denote as $\mathcal{NEXP}^{\mathcal{O}[\mathrm{poly}]}$, the Turing-machine-based definition of relativized NEXP where the oracle queries are restricted to be of polynomial length. In logical terms (Section 1.1), in every standard model of $\mathcal{ACT}$, the variable $\mathrm{NEXP}_{\mathrm{local}}$ gets interpreted as the class $\mathcal{NEXP}^{\mathcal{O}[\mathrm{poly}]}$ for some $\mathcal{O}$, and conversely for every $\mathcal{O}$, there is some standard model of $\mathcal{ACT}$ in which $\mathrm{NEXP}_{\mathrm{local}}$ is interpreted as $\mathcal{NEXP}^{\mathcal{O}[\mathrm{poly}]}$. Proposition 24 thus vindicates the use of poly-length query restriction in previous work; see the discussion on page 11.

We now proceed to prove Propositions 24 & 21.

Proof of Proposition 24. Part ($\subset$): Extend the standard Boolean basis with $\mathcal{O}$. Let $L \in 0\text{-gap-MIP}$. By Fact 18, $L \in 0\text{-gap-IP}^{\pi}$ for some language π. In other words, there is a uniform family of size-$\mathrm{poly}\, n$ circuits $V^{\pi} := \{V_n^{\pi}(x, r)\}_n$ over the basis extended with π (this extension is in addition to $\mathcal{O}$) such that $L(x) = 1$ iff

$$\bigwedge_{r_1} \bigvee_{r_2} \cdots \bigvee_{r_{|r|}} V_n^{\pi}(x, r)$$

evaluates to 1, where r ranges over length-$\mathrm{poly}\, n$ strings (wlog $|r|$ is even) and $n = |x|$. View this expression as a circuit $D_n^{\pi}(x)$, of size $O(2^{|r|})$ times the size of V_n^{π}, and then as a circuit $C_n(x, y_{\pi})$ over the *standard* basis, by replacing each π-gate in D with a y_{π}-gate (we still have $\mathcal{O}$ in the basis). Since V is a uniform family, it follows that $C := \{C_n(x, y_{\pi})\}_n$ is a strongly-uniform family. This proves the claim.

Part ($\supset$): Extend the standard Boolean basis with $\mathcal{O}$. Let $L \in \mathrm{NEXP}_{\mathrm{local}}$ with a corresponding strongly uniform circuit family $\{C_n(x, y)\}_n$ of size $= s(n) \in \exp \mathrm{poly}\, n$. For every $x \in \{0,1\}^n$ and $i \in \{0,1\}^{\log s(n)}$, let $\pi(x, i)$ be the value of the i^{th} gate in $C_n(x, y^*)$ for some fixed y^* maximizing the output of $C_n(x, y)$ over all eligible y. Then in the protocol for $L(x)$, the verifier $V(x)$ simply: (i) picks at random $i \leftarrow \{0,1\}^{\log s(n)}$; (ii) using the strong uniformity of $\{C_n\}$, finds out that the i^{th} gate is, say, of type f and is connected, say, to gates $i_1..i_m$ in that order; and (iii) checks that the transcript is consistent with the i^{th} gate, i.e., that $z = f(z_1..z_m)$, where z stands for $\pi(x, i)$ in general, with the special case being when gate

i is the output gate (then $z = 1$), and where z_k stands for $\pi(x, i_k)$ in general, with the special case being when gate i_k is an input gate (then $z_k = x_j$ for an appropriate j). It is easy to see the robustness of this protocol to π. The claim follows. □

Proof of Proposition 21. $\mathrm{P} \subset \mathrm{P}_{\mathrm{local}}$ by the strong Cook-Levin theorem [13], implying $\mathrm{NEXP} \subset \mathrm{NEXP}_{\mathrm{local}}$ and proving the first claim via Proposition 24.

For the second claim, we want to show a language $\mathcal{O}$ with its affine extension $\mathcal{A}$, such that $\mathrm{NEXP} \not\subset \mathrm{NEXP}_{\mathrm{local}}$ relative to $\mathcal{A}$. It actually suffices to show this non-containment relative to $\mathcal{O}$ instead of $\mathcal{A}$. To see why, recall that $\mathcal{A}$ reduces to $\oplus\mathrm{SAT}^{\mathcal{O}}$ (Corollary 11), and notice $\oplus\mathrm{SAT} \in \mathrm{EXP}_{\mathrm{local}}$ as a relativizing fact; therefore neither NEXP nor $\mathrm{NEXP}_{\mathrm{local}}$ changes if $\mathcal{O}$ is used as the basis extension instead of $\mathcal{A}$.

The claim now follows by a standard diagonalization-style construct that exploits the constraint on the length of the $\mathcal{O}$-queries in $\mathrm{NEXP}_{\mathrm{local}}$. □

3.4 Lower Bounds against General Boolean Circuits

Using the IP theorem and its variants, Buhrman, Fortnow, Thireauf [11] and Santhanam [23] succeeded in obtaining the strongest lower bounds known-to-date against general Boolean circuits. In both results, the lower bound is shown for the class of Merlin-Arthur protocols; in the case of Buhrman et al. it is for the class $\mathrm{MA}(\exp n)$, of protocols running in exponential time, and in Santhanam it is for $\mathrm{MA}(\mathrm{poly}\, n)$. For notational convenience, we use $\mathrm{MA}(t(n))$ to denote classes of partial languages, and make it explicit when we talk about the subclass of total languages.

In this section we give an affinely relativizing proof that unifies both results. We prove:

Theorem 25. *For every constant d,*

(i) $\mathrm{MA}(\exp n)$ *contains a language that does not have circuits of size* $O(2^{\log^d n})$.

(ii) $\mathrm{MA}(\mathrm{poly}\, n)$ *contains a partial language that does not have circuits of size* $O(n^d)$.

This also holds if the Boolean basis is extended with $\mathcal{A}$, and protocols are given access to $\mathcal{A}$.

The proof consists of three main ingredients. The first one shows that if the lower bound fails to hold, then this failure scales to $\oplus\mathrm{SAT}$.

Lemma 26 (Scaling)**.**

(i) If part (i) of Theorem 25 is false, then $\oplus\mathrm{SAT}$ *has circuits of size* $O(2^{\log^d n})$ *for some d.*

(ii) If part (ii) of Theorem 25 is false, then $\oplus\mathrm{SAT}$ *has circuits of size* $O(n^d)$ *for some d.*

This also holds if the Boolean basis is extended with $\mathcal{A}$.

We defer the proof of this lemma to the end of this section.

To proceed with the rest of the proof it will be convenient to introduce some notation.

Definition 27 ($\Sigma_3\mathrm{SAT}$)**.** Let $\Sigma_3\mathrm{SAT}$ denote the language mapping $\phi(x, y, z) \mapsto \exists x \forall y \exists z\, \phi(x, y, z)$ where ϕ is over the standard Boolean basis by default.

Let $\Sigma_3\mathrm{SAT}(t(n))$ denote the set of all languages Karp-reducible in time $t(n)$ to $\Sigma_3\mathrm{SAT}$, where $t(n)$ is a well-behaved resource bound.

The second ingredient in proving Theorem 25 is a collapse result: if the conclusion of the Scaling lemma holds, then the polynomial-time hierarchy collapses. We defer its proof to the end of this section.

Lemma 28 (Collapse)**.** *Let s be a well-behaved resource bound.*

If $\oplus\mathrm{SAT}$ has circuits of size $O(s(n))$, then $\Sigma_3\mathrm{SAT}$ is computable by a protocol in $\mathrm{MA}(s(\mathrm{poly}\, n))$.

This also holds if the Boolean basis is extended with $\mathcal{A}$ and the protocol is given access to $\mathcal{A}$.

The last ingredient of the proof is a classical result of Kannan [20], showing circuit lower bounds for $\Sigma_3\mathrm{SAT}$, and more generally for $\Sigma_3\mathrm{SAT}(t)$. His proof relativizes.

Fact 29 (Kannan's bound)**.** *Let s be a well-behaved resource bound.*

$\Sigma_3\mathrm{SAT}(\mathrm{poly}\, s(n))$ contains a language that does not have circuits of size $O(s(n))$.

This also holds if the Boolean basis is extended with $\mathcal{O}$.

With the three ingredients in hand — Scaling and Collapse lemmas, and Kannan's bound — we can prove Theorem 25. For part (i), we let $\mathcal{C}$ be the set of languages in $\mathrm{MA}(\exp n)$ and put $s(n) = 2^{\log^d n}$; for part (ii), we let $\mathcal{C}$ be the set of partial languages in $\mathrm{MA}(\mathrm{poly}\, n)$ and put $s(n) = n^d$.

The proof goes by contradiction. We give the argument using notation.

$$
\begin{aligned}
&\mathcal{C} \subset \mathrm{SIZE}(O(s(n))) \\
&\implies \oplus\mathrm{SAT} \in \mathrm{SIZE}(O(s(n))) && \text{(by Scaling lemma)} \\
&\implies \Sigma_3\mathrm{SAT} \in \mathrm{MA}(s(\mathrm{poly}\, n)) && \text{(by Collapse lemma)} \\
&\implies \Sigma_3\mathrm{SAT}(\mathrm{poly}\, s(n)) \in \mathcal{C} && \text{(*)} \\
&\implies \qquad \text{contradiction} && \text{(by Kannan's bound)}
\end{aligned}
$$

where step (*) follows from Definition 27 and the fact that $s(\mathrm{poly}\, s(n)) \subset \mathrm{poly}\, s(n)$ for the particular choices of $s(n)$.

What remains is the proof of Scaling and Collapse lemmas.

Proof of Scaling Lemma . There is nothing to prove in part (i), because a $\oplus\mathrm{SAT}$ instance of size n is computable by brute force in deterministic time $\exp n \cdot \mathrm{poly}\, n$, which by definition is a protocol in $\mathrm{MA}(\exp n)$.

For part (ii), we want to show that $\oplus\mathrm{SAT}$ has circuits of size $\mathrm{poly}\, n$, if every partial language in $\mathrm{MA}(\mathrm{poly}\, n)$ has circuits of size $O(n^d)$ for some fixed d. By Theorem 2, $\oplus\mathrm{SAT}$ reduces to some same-length checkable language K, so it suffices to show the claim for K instead of $\oplus\mathrm{SAT}$.

So let K be *any* same-length checkable language, and suppose towards a contradiction that K does not have polynomial-size circuits. Let $s : \mathbb{N} \to \mathbb{N}$ be such that $s(n)$ is the size of the smallest circuit deciding K on inputs of length n, for every n. By assumption, $s(n)$ is super-polynomial, i.e., $s(n) >_{\text{i.o.}} n^k$ for every constant k. Note that $s(n)$ might not be well-behaved.

Consider the partial language $K'(xy) := K(x)$ that is defined only on inputs of the form xy where $y \in 01^*$ serves as a pad of length $|y| = \lfloor s(|x|)^\epsilon \rfloor$, for some constant $\epsilon > 0$ to be later determined.

Now consider the following MA-protocol for K': given xy, the prover sends the smallest circuit for K on inputs of length $|x|$, i.e. a circuit of size $s(|x|)$, and the verifier uses the same-length checkability of K to compute $K(x)$, hence $K'(xy)$. This takes, on an input of length $|x|+|y|$, time $\mathrm{poly}\, s(|x|) \subset \mathrm{poly}\, s(|x|)^\epsilon \subset \mathrm{poly}(|x| + |y|)$. So K' is in $\mathrm{MA}(\mathrm{poly}\, n)$, and hence has circuits of size $O(n^d)$ by assumption. But then K has circuits of size $O(n + s(n)^\varepsilon)^d$, which is less than $s(n)$ for infinitely many n whenever $\varepsilon < 1/d$ because $s(n)$ is superpolynomial. But this contradicts $s(n)$ being the smallest circuit size for K. □

Proof of Collapse Lemma. Toda famously showed [29]

$$\Sigma_3\mathrm{SAT} \to \oplus\mathrm{SAT}$$

via a randomized reduction that is universal in the choice of a Boolean basis. (The same holds in general for Σ_kSAT for all constant k.) So if $\oplus$SAT has circuits of size $O(s(n))$ for formulas of size n, then the MA-protocol for computing Σ_3SAT, on a formula of size n, proceeds by the verifier doing the above reduction to obtain a formula of size $m \in \mathrm{poly}\, n$, then the prover sending a circuit for $\oplus$SAT at a large enough input length $\mathrm{poly}\, m$, hence a circuit of size $O(s(\mathrm{poly}\, n))$, and finally, the verifier running the checker for $\oplus\mathrm{SAT}_m$ (Theorem 2) on the circuit, in time $\mathrm{poly}(s(\mathrm{poly}\, n))$, i.e. in time $s(\mathrm{poly}\, n)$ since s is well-behaved. □

3.5 The ZKIP Theorem

AW made the surprising observation that the famous theorem of Goldreich, Micali, and Wigderson, NP $\subset$ ZKIP if one-way-functions exist [18], can be proven via the same techniques underlying the IP theorem. This is in contrast to the standard proof of this result involving a graph-based construction, which seems incompatible with the oracle concept.

IKK turned this idea into a complete proof by devising an indirect commitment scheme for this purpose. In this section we explain how this AW-IKK proof can be adapted to our framework, yielding an affinely relativizing proof of the ZKIP theorem.

Theorem 30 (ZKIP theorem)**.** $\mathrm{NP} \subset \mathrm{ZKIP}$ *if there is a one-way function in* P *secure against* BPP.
This also holds if the standard Boolean basis is extended with $\mathcal{A}$*.*

The key idea of AW is that, given a circuit C, and a satisfying assignment x of inputs to C, in order to show that C is satisfiable without leaking x, an efficient prover commits to a transcript of the computation of $C(x)$, but redundantly so, as follows. For each fragment of the circuit of the form

$$z_0 = f(z_1,..,z_n), \tag{4}$$

meaning gate z_0 is of type f and receives its inputs from gates $z_1,..,z_n$ in that order (where some z_i may refer to an input gate x_j), the prover commits not to the values of $z_0, z_1,..,z_n$, but to

$$\widehat{f}\circ\ell, \widehat{f}\circ\ell(1), \ldots, \widehat{f}\circ\ell(n) \tag{5}$$

where (i) $1..m$ are distinct non-zero elements in a large enough field $\mathbb{F}$, (ii) each $\ell(i)$ is a point on the random line ℓ passing through $\ell(0) := (z_1,..,z_n) \in \mathbb{F}^n$, and (iii) $\widehat{f}\circ\ell$ is the univariate polynomial of degree n, i.e. the coefficients thereof, corresponding to the restriction of $\widehat{f}$ to ℓ.

In order to check the fragment (4), it suffices to check that the information given in (5) consistently refers to the same polynomial. This consistency check amounts to a system of linear equalities, for which IKK devised a scheme of indirect commitments, thereby obtained a zero-knowledge protocol for circuit satisfiability (assuming one-way functions exist).

The make this proof work in our setting, we need to address the case where the f-gate in the fragment (4) is a non-standard basis element. This is not an issue in the setting of AW or of IKK, because there by definition $\widehat{f}$ is efficiently computable given access to f, as explained in Section 1.2. In our case, however, if f is a non-standard basis element, then f is an affine extension of some $g : \{0,1\}^m \to \{0,1\}$, hence is of the form $\mathrm{bool}(\widehat{g})$, which makes $\widehat{f}$ seem not efficiently computable given access to f.

Our solution to this, is to view f not as an atomic gate, but instead as a small circuit, consisting of a $\widehat{g}$-gate plus the circuitry for computing its decision version, bool. So in the beginning of the protocol, both parties already expect to give/receive a transcript involving $\widehat{g}$-gates as well as the standard gates. For fragments involving standard gates f, the commitment involves their extension $\widehat{f}$ as above, whereas for $\widehat{g}$-gates, no further extension is made and the commitment involves still $\widehat{g}$. We defer the details to the full version.

4 Negative Relativization Results

This section shows that several major conjectures in structural complexity are impossible to settle via an affinely relativizing proof, mirroring corresponding results of AW.

There are two main approaches to deriving such results: an interpolation approach, used for separations of the form $\mathcal{C} \not\subset \mathcal{D}$, and an approach based on communication complexity, used for containments $\mathcal{C} \subset \mathcal{D}$. Both of these approaches are model theoretic, in the sense that they construct an eligible language relative to which the statement in question is false.

The main novelty in this section, as explained in Section 1.3, is in the development of the interpolation approach, which is then used to show that $\mathrm{NEXP} \not\subset \mathrm{P/poly}$ is affinely non-relativizing. This is carried out in Section 4.1. The communication complexity approach is taken in Section 4.2.

Besides these two approaches there is a third, proof theoretically flavored approach, that is quite convenient to use when the situation allows. To show that ψ admits no proof that affinely relativizes, we find a statement ψ' for which this is already known, and then derive the implication $\psi \implies \psi'$ via an affinely relativizing proof. We thus show that ψ' is "no harder" to prove than ψ, in similar spirit to the use of reductions in structural complexity. It should be noted that in general, this approach cannot be used for the AW notion of algebrizing proofs, as it critically relies on the closure of such proofs under inference. Section 4.3 employs this approach.

In preparation for the rest of this section, let us introduce a piece of notation:

Definition 31 ($\widetilde{f}$). Given $f_m : \{0,1\}^m \to \{0,1\}$, denote its affine extension to $\mathbb{F}_{2^k}$, i.e. the Boolean version of $\widehat{f}_m^k$, with $\widetilde{f}_m^k$.

Given the language $f := \{f_m\}_{m\in\mathbb{N}}$, denote its affine extension, i.e. the family $\{\widetilde{f}_m^k\}_{k,m\in\mathbb{N}}$, with $\widetilde{f}$.

4.1 Interpolation Approach

The classical way to show that $\mathcal{C} \not\subset \mathcal{D}$ does not admit a relativizing proof is to construct a language $\mathcal{O}$ relative to which $\mathcal{C} \subset \mathcal{D}$ holds. Such a construction amounts to a balancing act of sorts; the goal, vaguely, is to have $\mathcal{O}$ give more power to $\mathcal{D}$ than it does to $\mathcal{C}$, so as to make $\mathcal{D}$ contain $\mathcal{C}$ in the $\mathcal{O}$-extended basis. This can be done nonetheless, and sometimes easily so, as can be seen by taking $\mathcal{C} = \mathrm{PSPACE}$, $\mathcal{D} = \mathrm{P}$, and $\mathcal{O}$ to be any PSPACE-complete language. Typically, however, the construction is more involved, and it was one of the main contributions of AW to develop an approach — the interpolation approach — that enables such constructions in the algebrization framework.

In this section we develop the interpolation approach within our framework. Our techniques are different from AW's; see Section 1.3 for a comparison. Below we develop the crux of the approach, and in Section 4.1.1 we complete the development by applying it to the NEXP vs. $\mathrm{P/poly}$ question.

Before we proceed let us note, like AW did, that the easy fact regarding PSPACE and P mentioned above carries over to our setting easily:

Proposition 32. $\mathrm{PSPACE} \not\subset \mathrm{P}$ *cannot be derived via an affinely relativizing proof.*

Proof. Relative to any PSPACE-complete language $\mathcal{O}$, it is well-known that $\mathrm{PSPACE} \subset \mathrm{P}$. This containment also holds relative to the affine extension $\mathcal{A}$ of that very $\mathcal{O}$. To see why, recall that $\mathcal{A}$ reduces to $\oplus\mathrm{SAT}^{\mathcal{O}}$ (Corollary 11), and notice $\oplus\mathrm{SAT} \in \mathrm{PSPACE}$ as a relativizing fact; therefore PSPACE does not change if $\mathcal{A}$ is used is used as the basis extension instead of $\mathcal{O}$. □

We now move to the interpolation approach. The crux of our development is two coding-theoretic ingredients. The first one states that knowing t bits of a binary codeword exposes at most t bits of its information word, and the second scales this result to affine extensions.

Lemma 33 (Interpolation)**.** *Let $\mathcal{E} : \mathbb{F}_2^K \to \mathbb{F}_2^N$ be linear and injective. Given a "dataword" $u \in \mathbb{F}_2^K$ and a set of indices $A \subseteq [N]$, consider the collection U of all datawords $u' \in \mathbb{F}_2^K$ such that $\mathcal{E}(u)$ and $\mathcal{E}(u')$ agree on A.*

There is a set of indices $B \subseteq [K]$, no larger than A, such that projecting U onto $G := [K] \setminus B$ gives all of $\mathbb{F}_2^G$.

Proof. The claim of the lemma on U is true iff it is true on $U^+ := U + u$. So it suffices to show that U^+ is a subspace of $\mathbb{F}_2^K$ with dimension at least $K - |A|$.

Now, $y \in U^+$ iff $y + u \in U$, which is iff $\mathcal{E}(y + u)$ and $\mathcal{E}(u)$ agree on A, which is iff $\mathcal{E}(y)$ vanishes on A. Therefore U^+ is identical to the space of all datawords whose encodings vanish on A.

All that is left is to bound $\dim U^+$, or equivalently, to bound $\dim \mathcal{E}(U^+)$ since $\mathcal{E}$ is injective. The latter quantity is the dimension of the space $\mathcal{C} \cap \mathcal{Z}$, where $\mathcal{C}$ is the image of $\mathcal{E}$, and $\mathcal{Z}$ is the space of all N-bit vectors that vanish on A. But then by the theorem on the dimension of a sum of subspaces (e.g., [4, Theorem 1.4])

$$\begin{aligned}\dim(U^+) &= \dim(\mathcal{Z}) \quad + \dim(\mathcal{C}) - \dim(\mathcal{Z} + \mathcal{C}) \\ &= (N - |A|) + \quad K \quad - \dim(\mathcal{Z} + \mathcal{C})\end{aligned}$$

which is at least $K - |A|$ because $\mathcal{Z} + \mathcal{C} \subseteq \mathbb{F}_2^N$. This finishes the proof. □

Theorem 34 (Interpolation)**.** *Given a language f and a finite set A of inputs, consider the collection $\mathcal{F}$ of all languages g such that $\widetilde{f}$ and $\widetilde{g}$ agree on A.*

There is a set B of inputs, no larger than A, such that every partial Boolean function g' defined outside B can be extended to some $g \in \mathcal{F}$.

Further, in extending g' to g, the values of g at length-n inputs depend only on those of g' at length n.

Proof. To begin with, consider the special case where $A \subseteq \mathrm{dom}(\widetilde{f}_m^k)$ for some fixed k and m. For the purpose of invoking Lemma 33, let $\mathcal{E}$ be the map that takes as input the truth table of a Boolean function g_m on m bits, and outputs the truth table of $\widetilde{g}_m^k$. So $\mathcal{E} : \mathbb{F}_2^K \to \mathbb{F}_2^N$, where $K = 2^m$ and $N = k2^{km}$ (to see the value of N, recall that $\widetilde{g}_m^k(\alpha, y)$ gives the y^{th} bit of $\widehat{g}_m^k(\alpha)$, where $\widehat{g}_m^k$ is the extension of g_m to $\mathbb{F}_{2^k}^m$).

Clearly $\mathcal{E}$ is injective; it is also linear because we represent $\mathbb{F}_{2^k}$ with $\mathbb{F}_2^k$ where addition is componentwise (Section 2). So $\mathcal{E}$ fulfils the conditions of Lemma 33, which in turn yields a set $B \subseteq \{0,1\}^m$ that is no larger than A, such that every partial Boolean function on $\{0,1\}^m \setminus B$ can be extended to a language in $\mathcal{F}$. This proves the theorem in the special case.

To handle the general case, partition A into $A_{m,k} := A \cap \mathrm{dom}(\widetilde{f}_m^k)$, and use the above special case as a building block to create a bigger code. In detail, for every m involved in the partition, define $\mathcal{E}_m$ as the map sending the truth table of g_m to the list comprising the truth tables of $\widetilde{g}_m^{k_1}, \widetilde{g}_m^{k_2}, \ldots$ for every A_{m,k_j} in the partition. Now, take each $\mathcal{E}_m$ thus obtained, and let $\mathcal{E}$ be their product. In other words, let $\mathcal{E}$ take as input a list $T_{m_1}, T_{m_2}, ..$ where T_{m_i} is the truth table of some Boolean function g_{m_i} on m_i bits, and outputs $\mathcal{E}_{m_1}(T_{m_1}), \mathcal{E}_{m_2}(T_{m_2}), .. .$ The theorem now follows from Lemma 33. □

4.1.1 Application — NEXP vs. P/poly

The Interpolation theorem enables us to adapt some of the classical constructions from relativization to affine relativization. The general idea is to construct a language $\mathcal{O}$ such that $\mathcal{C} \subset \mathcal{D}$ holds relative to $\mathcal{O}$, i.e.,

such that $\mathcal{C}^{\mathcal{O}} \subset \mathcal{D}^{\mathcal{O}}$, by taking each algorithm underlying $\mathcal{C}^{\mathcal{O}}$, say the first n algorithms, and by fixing $\mathcal{O}$ up to a certain length, say $m(n)$, so as to force the behavior of these algorithms on inputs of length n. While doing so, the goal is for $\mathcal{O}$ to encode those forced behaviors, in a way that can be easily queried by some algorithm in $\mathcal{D}^{\mathcal{O}}$.

This classical idea can be extended to our setting via the Interpolation theorem. Even though the goal here is to have $\mathcal{C} \subset \mathcal{D}$ hold not relative to $\mathcal{O}$, but relative to its affine extension $\mathcal{A}$, we can proceed almost as before. This is because thanks to the Interpolation theorem, forcing the behavior of a $\mathcal{C}^{\mathcal{A}}$ algorithm by fixing $\mathcal{A}$ bears little extra burden on $\mathcal{O}$ than does fixing $\mathcal{O}$ to force a $\mathcal{C}^{\mathcal{O}}$ algorithm (though there are some subtleties).

For details we refer to the proof of the next theorem, which is the main result of this section:

Theorem 35. $\mathrm{NEXP} \not\subset \mathrm{P/poly}$ *cannot be derived via an affinely relativizing proof.*

Proof. It is a basic fact that NEXP has polynomial-size circuits iff NE (the linear-exponential version of NEXP) has circuits of size a ***fixed*** polynomial, and that this relativizes. In notation,

$$\mathrm{NEXP}^{\mathcal{O}} \subset \mathrm{SIZE}^{\mathcal{O}}(\mathrm{poly}\, n) \iff \mathrm{NE}^{\mathcal{O}} \subset \mathrm{SIZE}^{\mathcal{O}}(n^d) \text{ for some } d \in \mathbb{N}.$$

Therefore, to prove Theorem 35, it suffices to show a language f satisfying

$$\mathrm{NE}^{\widetilde{f}} \subset \mathrm{SIZE}^{f}(n^d), \tag{6}$$

for some constant d because f trivially reduces to $\widetilde{f}$.

So let $M_0, M_1,..$ be a list (repetitions allowed) of all nondeterministic algorithms with access to an arbitrary language $\mathcal{O}$, such that M_i runs in time $\leq 2^{n \log n}$ on all inputs of length $n > i$. We construct the language f in such a way that when $\mathcal{O} = \widetilde{f}$, the information regarding how each M_i behaves on each large enough input x, is stored by f in a format retrievable by a small circuit. More precisely, we ensure that for every $n > 1$, a size-n^d circuit with access to f can compute the function $L_n : \{0,1\}^{\lfloor \log n \rfloor} \times \{0,1\}^n \to \{0,1\}$ defined as

$$L_n(i, x) := M_i^{\widetilde{f}}(x). \tag{7}$$

This yields (6), hence the theorem, because each language $K \in \mathrm{NE}^{\widetilde{f}}$ corresponds to some $M_i^{\widetilde{f}}$, and in order to compute $K(x)$ on all but finitely many inputs x (in particular for $x \in \{0,1\}^{>2i}$) we can just provide (i, x) to the circuit C^f said to compute $L_{|x|}$, implying $K \in \mathrm{SIZE}^{f}(n^d)$.

We construct f inductively, as the limit of a sequence $f_1, f_2,..$ of Boolean functions where f_n extends f_{n-1}. The domain of f_n will include all of $\{0,1\}^{\leq n^d}$, plus some additional $2^{4n \log n}$ strings at most. Set $f_1 : \{0,1\} \to \{0\}$.

At iteration $n > 1$, proceed to set f_n as follows. Consider all possible ways of extending f_{n-1} to a language f. Out of all such f, pick one that maximizes (7), i.e., one for which the collection

$$\mathcal{S}_f := \{(i, x) : L_n(i, x) = 1\} \tag{8}$$

of accepting algorithm-input pairs is maximal.

Now we want to "open up space" in f by un-defining it at some inputs, the idea being then to encode the function in (7) in the freed space so that a small circuit can look it up. In doing so, of course, we do not want to disturb (7), which, by the way we picked f, is equivalent to wanting that $\mathcal{S}_f$ does not shrink — i.e., as we restrict f to some f', no matter how we extend f' back to some language g, we want $\mathcal{S}_g = \mathcal{S}_f$.

Consider an accepting algorithm-input pair (i, x) in $\mathcal{S}_f$. Because M_i runs in nondeterministic $2^{n \log n}$-time on input $x \in \{0,1\}^n$, it could issue a great many oracle queries to $\widetilde{f}$, however, as far as the membership

of (i,x) in $\mathcal{S}_f$ is concerned, it suffices for $\widetilde{f}$ to honor only those queries of M_i along *one accepting* computation path. So each such pair (i,x) actually forces $\widetilde{f}$ to be fixed at only $2^{n \log n}$ inputs. There are at most $n2^n$ pairs in $\mathcal{S}_f$. Thus if we want $\mathcal{S}_f$ not to shrink, it suffices to fix $\widetilde{f}$ at $2^{3n \log n}$ inputs. By the Interpolation theorem, this means we only need to reserve a small set of "bad" inputs B, of size $\leq 2^{3n \log n}$, beyond those already reserved in previous iterations, i.e., beyond $\mathrm{dom}\, f_{n-1}$, such that on B we have no control as to how f behaves, but on the "good" inputs $\{0,1\}^* \setminus (B \cup \mathrm{dom}\, f_{n-1})$, we can change f arbitrarily. So let f_n be the restriction of f to $B \cup \mathrm{dom}\, f_{n-1}$.

Now that we opened up space in f, we are ready to store the information in (7) so that a small circuit can look it up. That information is the truth table of a function on $n + \log n$ bits, so it suffices to have $2^{2n \log n}$ bits available in $\mathrm{dom}\, f_n$ for this purpose. Since there are at most $2^{3n \log n}$ bad inputs in f_n by the previous paragraph, and since there are at most $2^{4(n-1)\log(n-1)}$ inputs in $\mathrm{dom}\, f_{n-1}$ that are outside $\{0,1\}^{\leq (n-1)^d}$ by induction, we know there are at most $2^{4n \log n}$ inputs currently in $\mathrm{dom}\, f_n$ that are outside $\{0,1\}^{\leq (n-1)^d}$. So there is plenty of space in $\{0,1\}^{n^d}$ for storage when d is large enough. As for how to actually store the information, initially consider each input (i,x) to L_n as prepended with zeroes until it becomes a string $Y_{(i,x)}$ of length n^d, and then set $f_n(Y_{(i,x)}) := L_n(i,x)$. Of course this may not work as some bad inputs may coincide with some $Y_{(i,x)}$, but this can be handled simply by changing the encoding of (i,x) to $Y_{(i,x)} \oplus Z$ for a suitably picked $Z \in \{0,1\}^{n^d}$ — a counting argument shows that such Z exists. This Z can be hardwired to a circuit of size n^d, as we wanted to do.

To finish, let f_n behave arbitrarily on the rest of the good inputs in $\{0,1\}^{\leq n^d}$, and then accordingly adjust f_n on the bad inputs in $\{0,1\}^{\leq n^d}$ — recall from the Interpolation theorem that on a bad input, f_n is a function of how it behaves on non-bad inputs of same length. We have thus constructed f_n as desired. □

4.2 Communication Complexity Approach

AW show that one can take a lower bound from communication complexity, and use it to construct an eligible language — an algebraic oracle in their case — relative to which $\mathcal{C} \not\subset \mathcal{D}$ holds, for an appropriate $\mathcal{C}$ and $\mathcal{D}$ depending on the lower bound picked. Therefore, AW conclude, $\mathcal{C} \subset \mathcal{D}$ cannot have an algebrizing proof.

In this section we develop this approach of AW for our framework. We start by making a notational convention involving the classical communication complexity classes $\mathrm{P}_{\mathrm{cc}}, \mathrm{NP}_{\mathrm{cc}}, \mathrm{BPP}_{\mathrm{cc}}$, etc.

Definition 36 (P_{cc} vs. $\mathrm{P}_{\mathrm{ticc}}$)**.** Define $\mathrm{P}_{\mathrm{ticc}}$ as the class of families $f := \{f_n\}$ satisfying the following. (i) Each f_n is a Boolean function (possibly partial, but not empty) on pairs of 2^n-bit strings, (ii) There is a protocol involving two algorithms M_0, M_1 such that for all n and all $(X,Y) \in \mathrm{dom}(f_n)$, the two parties $M_0^X(1^n), M_1^Y(1^n)$ compute $f_n(X,Y)$ in time $\mathrm{poly}\, n$.

Let P_{cc} denote the relaxation of $\mathrm{P}_{\mathrm{ticc}}$ where M_0, M_1 are allowed to be non-uniform, and only communication between M_0, M_1 is counted towards time elapsed.

Use $\mathrm{P}_{\mathrm{ticc}}$ to define $\mathrm{NP}_{\mathrm{ticc}}, \mathrm{BPP}_{\mathrm{ticc}}$, etc., just as we define $\mathrm{NP}, \mathrm{BPP}$, etc., from P. Similarly for $\mathrm{NP}_{\mathrm{cc}}, \mathrm{BPP}_{\mathrm{cc}}$, etc., versus P_{cc}.

The notation $\mathcal{C}_{\mathrm{ticc}}$ is meant to indicate that time is measured on equal grounds with communication. While $\mathcal{D}_{\mathrm{cc}}$ admits only languages according to the classical definition [8], Definition 36 makes certain partial languages eligible as well (namely those that are defined on every input length).

We formalize the high-level idea of AW with the following generic theorem in our framework:

Theorem 37. *If $\mathcal{C}_{\rm ticc} \not\subset \mathcal{D}_{\rm cc}$, then $\mathcal{C} \subset \mathcal{D}$ cannot be derived via an affinely relativizing proof. Here $\mathcal{C}, \mathcal{D}$ range over every class mentioned in the results of Section 4.2.1; in particular both are definable using* P *and are contained in* EXP.

A key ingredient in arguing Theorem 37 is the observation, mentioned on page 5, that affine extensions are compatible with disjoint unions in the following sense.

Proposition 38. *Let $\mathcal{A}_0, \mathcal{A}_1$ be the affine extension of the languages $\mathcal{O}_0, \mathcal{O}_1$ respectively. Then the disjoint union $\mathcal{A}_0 \coprod \mathcal{A}_1 : bx \mapsto \mathcal{A}_b(x)$ is equivalent, under Cook reductions, to the affine extension of the disjoint union $\mathcal{O}_0 \coprod \mathcal{O}_1 : bx \mapsto \mathcal{O}_b(x)$.*

Proof. Let $\mathcal{O} := \mathcal{O}_0 \coprod \mathcal{O}_1$. By definition, the affine extension of $\mathcal{O}$ is the Boolean version of the function that evaluates, given $B, X_1, .., X_n \in \mathbb{F}_{2^k}$ for any k, the polynomial

$$\begin{aligned}\widehat{O}(BX) &= \sum_{b,x_1,..,x_n \in \{0,1\}} \mathcal{O}(bx) \cdot \textstyle\prod_i (1 + (BX)_i + (bx)_i) \\ &= (\mathcal{O}_0(x) \cdot (1+B) + \mathcal{O}_1(x) \cdot B) \cdot \textstyle\prod_i (1 + X_i + x_i) \\ &= (1+B) \cdot \widehat{\mathcal{O}_0}(X) + B \cdot \widehat{\mathcal{O}_1}(X)\end{aligned}$$

which clearly can be evaluated given access to $\mathcal{A}_0$ and $\mathcal{A}_1$, i.e. to $\mathcal{A}_0 \coprod \mathcal{A}_1$, and vice versa. □

We now give a generic argument for Theorem 37. Supposing there is some $f := \{f_n\}$ in $\mathcal{C}_{\rm ticc} \setminus \mathcal{D}_{\rm cc}$, we construct a language $\mathcal{O}$ such that relative to its affine extension $\mathcal{A}$, the statement $\mathcal{C} \subset \mathcal{D}$ fails. For concreteness, the reader may take $\mathcal{C}$ to be NP, say, and $\mathcal{D}$ to be BPP.

Extend the standard basis with an arbitrary language $\mathcal{O}$, and let M_1, M_2,... be a list of all polynomial-time decision algorithms (with access to $\mathcal{O}$). Since $\mathcal{D}$ is definable from P, we can use the list M_1, M_2,... to define a list $N_1, N_2, ...$ of algorithms that includes every algorithm for $\mathcal{D}$. (The list of N_i's may include more than every algorithm for $\mathcal{D}$: it corresponds to the class $\mathrm{pr}\mathcal{D}$, the extension of $\mathcal{D}$ to partial languages.)

For every $n \in \mathbb{N}$ pick an arbitrary $(X_n, Y_n) \in \mathrm{dom}\, f_n$, and then initialize $\mathcal{O}$ to the disjoint union $\mathcal{O}_0 \coprod \mathcal{O}_1$, where $\mathcal{O}_0$ is the language with the same truth table as X_n for every n, and similarly for $\mathcal{O}_1$ versus Y_n. Because $f \in \mathcal{C}_{\rm ticc}$, the language L that maps every length-n input to the value of $f(X_n, Y_n)$ satisfies $L \in \mathcal{C}^{\mathcal{O}}$. Our objective is to modify $\mathcal{O}_0$,$\mathcal{O}_1$ so that L remains in $\mathcal{C}^{\mathcal{O}}$, and becomes out of $\mathcal{D}^{\mathcal{A}}$. Then we will be done.

For $i = 1..\infty$, the plan is to modify $\mathcal{O}_0, \mathcal{O}_1$ at some large enough input length n_i, by picking some $(X_{n_i}, Y_{n_i}) \in \mathrm{dom}\, f_{n_i}$ and then modifying the truth table of $\mathcal{O}_0, \mathcal{O}_1$ at length n_i to be X_{n_i}, Y_{n_i} respectively. Let us denote this operation by $\mathcal{O}_0 \leftarrow X_{n_i}, \mathcal{O}_1 \leftarrow Y_{n_i}$.

Notice that updating $\mathcal{O}$ in this way readily maintains $L \in \mathcal{C}^{\mathcal{O}}$. As for ensuring $L \notin \mathcal{D}^{\mathcal{A}}$, by Proposition 38, $\mathcal{D}^{\mathcal{A}}$ is the same as the class $\mathcal{D}^{\mathcal{A}_0 \coprod \mathcal{A}_1}$; hence the plan is to choose X_{n_i}, Y_{n_i} so that the i^{th} algorithm N_i disagrees with L on 1^{n_i} when given oracle access to $\mathcal{A}'_0 \coprod \mathcal{A}'_1$, where $\mathcal{A}'_0$ and $\mathcal{A}'_1$ is the affine extension of $\mathcal{O}'_0 := \mathcal{O}_0 \leftarrow X_{n_i}$ and $\mathcal{O}'_1 := \mathcal{O}_1 \leftarrow Y_{n_i}$ respectively.

So, to pick (X_{n_i}, Y_{n_i}), consider the following function g_n for each $n \in \mathbb{N}$. For each $(X, Y) \in \mathrm{dom}\, f_n$, let $g_n(X, Y)$ be the output of $N_i^{\mathcal{A}'_0 \coprod \mathcal{A}'_1}(1^n)$, where $\mathcal{A}'_0$ is the extension of $\mathcal{O}'_0 := \mathcal{O}_0 \leftarrow X$ and similarly $\mathcal{A}'_1$ extends $\mathcal{O}'_1 := \mathcal{O}_1 \leftarrow Y$. (It could be that N_i does not "output anything" on input 1^n because N_i computes a partial language which 1^n is outside the domain of; in this case let g_n be undefined on this X, Y.) Observe that $g := \{g_n\} \in \mathcal{D}_{\rm cc}$, as can be seen by considering the protocol where one party is given access to X and knows $\mathcal{O}_0$, whereas the other party is given Y and knows $\mathcal{O}_1$, and where the two parties simulate $N_i^{\mathcal{A}'_0 \coprod \mathcal{A}'_1}(1^n)$ by using each other as an oracle for $\mathcal{A}'_0$ or $\mathcal{A}'_1$.

Now since $f \notin \mathcal{D}_{\mathrm{cc}}$ by assumption, there are infinitely many (X,Y) on which f and g differ (perhaps by g being not even defined on (X,Y)). Pick any such $(X,Y) \in \mathrm{dom}\, f_{n_i}$ for n_i large enough, say, n_i double-exponential in n_{i-1}, which does not disturb previous phases of the construction since $\mathcal{C}, \mathcal{D}$ are contained in EXP. This completes the construction of $\mathcal{O}$, and finishes the generic argument for Theorem 37.

4.2.1 Applications

The above generic argument for Theorem 37 allows us to replicate the following negative algebrization results of AW.

Corollary 39. *Neither of the following statements can be derived via an affinely relativizing proof: (i)* $\mathrm{coNP} \subset \mathrm{MA}$, *(ii)* $\mathrm{P}^{\mathrm{NP}} \subset \mathrm{PP}$.

Proof sketch. As pointed out by AW, part (i) follows from a result of Klauck on the Disjointness predicate, implying $\mathrm{coNP}_{\mathrm{ticc}} \not\subset \mathrm{MA}_{\mathrm{cc}}$, and part (ii) from a result of Buhrman, Vereshchagin, and de Wolf, implying $(\mathrm{P}^{\mathrm{NP}})_{\mathrm{ticc}} \not\subset \mathrm{PP}_{\mathrm{cc}}$. □

We can use the same argument to replicate a result of IKK as well.

Corollary 40. $\mathrm{RP} \subset \mathrm{SUBEXP}$ *cannot be derived via an affinely relativizing proof.*[8] *Here* SUBEXP *denotes* $\cap_{\varepsilon}\mathrm{DTIME}(2^{n^{\varepsilon}})$.

Proof sketch. Yao's classical result on the Equality predicate implies that the partial function $f(\widetilde{X},\widetilde{Y})$, defined on all $\widetilde{X},\widetilde{Y}$ that is the affine extension to $\mathbb{F}_{2^k}$ of some $X,Y : \{0,1\}^m \to \{0,1\}$, and that is 1 if $\widetilde{X} \neq \widetilde{Y}$ and 0 if not, satisfies $f \notin \mathrm{SUBEXP}_{\mathrm{cc}}$, for a suitable choice of $k \in \Theta(\log m)$. By padding inputs to f so that at each input length $\mathrm{dom}\, f$ is nonempty, and by the error-correcting properties of the affine extension, it follows that $f \in \mathrm{RP}_{\mathrm{ticc}}$. □

4.2.2 Extensions

Refining the AW approach, IKK considerably strengthened a result of AW: they showed that no algebrizing proof (for their notion of algebrization) exists for NP having sub-linear-exponential circuits, even at infinitely many input lengths. We can extend Theorem 37 to replicate this result in our framework as well.

Theorem 41. $\mathrm{NP} \subset \mathrm{i.o.\text{-}SIZE}(2^{\varepsilon n})$ *cannot be derived via an affinely relativizing proof for some* $\varepsilon > 0$.

Proof sketch. Similar to the argument for Theorem 37, we consider a list of all size-$2^{\varepsilon n}$ Boolean circuits on n-bits, for each $n \in \mathbb{N}$. Similarly again, we define a language L that encodes, at each input length n, a single instance of a communication problem $f(X,Y)$, with X and Y being encoded in the oracle. The difference here, following IKK, is that f is the direct product of a Boolean problem instead of a merely Boolean one, for which we know a much stronger variant of $f \notin \mathcal{D}_{\mathrm{cc}}$, namely the strengthening of this to average-case hardness on all input lengths (as opposed to worst-case hardness at infinitely many input lengths). This allows us to use a randomized process to define the oracle "at once", thereby obtaining a hardness for L that holds at every input length. We refer to IKK for details. □

[8] IKK show the stronger result where $\mathrm{SUBEXP} = \cap_{\varepsilon}\mathrm{DTIME}(2^{n^{\varepsilon}})$ is replaced by $\cap_{\varepsilon}\mathrm{DTIME}(2^{\varepsilon n})$. Using a less modular argument we can derive this result as well.

4.3 Proof Theoretic Approach

As mentioned in the beginning of Section 4, sometimes we can get away without constructing oracles, and still show that ψ admits no proof that relativizes affinely. To do so, we find some ψ' which we already know has that status, and then derive the implication $\psi \implies \psi'$ via an affinely relativizing proof. We thus reduce the task of creating an oracle relative to which ψ is false, to doing the same for ψ', with the proof of $\psi \implies \psi'$ serving as the reduction. More generally, we reduce the task of showing that ψ admits no affinely relativizing proof, to doing the same for ψ'.

Using the results of Section 4.1-4.2 we can readily show:

Theorem 42. *None of the following statements can be derived via an affinely relativizing proof: (i)* $\mathrm{NP} \subset \mathrm{P}$*, (ii)* $\mathrm{NP} \not\subset \mathrm{P}$*, (iii)* $\mathrm{NP} \subset \mathrm{BPP}$.

Proof. Part (i): Theorem 35 showed that $\mathrm{NEXP} \not\subset \mathrm{P/poly}$ cannot have an affinely relativizing proof, and Theorem 25 showed that $\mathrm{MA(exp)} \not\subset \mathrm{P/poly}$ via an affinely relativizing proof. The claim follows because $\mathrm{NP} \subset \mathrm{P}$ implies $\mathrm{MA} \subset \mathrm{P}$, which in turn implies $\mathrm{MA(exp)} \subset \mathrm{NEXP}$, both implications being derivable via a relativizing (hence affinely relativizing) proof.

Part (ii): Proposition 32 showed that $\mathrm{PSPACE} \not\subset \mathrm{P}$ cannot have an affinely relativizing proof. The claim follows since $\mathrm{NP} \subset \mathrm{PSPACE}$ via a relativizing proof.

Part (iii): Corollary 39 states that $\mathrm{coNP} \subset \mathrm{MA}$ does not have an affinely relativizing proof. The claim follows since $\mathrm{NP} \subset \mathrm{BPP}$ implies $\mathrm{coNP} \subset \mathrm{MA}$ via a relativizing proof. □

5 Conclusions and Open Problems

Our results counter the folkloric belief that relativizing techniques treat computation only as a "black box" mapping inputs to outputs (e.g., [1, p. 2]), and that arithmetization, or more generally a circuit-based view of computation, seems to let us "peer into the guts of it" [2, p. 115], and hence circumvents the limits of relativizing techniques.

In contrast, according to our definitions, a Boolean formula in the relativizing view, say over the basis $\{\wedge, \oplus, \mathcal{O}\}$, gives complete freedom regarding how the $\mathcal{O}$-gates behave, and in this sense each $\mathcal{O}$-gate is a black box, of "volume" the size of its truth-table. In the affinely relativizing view, however, each $\mathcal{O}$-gate redundantly encodes a Boolean function, by extending its domain from $\mathrm{GF}(2)^n$, say, to $\mathrm{GF}(2^k)^n$; this means that the behavior of the gate is determined by 2^n entries of a truth table of size roughly 2^{kn}. So each $\mathcal{O}$-gate has a black-box "core", carrying on with the metaphor, of volume roughly k^{th} root of its overall volume; here k must be $\Omega(\log n)$ for all the results catalogued in this paper, and can be taken as $\Theta(\log n)$ for a formula of size $O(n)$.

So it seems that: (i) circuit-based techniques *are* relativizing, if they are insensitive to enlarging the basis arbitrarily, (ii) arithmetization-based techniques *are* also relativizing, only "slightly less" so. To make this a bit more precise, consider the following question: what can be the circuit complexity, over the standard basis $\{0, 1, \wedge, \oplus\}$, of a size-n circuit over the extended basis $\{0, 1, \wedge, \oplus, \mathcal{O}\}$? In the relativizing view, i.e., in $\mathcal{RCT}$, the answer is $2^{O(n)}$ — just consider a single $\mathcal{O}$-gate with $n-1$ inputs. To see this in the affine-relativizing view $\mathcal{ACT}$, let us first clean up the definition of affine extension a bit, so that if f is a Boolean function on n inputs, then its affine extension involves $\mathrm{GF}(2^k)$ for $k \geq \log n$ only, instead of $k \geq 1$. By the above discussion, this makes no difference for the results catalogued in this paper, but now the answer is easily seen to be $2^{O(n/\log n)}$, again via an $\mathcal{O}$-gate with $n-1$ inputs. Dividing by n and taking logarithms, we get what might be called the "opacity" of each theory, a quantity that ranges from $O(1)$ at the real-world end

of complexity theory, to $O(n)$ at the fully relativized end, with affine relativization being above $O(n^{1-\varepsilon})$ for every $\varepsilon > 0$, just "slightly less" than relativization.

We finish by listing some suggestions for further research.

A quantitative theory of relativization. Both relativization and affine/algebraic relativization are rigid notions, in the sense that something either relativizes or does not. However, the discussion just above, on the various degrees of being opaque, calls for a theory of relativization that is gradual, based on the information content — or density, so to speak — in an oracle.

Can we associate to each statement a "relativization rank", so that the algebrization barrier arises as a quantitative gap, between a lower bound on one hand for the rank of algebrizing statements, and an upper bound on the other, for the rank of non-algebrizing statements? If so, then we could view the reciprocal of the rank as a useful complexity measure on theorems and conjectures, just as we have complexity measures on algorithmic tasks: the larger the reciprocal of the rank, the higher the "relativization sensitivity" of the statement in hand, indicating more resources — stronger axioms — required to prove it.

New oracles from old. Section 4.3 showed that sometimes we can evade the task of constructing an oracle, by reducing the task to another one already done. For example, there is no need to construct an (affine) oracle refuting $\mathrm{NP} \subset \mathrm{P}$ when we already have one refuting $\mathrm{NEXP} \not\subset \mathrm{P/poly}$, because $\mathrm{NP} \subset \mathrm{P} \implies \mathrm{NEXP} \not\subset \mathrm{P/poly}$ via an affinely relativizing proof — meaning $\mathrm{NP} \subset \mathrm{P}$ is harder to prove than $\mathrm{NEXP} \not\subset \mathrm{P/poly}$ in some sense.

Can we use this idea to simplify the landscape of oracle constructions? For example, many of the statements shown not affinely relativizing in Section 4.2 are containments of the form $\mathcal{C} \subset \mathcal{D}$ for which various circuit lower bound consequences are known. This suggests that a handful of oracles, each refuting some circuit lower bound, may yield a rich collection of statements getting indirectly refuted via reductions of the form given in the above example.

Weaker theories for arithmetization. As asserted in Section 1.2, we can replicate all the classification given by IKK (as well as by AW) for what algebrizes and what does not, however, we do not know if algebrization in the IKK sense implies affine relativization, or vice versa.[9] This suggests that there should be a weaker characterization of arithmetization-based techniques that subsumes both notions.

Is there a constraint that we can place on $\mathcal{O}$, besides that it is a language, so that the resulting theory is a consequence of both versions of $\mathcal{ACT}$, ours and IKKs, and still derives all the theorems shown algebrizing by AW? (Notice that such a theory would automatically be unable to prove anything unprovable by $\mathcal{ACT}$, hence all the non-algebrizing statements of AW.)

Of course, the weakest axiom deriving a theorem is the theorem itself, so there is a trivial answer to the question the way stated above: just take the conjunction of all the algebrizing statements, IP theorem, MIP theorem, etc., and add it as an axiom. This kind of "overfitting" clearly lacks the succinctness desired in a theory, so we need to amend the question a bit. Say that a proof is nontrivial if the proof remains valid when viewed in the theory $\mathcal{RCT} \cup \{\mathcal{O} \text{ is empty}\}$. Then we want a theory that is a consequence of both versions of $\mathcal{ACT}$, and that *nontrivially* derives all theorems shown algebrizing by AW.

The PCP theorem. Section 1.3 explained that both the IP theorem and the MIP theorem can be naturally viewed as a gap-amplification result, and from that point of view both theorems have affinely relativizing proofs. Can we extend this reasoning to the PCP theorem? If so, this would bolster the candidacy of affine relativization as a proxy for arithmetization-based techniques.

[9] The IKK approach is over $\mathbb{Z}$ but can be adapted to $\mathrm{GF}(2^k)$, so this is not the issue. Also, the IKK approach builds on the AIV formulation of $\mathcal{RCT}$, but it can also use our version of $\mathcal{RCT}$, so again this is not the issue.

A completeness theorem for oracles. If we can prove ψ, and that ψ relativizes, then is there a relativizing proof of ψ? It is consistent with experience that such a "completeness" phenomenon holds. Confirming this would allow us to focus solely on proving facts about statements, and not on how we prove those facts.

Along the same lines, what if each statement in a proof relativizes — then does the proof itself relativize? If so, then we could say that a proof relativizes if and only if each of its intermediate statements does. (The "only if" direction is already true by the way we defined things in Section 1.1; the non-trivial part is to make the jump from the semantic fact that each step relativizes, to the syntactic one that the proof relativizes.)

A genuine independence result. Be it in our version of $\mathcal{RCT}$ and $\mathcal{ACT}$, or in AIVs and IKKs, Section 1.2 pointed out that the axioms go on top of an existing collection of axioms governing everyday mathematics. Another approach to formalizing these barriers, would be to propose a *subset* of axioms governing everyday math, the idea being to find the "weakest" version of everyday math that can derive each algebrizing statement, and then to show that no non-algebrizing statement can be derived by that much of mathematics.

Acknowledgements

We thank Scott Aaronson and Eric Allender for their helpful comments. This research was supported by NSF Grant CCF-1420750, and by the Graduate School and the Office of the Vice Chancellor for Research and Graduate Education at the University of Wisconsin-Madison with funding from the Wisconsin Alumni Research Foundation.

Bibliography

[1] Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Transactions on Computation Theory*, 1(1), 2009.

[2] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[3] Sanjeev Arora, Russell Impagliazzo, and Umesh Vazirani. Relativizing versus nonrelativizing techniques: the role of local checkability. Manuscript retrieved from http://cseweb.ucsd.edu/ russell/ias.ps, 1992.

[4] Emil Artin. *Geometric Algebra*. John Wiley & Sons, 1957.

[5] László Babai. E-mail and the unexpected power of interaction. In *Proceedings of the Structure in Complexity Theory Conference*, pages 30–44, 1990.

[6] László Babai and Lance Fortnow. Arithmetization: A new method in structural complexity theory. *Computational Complexity*, 1:41–66, 1991.

[7] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991.

[8] László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory (preliminary version). In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 337–347, 1986.

[9] Theodore P. Baker, John Gill, and Robert Solovay. Relativizatons of the P =? NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

[10] Manuel Blum and Sampath Kannan. Designing programs that check their work. *Journal of the ACM*, 42(1):269–291, 1995.

[11] Harry Buhrman, Lance Fortnow, and Thomas Thierauf. Nonrelativizing separations. In *Proceedings of the IEEE Conference on Computational Complexity*, pages 8–12, 1998.

[12] Alan Cobham. The intrinsic computational difficulty of functions. In *Proceedings of the International Conggress for Logic, Methodology, and Philosophy of Science II*, pages 24–30, 1964.

[13] Stephen A. Cook. Short propositional formulas represent nondeterministic computations. *Information Processing Letters*, 26(5):269–270, 1988.

[14] Lance Fortnow. The role of relativization in complexity theory. *Bulletin of the EATCS*, 52:229–243, 1994.

[15] Lance Fortnow. [Blog post: The great oracle debate of 1993]. Retrieved from http://blog.computationalcomplexity.org/2009/06/great-oracle-debate-of-1993.html, 2016.

[16] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-prover interactive protocols. *Theoretical Computer Science*, 134(2):545–557, 1994.

[17] Lance Fortnow and Michael Sipser. Are there interactive protocols for co-NP languages? *Information Processing Letters*, 28(5):249–251, 1988.

[18] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(3):691–729, 1991.

[19] Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. An axiomatic approach to algebrization. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 695–704, 2009.

[20] Ravi Kannan. Circuit-size lower bounds and nonreducibility to sparse sets. *Information and Control*, 55(1):40–56, 1982.

[21] Richard Lipton. [Blog post: I hate oracle results]. Retrieved from http://rjlipton.wordpress.com/2009/05/21/i-hate-oracle-results, 2016.

[22] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992.

[23] Rahul Santhanam. Circuit lower bounds for Merlin-Arthur classes. *SIAM Journal on Computing*, 39(3):1038–1061, 2009.

[24] Rahul Santhanam. [Comment to blog post: Barriers to proving P!=NP]. Retrieved from http://www.scottaaronson.com/blog/?p=272#comment-7634, 2016.

[25] Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.

[26] Alexander Shen. IP = PSPACE: simplified proof. *Journal of the ACM*, 39(4):878–880, 1992.

[27] Victor Shoup. New algorithms for finding irreducible polynomials over finite fields. *Mathematics of Computation*, 54(189):435–447, 1990.

[28] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time: Preliminary report. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 1–9, 1973.

[29] Seinosuke Toda. On the computational power of PP and +P. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 514–519, 1989.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de