Relativizing versus Nonrelativizing Techniques: The Role of Local Checkability

(early draft)

Sanjeev Arora — U. C. Berkeley
Russell Impagliazzo — U. C. San Diego
Umesh Vazirani — U. C. Berkeley

March 31, 2007

Abstract

Contradictory oracle results have traditionally been interpreted as giving some evidence that resolving a complexity issue is difficult. However, for quite a while, there have been a few known complexity results that do not hold for every oracle, at least in the most obvious way of relativizing the results. In the early 1990's, a sequence of important non-relativizing results concerning "non-controversially relativizable" complexity classes has been proved, mainly using algebraic techniques. Although the techniques used to obtain these results seem similar in flavor, it is not clear what common features of complexity they are exploiting. It is also not clear to what extent oracle results should be trusted as a guide to estimating the difficulty of proving complexity statements, in light of these non-relativizing techniques.

The results in this paper are intended to shed some light on these issues. First, we give a list of simple axioms based on Cobham's machineless characterization of P [Cob64]. We show that a complexity statement (provably) holds relative to all oracles if and only if it is a consequence of these axioms. Thus, these axioms in some sense capture the set of techniques that relativize. Oracle results, while not necessarily showing that resolving a complexity conjecture is "beyond current technology" at least show that the result is independent from these axioms.

We then suggest an axiom which we call the principle of local checkability. When added to the previous axioms, local checkability implies all of the new non-relativizing results. The axiom is intended to capture the fact that, if someone provides a complete transcript of a computation, its correctness is easily checkable through examining its local properties. Different versions of this axiom have been "folklore" in the complexity community, but the extent to which it separates relativized and non-relativized complexity has not been fully appreciated. (In fact, many people remain unconvinced after reading this paper!) We show that all the recent results on interactive proofs remain valid under all oracles that respect this principle, (although they are false with respect to general oracles). We also prove that relative to such oracles, many open questions about complexity classes have essentially the same answer as they do without any oracles. We conclude that relativization techniques cannot prove any meaningful restrictions on the power of the principle of local checkability, which is a known proof technique in complexity theory.

Unfortunately, our axiom is not "uncontroversially relativizable", as is evident from responses to earlier versions of this paper ([For94]). To at least partially respond to our critics, we will give some alternative formulations of "locally checkable" and show that many of our results are quite robust under changes in formulation.

1 Introduction

1.1 History

Relativization was a technique introduced by Baker, Gill and Solovay [BGS75] to provide evidence that standard complexity theory techniques like simulation and diagonalization (see Rabin [Rab59], and Hartmanis and Stearns [HS64]) would not suffice to resolve the P v/s NP question.

Briefly, the relativization technique goes as follows. An *Oracle Turing machine* is defined to be a Turing machine with random access to an *oracle*, which is an infinite array of bits. The array is understood to represent a function from $\{0,1\}^* \to \{0,1\}$, or in other words, a language. For a complexity class $\mathcal{C}$ and an oracle O, the complexity class $\mathcal{C}^O$ is defined as $\{L : L \text{ is accepted by a machine in } \mathcal{C} \text{ with access to } O\}$. A complexity theoretic statement (i.e. a statement like "P = NP" or "NP $\neq$ EXPTIME"), is said to *not relativize* if there are oracles A and B such that the statement holds in the presence of A and not in the presence of oracle B. Thus any proof technique that that is insensitive to the presence of an oracle cannot prove or disprove the statement (for example, the diagonalization/simulation arguments used in [HS64] treat the simulated machine as a black box, and therefore appear to remain valid in the presence of oracles). An example of a statement that doesn't relativize is "P = NP." For this statement, A can be $TQBF$, the language of all true quantified boolean formulae, since $\mathrm{P}^{TQBF} = \mathrm{NP}^{TQBF} = \mathrm{PSPACE}$. A language B such that $\mathrm{P}^B \neq \mathrm{NP}^B$ was exhibited in [BGS75]. Combining this fact with the working assumption that "all known proof techniques in complexity theory are relativizing" gave strong evidence about the difficulty of the P versus NP question.

This method of relativization has been brought to bear upon many other open questions in Complexity Theory, for example: P versus PSPACE [BGS75], NP versus EXPTIME [Dekh69, GH83, Lis86], BPP versus NEXPTIME [H86], IP versus PSPACE [FS87], and a long list of other classes.

In an informal sense, contrary relativizations of a complexity theory statement have been viewed as a *mini-independence result*, akin to the independence results shown in mathematical logic. This notion has never been made precise: in other words, what *independence* is implied by contradictory relativizations, and what are the *proof techniques* from which this independence is implied? (a survey by Hartmanis [Har88] provides further background on this). Of course, it is possible to derive a lot of intuition about a complexity class by studying its behaviour in a relativized setting (see a survey by Allender [All90] for this viewpoint). This informal use of relativization should be distinguished from the formal consequences referred to above.

Although the relativization of time complexity classes is rather straightforward, the relativization of space complexity classes has proved to be a serious problem. The problem is that several provable statements about space complexity classes admit to contrary relativizations - a good example of this is the equivalence between deterministic time and alternating space. This appears to contradict the working hypothesis that "all known proof techniques in complexity theory are relativizing". Attempts have been made to fix this problem by observing that space-bounded relativized computation can be defined in several ways, and one of these choices removes the above-mentioned anomalies (see [Buss88]). In other words, since this was an isolated non-relativizing result, and did not seem to involve "revolutionary new techniques", it seemed more reasonable to doubt the model of relativization with respect to an oracle than to abandon the useful working hypothesis that oracle results meant new techniques would have to be developed to resolve complexity issues.

However, recent results on interactive proof systems and counting classes have made it impossible to save the hypothesis that "all known proof techniques in complexity theory are relativizing". (It should be made clear that we know of no one in the complexity community who actively advocated such a hypothesis, except as an informal rule of thumb.) The following is a partial list of known non-relativizing complexity theoretic statements:

1. $AltSpace(s(n)) = DTIME(exp(O(s(n)))$ ([CKS81]);
2. $NP \subset ZKIP$ if one-way functions exist ([GMW91]);
3. If PP $\neq$ BPP, then there are distributional problems in $Dist$PP $-$ AvgP (i.e., a "difficult on average-case" problem). ([Lip91])
4. IP = PSPACE ([Sha90])
5. MIP = NEXPTIME ([BFL90])
6. NP = PCP($\log n, 1$) [AS92, ALMSS92]).

Many of these results use "algebraic techniques", where Boolean functions are extrapolated to low-degree polynomials. Several people have attributed the non-relativization of these results to use of algebraic methods [For]). However, the first two results do not use algebraic techniques. While the first and last involve classes that are somewhat controversial in exactly how they should be appropriately relativized, all of the rest are non-controversially non-relativing results. The results from [Sha90] and [BFL90] were already known to be false under some oracles [FS87, FRS88]; all of the others except [Lip91] have had relativized counter-examples found after the fact. (Although the proof in [Lip91] certainly uses non-relativizing techniques, we do not know of any relativized counter-example.) For example, the statement NP = PCP($\log n, 1$) also doesn't relativize, as shown in [For]. The *random oracle hypothesis* (the conjecture that any true complexity statement is true with probability 1 in the presence of a random oracle), is also refuted by these results, since all results about interactive proofs mentioned above do not hold in the presence of a random oracle (see, e.g, [CGH90]). Previous counter-examples existed [Kurtz], but seemed somewhat technical.

In view of this, and the wide range of novel techniques used in the proof of these results, there is great interest in identifying the new non-relativizing techniques embedded in these results (and to try to use them to resolve other open questions which had hitherto seemed formidable because of contrary relativizations). It has been felt (see for example the discussion in [LFKN90, BF90]) that the algebraic techniques (like extending boolean formulae to finite fields) used in these results break down in the presence of oracles, and there is some evidence that this is true [For].

At the very least, it seems that the role of relativized computation has to be re-examined, in order that oracles may be used constructively in the future. In this paper, we hope to contribute to this process of re-examining, and to present a framework that clarifies some of the issues.

1.2 Our Results

We give an axiomatic theory, which we call RCT, that makes precise what "relativizing techniques" are. This theory can express any complexity theoretic statement $\mathcal{S}$, in which time and space are specified only within a polynomial (e.g. the statement P = NP). (A more refined version allows space to be specified within a constant factor.) We show that $\mathcal{S}$ relativizes (i.e. holds in the presence of all oracles) iff it is true in every (standard) model of RCT. We in fact show that any $\mathcal{S}$ which can be proved (using the normal axioms of mathematics) to be a relativizing statement, is a theorem of RCT. The contrapositive of this claim is that the kind of "independence" implied when one proves contrary relativizations for $\mathcal{S}$ is precisely independence from RCT.

We then examine the question of how this theory can be strengthened so that recent non-relativizing results may also become provable.

Towards this goal, we identify a general principle, the *principle of local checkability*, which appears to us to be at the root of the above-mentioned non-relativizing results. While this principle can be formulated in several ways, all of them expresses the fact that computation is a very local process, and hence checking the correctness of a computation is simpler than performing the computation itself [1]. We show that "local checkability" is a fact that does not relativize.

Adding the "local-checkability" axiom to our theory enables us to model all relativized worlds in which computations are locally checkable. In such worlds the power of the oracle turns out to be very restricted, and thus the nature of computations is pretty close to that of oracle-less computation. We show that many common complexity theoretic questions resolve in our strengthened theory in essentially the same way (sometimes in a slightly weaker form) as they would in the real world. We present several formulations of this principle, and the extent of characterization of computation varies with the formulation. However, all of the formulations turn out to be essentially equivalent to "SAT (unrelativized) is P^O-complete" (with varying notions of reduction). Thus, local checkability is indeed related to the Cook-Levin Theorem. We conclude that for many interesting unknown non-relativizing complexity facts, if they are resolvable in any standard mathematical system (such as Peano Arithmetic) then they are provable from our axioms.

Admittedly, most of the results presented here are very easy, more observations than theorems. We also admit that the usefulness of our characterization by axioms of "relativizing proof techniques" and " modern proof techniques = relativizing proof techniques + local checkability" is not a mathematical result, and is subjective in nature. It depends on the extent the reader is convinced of the naturalness of the axioms. Indeed, there is much disagreement about exactly what constitutes a "structural complexity-theoretic statement" and whether local checkability in any of its formulations qualifies as such. We hope that by providing several formulations and showing that most of our results are independent of formulation that we convince the reader that local checkability is a basic, natural, and powerful fact about computation. It is a mathematical fact that the local checkability axioms as presented here do imply recent non-relativizing results. However, this in no way denies the novelty, beauty, or power of the algebraic techniques used to establish these results, any more than Peano can claim credit for Gauss's law of reciprocity simply because the latter is provable in Peano Aritmetic!

[1] This fact is also at the root of the Cook-Levin Theorem, and a preliminary version of this paper identified "local checkability" with the term "Cook-Levin theorem"

The rest of the paper is organized as follows: In Section 2 we introduce the axiomatic theory that precisely models "relativizing statements." In Section 3 we formulate the principle of local checkability in several ways, and show how all the ways imply recent results on interactive proofs. In Section 4 we explore the extent to which local checkability characterizes complexity. In particular, the strongest version is powerful enough to resolve almost all important complexity conjectures. We conclude with a discussion of the issues raised by this paper.

2 An Axiomatic Theory

In this section, we give a first order theory which gives an exact characterization of "relativizating tecniques." The theory, which is an extension of Cobham's machine-less characterization of P ([Cob64]), consists of a finite set of axioms concerning the class of " polynomial-time functions" which do not refer to any Turing machines or their inner working.

Thus, we will not be able to even phrase statements which involve exact time or space as opposed to time and space up to a polynomial. In the full version of this paper, we will also give a modification of the axioms which capture relativizing statements about the class $DTIME-SPACE(poly(n), O(n))$, simultaneous polynomial time and linear space. This will allow us to talk about space classes defined up to a constant factor. For simplicity and because relativized space classes are somewhat controversial [For], we will omit this extension in the current abstract.

Instead of having other axioms for other complexity classes such as NP or PSPACE, we prefer to give definitions of these classes directly in terms of $\mathcal{P}$. For example, we would define NP in terms of languages with polynomial-time verifiable certificates, and PSPACE as languages expressible as reachability problems in graphs whose adjacency matrix have entries that are polynomial-time computable and where each node has out-degree 1. (Alternately, a function is in PSPACE if it can be expressed as the first fixed point reached by iterating a decreasing polynomial-time computable function starting at the input.) Statements concerning these other classes are shorthand for the statement concerning P obtained if one translates the terms involved.

2.1 Relativized Complexity Theory

Our presentation in this abstract will be somewhat informal. The following lists some ways in which we will abuse notation and how these abuses would be removed in a more formal presentation. Most readers can skip the next paragraph without much loss in comprehension.

We will be working in a two-sorted extension of number theory where we have variables $f_1, f_2, ...$ representing functions from non-negative integers to non-negative integers. Function variables are meant to mean generic elements of some unspecified set quasi-FP of presumed tractible functions. We use the notation quasi-P to represent the subclass of Boolean (0,1 valued) functions within quasi-FP. [2] First order statements quantifying over the function

[2] quasi-P is sometimes used to denote the time class of functions computable in slightly more than polynomial-time, time exponential in $poly \log n$. Here, the term means an incompletely specified set of functions that has properties similar to those polynomial time has.

variables would intuitively correspond to satements such as "for all tractible functions f there is a tractible function g so that...". We will often write "$\forall f \in$ quasi-FP", but technically the range of the quantifier is implicit. To keep the theory first-order, we would use the standard logical device of using a two-input function symbol *Apply* and re-writing $f(x)$ as $Apply(f, x)$. In the following description, we often represent functions as having multiple arguments, which can be justified (as is usually done) by the use of some specific pairing and projection functions $< x, y >, \Pi_1, \Pi_2$. We can use any such functions computable in polynomial-time, but there are many simple examples computable in linear time and logarithmic space. To say that an explicitly defined function (say addition) is in quasi-P would technically be written $\exists f, f(x, y) = x + y$, but we will use the more intuitive abbreviation $f \in$ quasi-FP, for $f(x, y) = x + y$. We omit such details in the future presentation.

To get our set of axioms, we recall Cobham's machineless characterization of $\mathcal{FP}$, the class of polynomial time computable functions over the integers. The following is paraphrasing Cobham's result, although we have cleaned up some of the recursion schemes slightly.

Theorem 1 *(Cobham [Cob64]) $\mathcal{FP}$ is exactly equal to the smallest class of functions, quasi-FP, satisfying the following:*

1 *quasi-FP contains all the following functions:*

- *Any function f which is non-zero only finitely often.*
- *Any constant function.*
- $x + y$ *and* $x - y$.
- $|x|$, *the length of* x.
- $f(x, y) =$ *yth bit of x ($= 0$ if $y > |x|$).*
- $f(x, y) =$ *the number which differs from x in exactly the yth bit.*
- Π_1 *and* Π_2 *where* $\Pi_1(x, y) = x$ *and* $\Pi_2(x, y) = y$.
- $x \cdot y$.
- $2^{|x|^2}$ *(the so-called "smash function")*

2 *If $f, g \in$ quasi-P, then for $h(x) =< f(x), g(x) >, h \in$ quasi-FP.*

3 *If $f, g \in$ quasi-FP , then $f \circ g \in$ quasi-FP ($\circ$ is functional compostion).*

4 *Let $g(x)$ be any function in quasi-FP so that $\forall x, |g(x)| \leq |x|$. Then $g' \in$ quasi-FP, where $g'(x, k)$, called the repetition of g for $|k|$ steps, is defined recursively by: $g'(x, k) = g(g'(x, k/2))$ if $k > 1$ and $g'(x, 1) = x$.*

Note that all of these axioms are easy to express in our two-typed number theory following the conventions mentioned before. Some of them are a bit forced; we could probably get a more elegant theory by using the characterizations of Bellantoni and Cook ([BC92]) or of Leivant ([Leiv91]) instead of Cobham's.

We add to the above axioms the usual Peano Arithmetic schema of induction axioms for formulas in our language (where the variable being induced on is a natural number, and we are allowed to have both kinds of variables as free variables in the formula.) We call the resulting theory $\mathcal{C}$, the Cobham theory.

We also need two new axioms, one technical, the other asserting the existence of a universal function for quaşi-FP.

Definition 1 *We call the following sentence the* Axiom Length: *For all $f \in cquasi - FP$, there is a $c > 0$ with $|f(x)| < |x|^c$.*

Definition 2 *We call the following sentence* Axiom $\mathcal{U}$: *There exists a function $U(i,t,x) \in$ quaşi-FP such that for every function $f \in$ quaşi-FP, there exists an i and a c such that*

$$f(x) = U(i, 2^{|x|^c}, x).$$

We call the theory obtained by adding $\mathcal{U}$ and Length to the Cobham axioms as Relativizing Complexity Theory or RÇT. We will use CŢ (language of Complexity Theory) to refer to the first order language used to express these axioms. Now we show that RÇT characterizes relativized polynomial time.

Definition 3 *A* standard *model for CŢ is one in which the interpretations of the range of natural number variables and the arithmetical function and relation symbols are the natural numbers with the usual meanings for symbols.*

Note that a standard model of CŢ is specified completely by any set of functions from non-negative integers to non-negative integers. We will abuse notation and refer to a standard model as a set of functions quaşi-FP.

Theorem 2 *Any standard model quaşi-FP satisfies RÇT if and only if there is a set $O \subset N$ with quaşi-FP $= \mathcal{FP}^O$.*

Proof: (Sketch) Clearly, for all oracles O, $\mathcal{FP}^O$ is a model of RCT. Conversely, if quaşi-P is a model of RŞT, then axiom $\mathcal{U}$ gives us a universal function in quaşi-P. Define O as the oracle that on input i, t, x, j outputs the j'th bit of $U(i,t,x)$. It is easy to see from the universality property and $|U(i,t,x)|$ is polynomial, that quaşi-P $\subseteq \mathcal{FP}^O$. We then show that the Cobham Axioms imply that quaşi-P is closed under polynomial-time Turing reducibility. Then just observe that $O \in$ quaşi-P from Axioms 1 and 3. □

We stress the simplicity and "obviousness" of our axioms. The moral here is that if we restrict ourselves to formal models that have the "natural" properties we expect in polynomial time computation, then polynomial time oracle Turing machines are all we can get. In this sense, there's no getting away from oracles if one wants to show that a result is beyond the reach of a set of techniques that contains simple compositions and the use of universal machines.

It is clear that any theorem in RCT is a relativizing statement about $\mathcal{FP}$. Some reflection also shows the following: any relativizing statement we can prove (using Normal Math + usual notion of oracle Turing machines) about P is just a theorem of the system Normal Math + RÇT. More precisely, let Normal Math be any theory extending Peano Arithmetic. Then the corresponding theory for relativized Normal Math would be Normal Math with an undefined function symbol O meaning the oracle, and an extended induction schema allowing formulas to mention O. Similarly, let NormalMath+RCT be an extension of Normal Math to CŢ where we allow induction to involve formulas using function variables. Any statement in C̦T has a corresponding statement in relativized normal math where we replace quasi̧-FP by F̧P^O. Then it is easy to show that a statment in LCT is a theorem of NormalMath + RCT if and only if its translation is a theorem of relativized $NormalMath$. We will present this in more detail in the final version.

Thus all "relativizing" facts one can hope to prove about P are provable within RCT. We conclude that "relativizing techniques" are just derivations involving RÇT.

2.2 Defining Other Classes in Terms of P

Note that the language above, LCT, cannot directly mention other complexity classes besides P (or more exactly, quaşi-P.) However, most other complexity classes have straight-forward definitions in terms of P. For example, we would use quaşi-P = quasi̧-NP as an abbreviation for: $\forall c, \forall f \in$ quaşi-P, $\exists g \in$ quaşi-P, so that $g(x) = 1 \iff \exists y|y| \leq |x|^c \wedge f(x, y) = 1$. Only classes that are robust as to changes in time and space up to a polynomial are formalizable in our language. When wanting to formulate complexity statements about these classes in our language, we use these definitions to translate the statement into one about only quaşi-P. (Note that if one drops the existence of the smash function and strengthens the length axiom to linear from polynomial, then the axiomization given captures TIME-SPACE(p(n), n), simultaneous polynomial time and linear space. We could use this theory instead of RCT if we want to capture time up to a polynomial and space up to a constant.)

In this version, we will not go on at length as to how to define other classes in terms of quaşi-P. However, we will give a few simple rules. First, to get time classes that involve greater than polynomial time, use padded versions of polynomial-time functions. For example , quasi̧EXP = $f(x, 2^{|x|^k})|k \in N, f \in$ quaşi-P.

To define sub-linear classes, we use the non-uniform versions of these classes, but insist that the families of non-uniform computational devices be locally quaşi-P-uniform. For example, quasi-NĻogspace would be defined as reachability problems on directed graphs whose node names are $log(n)$ bit strings, and whose edges each depend on one bit of the input.

By adopting this point of view, we want to avoid questions about oracle access that are specific to classes. By looking at questions about other classes in terms of closure properties of the polynomial-time computable functions, we hide details of how the oracle is accessed in various models of computation. In other words, we want C^O to represent those problems that would be computable in C if O were computable in P, not the set of problems C-reducible to O. Comparing the set of problems C_1-reducible to O versus those C_2-reducible to O is comparing consequences of O being in C_1 to consequences of O being in C_2, rather than the consequences of a single hypothesis.

In any case, if any class is uncontroversially relativized, it is P. So results about P should also be uncontroversial in how to relativize them (or such was our hope.)

If O were in C, what else would be in C? One of the reasons we prefer the first type of question is that the premise is the same for each complexity class. Thus, when we compare C_1^O to C_2^O, we are asking: if O were in P, would $C_1 = C_2$?, rather than: is the set of functions C_1 reducible to O the same as the set of functions C_2 reducible to O? The other is that it makes clear that questions about many other classes are really questions about closure properties of P, so the issue of how to relativize a class becomes less relevant.

3 The Principle of Local Checkability

In this section, we introduce some notions of *local checkability* that do not relativize. (Fortnow has pointed out that there are other models and formulations of similar questions that do relativize ([For]). However, we feel that that is irrelevant, since the principles as stated both do not relativize and do imply many of the powerful non-relativing results in a relativizing way.)

As suggested in the introduction, this principle can be formulated in different ways, none of which will affect the core of our arguments. We settle upon two formulations that may be viewed as stronger forms of the usual Cook-Levin Theorem.

We define a *proof-checker* to be a Turing machine M that uses $\forall$ quantification and which is provided, in addition to the input, a *proof string.* It is allowed random access to both the input and the proof string. It is said to *accept* an input x using proof-string Π (denoted $M(x, \Pi) = 1$) iff all branches created by its $\forall$ branching accept. It is said to run in time $t(n)$ if its running time is $O(t(n))$ on inputs of size n.

Definition 4 *(PF-CHK($t(n)$)) A language L is in the class PF-CHK($t(n)$) iff there is a proof-checker M that runs in $t(n)$ time and has the property*

- *$\forall x \in L$, there exists a Π such that $M(x, \Pi) = 1$.*
- *$\forall x \notin L, \forall \Pi$, $M(x, \Pi) = 0$.*

In order to make the concept of proof checker more robust, we can define a version just in terms of polynomial-time computable functions. While giving a less exact characterization of "known non-relativizing techniques", weak local checkability is possibly more convincing to skeptics, since it allows full access to the oracle.

To make it a bit simpler, we'll just look at non-adaptive proof checkers. For $\Pi \in \{0, 1\}^m$, and $S \subseteq \{1, ..m\}$, let $\Pi_{|S}$ denote the substring obtained by restricting Π to the co-ordinates in S.

Definition 5 *(WPF-CHK($t(n)$)) A language L is in the class WPF-CHK($t(n)$) iff there are functions $f, g \in FP$ and constants $c, d > 0$ so that for each $y, |y| \leq ct(n)$, $g(x, y)$ returns a set $S \subset \{1, ..n^d\}$ with $|S| \leq ct(n)$ and so that:*
$\forall x, x \in L$ if and only if there exists a $\Pi, |\Pi| = n^d$, such that $\forall y, |y| \leq ct(n)$, $f(x, y, Pi_{|g(x,y)}) = 1$.

Note that f and g are even allowed to run in time polynomial in n, and to use x arbitrarily. The only "locality" constraint is that, for each y, only $t(n)$ bits of the proof Π can be read.

Theorem 3 *PF-CHK($\log n$) $\subseteq$ WPF-CHK($\log n$).*

Proof: The only advantage a proof system in PF-CHK($\log n$) has is that it can read bits of the proof adaptively. To simulate this, we can append to Π, for each sequence y of non-deterministic choices and each sequence z of supposed bits of the proof read so far, a bit $P(y, z)$ which represents "the bit of the proof Π that would be read next". Then we make the new list of non-adaptive constraints: On y, z, i, read all the $P(y, z')$ for z' a subsequence of z, and read Π_i. Accept if one of the bits of z does not match the corresponding $P(y, z')$. Otherwise, reject if either the machine on path y with answers z would reject, or if the machine on path y with answers z reads i and gets an answer other than Π_i. $\square$

The term "Local Checkability Theorem" or "LCT" (also, the Levin-Cook Theorem!) from now on will refer to the following statement.

Proposition 4 *$NP = PF\text{-}CHK(\log n)$.*

Proof: Clearly, PF-CHK($\log n$) $\subseteq$ NP. The other direction follows from the observation from the Cook-Levin Theorem ([Coo71, Lev73]), that witnesses for NP languages can, w.l.o.g. be in the form of a tableau. The correctness of the tableau can be checked by examining all $O(1)$ sized "windows". $\square$

It follows that:

Proposition 5 *$NP = WPF\text{-}CHK(\log n)$.*

Theorem 6 *If O is a random oracle then*

$$P^O \not\subseteq PF\text{-}CHK(\log n)^O$$

with probability 1*, and*

$$P^O \not\subseteq WPF\text{-}CHK(\log n)^O$$

with probability 1*.*

Proof: Consider the language O. Clearly, $O \in \mathrm{P}^O$. We claim that $O \notin$ PF-CHK$(\log n)^O$ with probability 1. The reason is that a proof-checker running in $O(\log n)$ time can only query the oracle about strings of size $O(\log n)$. Therefore for large enough strings, its answer to "Is $x \in O$?" is a random variable that is uncorrelated with whether $x \in O$ (i.e. is correct only with probability $\frac{1}{2}$). A standard probability analysis now yields the result.

$\square$

We conclude that most oracles are unrealistic in this sense, and define a class of oracles that model reality better.

Definition 6 *An oracle O is* consistent with LCT *if $NP^O = PF\text{-}CHK(\log n)^O$. It is* consistent with weak LCT *if $NP^O = WPF\text{-}CHK(\log n)^O$.*

The following theorems show that adding such oracles can not change the answers to many complexity questions.

Theorem 7 *If O is an oracle that is consistent with LCT, then*

(i)$NP^O = NP$ (ii) $O \in NP \cap co\text{-}NP$.

Proof: (i) Let $L \in \mathrm{NP}^O$. Then $L \in \mathrm{NP}/poly$, for the following reason: the proof-checker can only query the oracle for strings of size $O(\log n)$. Hence it only uses $n^{O(1)}$ bits of information from the oracle over all its branches, and these bits can be provided instead as an advice string.

Now we show that actually $L \in \mathrm{NP}$. In other words, if the oracle doesn't impart any more power to polynomial time computation than to a $\log n$ time proof-checker, then it provides no information that an NP machine couldn't guess and verify on its own. Notice that once the proof-checker is done with its $\forall$ guessing, its decision is a deterministic poly time computation on some $O(\log n)$ bits. So the proof-string can just be required to contain a sub-proof (for all possible choices of $\forall$ guesses) that this computation accepts. The subproof can be checked by a $\log\log n$ time proof-checker. Now the only oracle queries it can make are of size at most $O(\log\log n)$. Repeating this recursion idea some more, we conclude that only the first $O(1)$ bits of the oracle are needed, which can be incorporated into the checker's program.

(ii) It is well known that $\mathrm{NP}^O = \mathrm{NP}$ iff $O \in \mathrm{NP} \cap \text{co-NP}$. □

Since oracles consistent with LCT are so restricted, it is clear that that resolving the P versus NP question with respect to any such an oracle would essentially resolve the question in the real world.

Theorem 8 *If O is any oracle that is consistent with LCT, then the following hold:*

- *If $P^O \neq NP^O$ then $P \neq NP$.*
- *If $P^O = NP^O$ then $NP =$ co-NP.*
- *$EXPTIME^O = NP^O$ iff $EXPTIME = NP$.*
- *If $PSPACE^O \neq P^O$ then $PSPACE \neq P$. If $PSPACE^O = P^O$ then $PSPACE = NP \cap$ co-NP.*

Proof: Follows from Theorem 7. □

Even weak local checkability is very restrictive:

Theorem 9 *If O is an oracle that is consistent with weak LCT, then SAT (unrelativized) is NP^O-complete under P^O reductions.*

Proof: We show SAT is P^O hard for WPF-CHK$(\log n)^O$ for every oracle O, whether or not it satisfies weak local checkability. The claim then follows immediately.

Fix polynomial-time computable functions with oracle, $f^O(x, y, z)$ and $g^O(x, y)$, where $|y|, |z| \leq c \log n$. We'll show how to reduce the corresponding WPF-CHK$(\log n)^O$ language to SAT. Given input x of length n, let $p_1, ..p_{n^d}$ be variables representing the bits of an unknown witness Π. For each y of length n, compute $S = g^O(x, y)$. There are only polynomially many settings z of Π_S. For each one, if $f^O(x, y, z)$ does not accept, write down a clause saying that the bits of Π in S cannot all match z. Thus, an assignment to the variables $p_1, ..p_{n^d}$ satisfies the resulting formula if and only if, for the corresponding witness Π, for all y, $f^O(x, y, \Pi_{|g^O(x,y)})$ accepts.

□

A few corollaries follow:

Proposition 10 1. *With probability* 1*, a random oracle O does not satisfy weak local checkability*

2. *If there is an oracle O that satisfies weak local checkability, and $P^O \neq NP^O$, then $P \neq NP$*

3. *Any oracle O for which $P^O = NP^O$ satisfies weak LCT.*

Proof: The first part follows by relativizing the argument of Bennett and Gill that $NP^O \neq P^O$ with probability 1 to show that $NP^O \not\subseteq P^{O,SAT}$ with probability 1 ([BG81]). From the above theorem, WPF-CHK$(\log n)^O \subseteq P^{O,SAT}$ for every O.

If $P = NP$, then $SAT \in P \subseteq P^O$, so WPF-CHK$(\log n)^O \subseteq P^O$ for every oracle O.

Finally, for each oracle O, $P^O \subseteq$ WPF-CHK$(\log n)^O$, since f can ignore the proof and perform any polynomial-time computation on x. Therefore, if $P^O = NP^O$, $P^O \subseteq$ WPF-CHK$(\log n)^O \subseteq NP^O = P^O$, so all classes in the chain are equal. □

Therefore, unlike for LCT, we have oracles of arbitrary power satisfying weak LCT. However, seperating P from NP with any such oracle also separates unrelatived P from unrelativized NP.

3.1 Frequent Objections

An objection that is frequently raised to studying oracles that're consistent with LCT is the following: the PF-CHK$(\log n)$ machine and the NP machine have very different types of access to the oracle, and therefore studying oracles in this situation is uninteresting. We gave a partial answer by introducing the notion of weak local checkability, which is more robust in access. Also, weak local checkability is defined in terms of what is computable in polynomial-time. So weak LCT is a statement about the closure properties of P. Fortnow ([For]) points out there is a relativizing local checkability property if the oracle access model is changed. However, this does not refute the idea that our particular concept of local checkability is responsible for many of the non-relativizing complexity results. Local checkability is dependent on logarithmic time computation, which is not robust under changes of machine model. However, many of the results hold assuming weak local checkability, which is robust under model changes. We could introduce some intermediate concepts of local checkability, where more of the consequences would hold.

4 The Local Checkability Axiom

Now if we add the principle of local checkability (from Section 3) to RCT as an axiom, we can characterize all relativized worlds in which the principle holds. We call this new theory $RCT + LCT$.

The interesting point which follows from Theorem 8 is that if any interesting facts (like P $\neq$ NP) are provable in normal mathematics, then essentially the same facts are provable with $RCT + LCT$.

It is also the case that most of the non-relativizing results in complexity follow from these extended axioms.

Theorem 11 *If O is any oracle that satisfies local checkability,*

- $IP^O = PSPACE^O$,
- $MIP^O = NEXPTIME^O$, *and*
- $PCP(\log n, 1)^O = NP^O$

The last two also hold if O is consistent with weak LCT.

Proof: Notice that that any O consistent with LCT is just a language in NP $\cap$ co-NP, and therefore $\text{PSPACE}^O, \text{NP}^O, \text{NEXPTIME}^O$ etc. are the same as the unrelativized classes. The claim follows. For weak LCT, we can reduce any NP^O problem to SAT and then use any PCP for SAT to prove the PCP theorem. We can use a padding argument to convert this to get the multi-prover proof for NEXPTIME. □

To clarify why the above theorem is true we carefully trace through the role of local-checkability in the results on interactive proofs. If O is consistent with LCT, then NP^O is polynomial time reducible to $3SAT$ (as implied by Theorem 7). Similarly, a padding argument shows that NEXPTIME^O is reducible to an exponential size $3SAT$ formula in such a way that each clause can be computed in polynomial time. The techniques of [BFL90, ALMSS92] now show that $\text{MIP}^O = \text{NEXPTIME}^O$ and $\text{NP}^O = \text{PCP}(\log n, 1)^O$. Similarly, the polynomial time reduction from PSPACE to $TQBF$ in [SM73] also merely uses the "local checkability" of a computation in P (in other words, that PSPACE = Alt-P). The reduction to $TQBF$ carries over in the presence of O, since NP (and hence P) is reducible to $3SAT$. Then the techniques of [Sha90] show that $\text{IP}^O = \text{PSPACE}^O$.

5 Discussion

In this section we expand upon the ideas presented in the previous sections.

Firstly, it is a very surprising fact that all intuition about oracles can be stated by such simple axioms. Oracles were developed as intuitive ways to separate "easily provable" statements from "hard-to-prove" ones. Our axioms exhibit the that the rigorous underpinning of that body of work is surprisingly simple. This exact logical characterization in a sense demonstrates both the strengths and the limitations of oracle methods.

The positive side is that if we want to stick only with models of computation that follows "common-sensical" properties like closure under composition (and other properties given in RCT) then oracle turing machines are the only such model. In this sense there is no getting away from oracle TM's.

On the negative side is the extreme simplicity of our axiomatic system. Since "techniques that relativize" correspond exactly to logical derivations within this simple system, one concludes that proving contrary relativizations for a complexity-theoretic statement is, at best, a very limited independence result.

More seriously, the axioms of RCT do not capture the notion of what is currently provable in complexity theory; in particular, the recent non-relativizing results about interactive proofs.

Hence if we wish make oracles a method of separating "provable" statements from "ones-we-don't-know-how-to-prove", we are left with no option but to extend *RCT* by adding new axioms.

One option would be to just add all "significant" non-relativizing facts as new axioms whenever they are discovered, since they will all be contributions to the repertoire of "known techniques." Such ongoing "patching" clearly lacks the succinctness one seeks in the axiomatic approach.

Thus the choice of a new axiom has to be guided at the minimum by the following considerations (which guide mathematics in general): Firstly, the new axiom has to be independent of existing axioms. Secondly, it it must correspond to our intuitions and let us prove interesting theorems. Thirdly, the axiom has to express a real fact about computation as we understand it.

To us, the principle of local checkability seems to satisfy all these criteria. Certainly its incorporation makes *RCT* powerful enough to prove almost all interesting non-relativizing statements (or some weaker forms of these statements). Unfortunately, adding LCT as an axiom also rules out most oracles. (This seems to be inevitable if statements like MIP = NEXPTIME have to be provable in our system.)

But our hope is that this new axiom will focus attention on the true non-relativizing part of recent results in complexity theory, and hopefully, motivate more such non-relativizing results.

References

[All90] E. Allender, Oracles versus Proof Techniques that Do Not Relativize. Invited paper at *SIGAL's Int'l Symposium on Algorithms, 1990.*

[ALMSS92] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof verification and intractability of approximation problems. March 1992.

[AS92] S. Arora and S. Safra. Probabilistic Checking of Proofs; A New Characterization of NP. February 1992.

[BF90] L. Babai, L. Fortnow. Characterization of $\sharp$P by Arithmetic Straight Line Programs. In *Proceedings of the 31st FOCS*, 1990, pages 26–34.

[BFL90] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. In *Proc. 31th IEEE Symp. on Foundations of Computer Science*, pages 16–25, 1990.

[BGS75] T. J. Baker, J. Gill, and R. Solovay. Relativizations of the P = NP question. *SIAM Journal of Computing* 1975 (4), pages 431-442.

[BC92] S. Bellantoni, and S. A. Cook, A New Recursion-Theoretic Characterization of the Polytime Functions. *Computational Complexity*, Vol. 2: 97-110, 1992.

[BG81] C. H. Bennet, J. Gill. Relative to a Random Oracle P(A) $\neq$ NP(A) $\neq$ co-NP(A) with probability 1. *SIAM Journal of Computing*, 1981 (10), pages 96 -113.

[Buss88] J. Buss. Relativized Alternation and Space-Bounded Computation, *J. Comput. Syst. Sci.* 36(3): 351-378 , 1988.

[CKS81] A. K. Chandra, D. Kozen, and L. J. Stockmeyer, Alternation. *J. ACM*, 28(1), pp. 114-133 , 1981.

[CGH90] B. Chor, O. Goldreich, J. Hastad. The Random Oracle Hypothesis is False. *Manuscript*, 1990, quoted in [LFKN90].

[Cob64] A. Cobham. The Intrinsic Computational Difficulty of Functions. In *Proc. 1964 International Congress for Logic, Methodology and the Philosophy of Science*, pages 24-30.

[Coo71] S. A. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd ACM Symp. on Theory of Computing*, pages 151–158, 1971.

[Dekh69] M.Dekhtiar. On the impossibility of eliminating exhaustive search in computing a function relative to its graph. *DAN SSSR = Soviet Math. Dokl.* 14:1146-1148, 1969.

[For94] L. Fortnow. The Role of Relativization in Complexity Theory. *Bulletin of the EATCS*, 52: 229-243, 1994.

[For] Lance Fortnow. Personal Communication.

[FRS88] L. Fortnow, J. Rompel, and M. Sipser. On the power of multi-prover interactive protocols. In *Proc. 3rd STRUCTURES*, pages 156–161, 1988.

[FS87] L. Fortnow and M. Sipser. Are there interactive protocols for co-NP Languages? In *Information Processing Letters* 28, North-Holland, 1988, pages 249-251.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[GH83] I. Gasarch and S. Homer Relativizations comparing NP and EXPTIME. *Information and Control*, 1983 (58) pages 88-100.

[GMW91] O. Goldreich, S. Micali, and A. Wigderson, Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *J. ACM*, 38(3), pp. 691-729, 1991.

[Har88] J. Hartmanis. New Developments in Structural Complexity Theory. In *Proc. Structures, '88*, pages 271-286.

[HS64] J. Hartmanis and R.E. Stearns. Computational complexity of recursive sequences *FOCS*, pp. 82-90, 1964

[H86] Heller. On Relativized Exponential and Probabilistic Complexity Classes. *Information and Computation* (1986), pages 231–243.

[IL] G. Itkis and L. Levin. *Personal Communication.*

[Kar72] Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[Kurtz] S. Kurtz, On the random oracle hyptothesis, *Information and Control*, 57(1), pp. 40-47, 1983.

[Leiv91] D. Leivant, A foundational delineation of computational feasibility, LICS 91, pp. 2-11, 1991.

[Lev73] L. Levin. Universal'nyĭe perebornyĭe zadachi (universal search problems : in russian). *Problemy Peredachi Informatsii*, 9(3):265–266, 1973.

[Lip91] R. Lipton, New directions in testing, In J. Fegenbaum and M. Merritt, editors, *Distributed Computing and Cryptography*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 2, pages 191-202. American Mathematical Society, 1991.

[Lis86] G. Lischke. Relationships Between Relativizations of $\mathrm{P}, \mathrm{NP}, \mathrm{EL}$, NEL, EP and NEP. *Zeitschrift fur Mathematische Logik und Grundlagen der Mathematik*, 1986 (2) pages 257 –270.

[LFKN90] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. In *STOC90*, pages 2–10, 1990.

[Orp83] P. Orponen. Complexity Classes of Alternating Machines with Oracles. In *Proc. 10th ICALP, LNCS 154* pages 573-584.

[Rab59] M.Rabin. Speed of computation and classification of recursive sets. *Third Conv. Scient. Societies* 1-2, Israel, 1959; (also Tech. Rep. 2, Hebrew Univ., Jerusalem, 1960).

[Sha90] A. Shamir. IP=PSPACE. In *Proc. 22nd ACM Symp. on Theory of Computing*, pages 11–15, 1990.

[SM73] L. J. Stockmeyer, A. R. Meyer. Word Problems requiring exponential time. *Proceedings of 5th STOC*, 1973, pages 1–9.