A tight characterization of NP with 3 query PCPs

Venkatesan Guruswami* Daniel Lewin* Madhu Sudan* Luca Trevisan*†

July 24, 1998

Abstract

It is known that there exists a PCP characterization of NP where the verifier makes 3 queries and has a *one-sided* error that is bounded away from 1; and also that 2 queries do not suffice for such a characterization. Thus PCPs with 3 queries possess non-trivial verification power and motivate the task of determining the lowest error that can be achieved with a 3-query PCP. Recently, Håstad [10] has shown a tight characterization of NP by constructing a 3-query PCP verifier with "error" arbitrarily close to 1/2. Unfortunately, this verifier makes *two-sided* error and Håstad makes essential use of this feature. One-sided error, on the other hand, is a natural notion to associate with a proof system, since it has the desirable property that every rejected proof has a short counterexample. The question of determining the smallest error for which there exists a 3-query PCP verifier making one-sided error and accepting an NP-complete language, however, remained open.

We resolve this question by showing that NP has a 3-query PCP with a one-sided error that is arbitrarily close to 1/2. This characterization is *tight*, i.e., the error cannot be lower. This result is in seeming contradiction with the results of Trevisan [14] and Zwick [16] who show that in order to recognize an NP-complete language, the error probability of a PCP verifier making 3 *non-adaptive* queries and having one-sided error must be at least 5/8. We get around this bottleneck by designing an *adaptive* 3-query PCP for NP. Our result yields the first tight analysis of an adaptive PCP; and reveals a previously unsuspected separation between the powers of adaptive and non-adaptive PCPs.

Our design and analysis of adaptive PCPs can be extended to higher number of queries as well and we give an example of such a proof system with 5 queries. Our adaptive verifiers yield proof systems whose error probabilities match those of previous constructions, while also achieving one-sidedness in the error. This raises new questions about the power of adaptive PCPs, which deserve further study.

*Laboratory for Computer Science, MIT, 545 Technology Square, Cambridge, MA 02139, USA. {venkat,danl,madhu,luca}@lcs.mit.edu.

†Research supported by the Italian CNR.

1 Introduction

The construction of efficient probabilistically checkable proofs (PCPs) has been a subject of active research in past few years. A sequence of surprising developments [4, 3, 8, 2, 1, 6, 9, 7, 5] recently culminated in the striking results of Håstad [10] showing that every language in NP has a PCP verifier querying 3 bits and having error probability[1] arbitrarily close to $\frac{1}{2}$. This characterization of NP is tight in the sense that no verifier querying 3 bits could achieve an error strictly smaller than $\frac{1}{2}$. Håstad actually describes a general machinery for determining the error of PCP verifiers using Fourier analysis — a machinery which could in principle yield a tight analysis of *any* given verifier.

In the process of proving this "tight" characterization of NP, Håstad's work exposes some of the previously unexplored subtleties in the definition of a PCP proof system. Recall that such a proof system is described by an (r, q)-restricted PCP verifier, i.e., a probabilistic polynomial time oracle machine, who on input x, tosses $r(|x|)$ random coins and makes $q(|x|)$ queries to a proof oracle Π. A language $L \in \mathrm{PCP}_{c,s}[r, q]$ if there exists an (r, q)-restricted verifier V satisfying: (1) *(completeness)* If $x \in L$, then $\exists\ \Pi$ s.t. $\Pr_R[V^{\Pi}(x : R)$ accepts$] \geq c$. (2) *(soundness)* If $x \notin L$, then $\forall\ \Pi\ \Pr_R[V^{\Pi}(x : R)$ accepts$] \leq s$ (where $V^{\Pi}(x; R)$ denotes the computation of V on input x and random string R with oracle Π). We refer to the ratio s/c as the *error* of the verifier. Notice that while the definition allows for the verifier to make *two-sided* error, most PCP constructions to date restricted their attention to verifiers making one-sided error (or have *perfect completeness* i.e., have $c = 1$). There are several reasons for this: (1) It was not known how to exploit the power of two-sided error and (2) Verifiers with one-sided error work better in "composition" and in many applications to inapproximability results. Moreover, there is an aesthetically pleasing element to PCP proof systems with one-sided error: The proof system explicitly exhibits a flaw in any proof it rejects; and in the case of 3 query verifiers, the flaw is in the 3 bits queried.

Håstad's construction, however, yields a verifier making two-sided error. Specifically, when $x \in L$ the verifier makes an arbitrarily small but non-zero error. This gap is inherent in his construction and leaves open the question: What is the lowest error that can be achieved in a 3-query PCP for NP, making one-sided error? This question is especially relevant to one-sided error, because (only) in this case is it known that 3 is the smallest number of queries for which the error can be bounded away from 1. In light of the newly acquired ability to perform (at least in principle) a tight analysis of almost any PCP verifier, it seems feasible to examine this question: The only challenge seems to be in designing the right PCP verifier. Yet, the best previous construction of a PCP verifier that queries three bits and has perfect completeness only achieve an error probability arbitrarily close to 3/4 [10].

Trevisan [14] and Zwick [16] show a fundamental barrier to this quest: They show that any verifier making 3 *non-adaptive* queries to the proof oracle, and achieving a one-sided error better than 5/8 can only recognize languages in P. This brings up another subtlety in the definition of PCP. The definition actually allows the queries of the verifier to be generated *adaptively*: i.e., the questions asked may depend upon answers to previous questions. Most previous constructions do not use adaptivity. However, to get a tight answer to the 3-query question, it seems necessary to build an adaptive verifier; and the only construction of an adaptive PCP verifier in the literature is a construction of Bellare et al. [5]. Thus this entire area seems relatively unexplored.

We build a new *adaptive* 3-query verifier for NP. This verifier is based on a combination of

[1] We will use this term loosely for the present and formalize it shortly.

the adaptive verifier of Bellare et al. [5] and the non-adaptive verifier with perfect completeness of Håstad [10]. We perform a tight analysis of this verifier and obtain a somewhat surprising result:

Theorem 1.1 *For every $\varepsilon > 0$,* $\mathrm{NP} = \mathrm{PCP}_{1,\frac{1}{2}+\varepsilon}[\log, 3]$.

The theorem is tight since, as pointed out earlier, any 3-query verifier for NP must make an error with probability at least $\frac{1}{2}$. This theorem, therefore, resolves a central question relating to PCPs by obtaining a *tight characterization of NP* in terms of a 3-query PCP making one-sided error. The surprising element of the result above is that it shows that an adaptive verifier can achieve a lower error than any non-adaptive verifier — thereby establishing a separation between adaptive and non-adaptive PCPs. Prior to this result there was no evidence indicating that such a separation might exist. In fact, on the contrary, Trevisan [13] points out that adaptive and non-adaptive PCPs actually have the same power for PCPs with *two-sided* error. Of technical interest is that we extend (in retrospect, quite easily) the Fourier analysis method of Håstad to the case of adaptive PCPs.

We move on to examine PCPs with slightly higher number of queries. This examination is motivated primarily by the following question: Is it true that every additional query increases the power of PCPs ? (I.e., is $\mathrm{PCP}_{1,s}[\log, q] \subseteq \mathrm{PCP}_{1,s'}[\log, q+1]$, for some $s' < s$?[2]) It is easy to see that 3 additional bits certainly reduce the error. Yet, for one additional bit we do not know the answer. In fact, prior to this paper, it was not even known if there exists a non-trivial q ($q \geq 3$) for which this statement is true. To answer such questions, we prove some more new (but not necessarily tight) characterizations of NP in terms of PCP. These are described below, where the notation naPCP below stands for non-adaptive PCP.

Theorem 1.2 *For every $\varepsilon > 0$, the following hold:*

(1) (4 **non-adaptive queries**) $\mathrm{NP} = \mathrm{naPCP}_{1,\frac{1}{2}+\varepsilon}[\log, 4]$.

(2) (5 **adaptive queries**) $\mathrm{NP} = \mathrm{PCP}_{1,\frac{1}{4}+\varepsilon}[\log, 5]$.

(3) (5 **non-adaptive queries**) $\mathrm{NP} = \mathrm{naPCP}_{1,\frac{7}{16}+\varepsilon}[\log, 5]$.

(4) (6 **non-adaptive queries**) $\mathrm{NP} = \mathrm{naPCP}_{1,\frac{1}{4}+\varepsilon}[\log, 6]$.

Part (2) of result is where the main technical work is done. Parts (1) and (4) are immediate corollaries of the adaptive protocols in Theorem 1.1 and Part (2) of the theorem above: The non-adaptive verifier is obtained by reading all possible bits that may be read by the adaptive verifier. It is interesting to observe that that for several choices of q, the best known non-adaptive PCP verifier is obtained by starting from an adaptive one. Part (3) requires some modification of the verifier of Part (2).

Finally, using the semidefinite programming methodology of Karloff and Zwick [11, 16], we also prove the following containment result for 4-query PCP.

Theorem 1.3 *For any c, s such that $s/c < 0.33$,* $\mathrm{PCP}_{c,s}[\log, 4] \subseteq \mathrm{P}$.

Theorem 1.1, 1.2 and 1.3 together yield some partial answer to the question posed earlier, since they exhibit some non-trivial values of q for which $q+1$ queries give more power than q:

[2]This question was posed to us by Oded Goldreich.

- For non-adaptive PCP, 4 queries are stronger than 3. (from Theorem 1.2 (1) and [16].)
- For adaptive PCP, 5 queries are stronger than 4. (from Theorem 1.2 (2) and Theorem 1.3.)
- 5 non-adaptive queries are stronger than 4 adaptive queries. (from Theorem 1.3 with $c = 1 - \varepsilon$ and the fact – proved in [15] – that $\mathrm{NP} \subseteq \mathrm{PCP}_{1-\varepsilon,1/4}[\log, 5]$ for any $\varepsilon > 0$.)

These results further highlight some aspects in which our understanding of PCPs are still not tight. We summarize this with the following open questions: (1) What is the smallest s for which $NP = \mathrm{naPCP}_{1,s}[\log, 3]$? (2) Is it true that for every q, there exist $s' < s$, such that $\mathrm{PCP}_{1,s}[\log, q] \subseteq \mathrm{P}$ while $\mathrm{NP} = \mathrm{PCP}_{1,s'}[\log, q+1]$? (3) If $\mathrm{NP} = \mathrm{PCP}_{c,s}[\log, q]$, then is it the case that for every $\varepsilon > 0$, $\mathrm{NP} = \mathrm{PCP}_{1,s/c+\varepsilon}[\log, q]$? Theorem 1.1 answers this question positively for $q = 3$, but leaves it open for higher values of q.

Organization Section 2 gives some background and describes the notion of recursive proof checking as needed for our constructions. Sections 3 and 4 describes the proof of Theorem 1.1. Section 5 describes the verifier of Theorem 1.2, Part (2). Highlights of the other PCP constructions are presented in Section 6 and certain containments of PCP classes in P are given in Section 7.

2 Background

Our PCP constructions rely on the proof-composition methodology introduced by Arora and Safra [2] and then refined in [1, 6, 7, 5, 10]. In this methodology one uses a verifier by Raz [12] and then "composes" it with an "inner verifier": the result of the composition is a PCP construction. This methodology is very useful since it reduces the task of constructing a PCP protocol to the simpler task of finding inner verifiers. In this section we give a brief overview of this methodology and we give a formal definition of an inner verifier.

We first review the properties of Raz's construction that will be used in this paper. Raz's construction of a 2-Prover 1-Round proof system is parameterized by an integer n, which should be thought of as a fixed large constant. In a 2-Prover 1-Round protocol the verifier has oracle access to *two* provers P_1 and P_2, and can ask one query to each oracle. Upon being queried, P_1 answers with a binary string of length n and P_2 answers with a binary string of length $3n$ (in contrast, a PCP verifier has oracle access to one proof oracle and can make a constant number of queries, receiving one bit for each query). The computations of Raz's verifier have a particularly nice structure. The verifier computes queries q_1 and q_2 for prover P_1 and P_2 respectively, and also a function $\pi : \{0,1\}^{3n} \to \{0,1\}^n$ and a set $S \subseteq \{0,1\}^{3n}$ (we omit details on how the queries, the function and the set are generated). Upon receiving the answers $a = P_1(q_1)$ and $b = P_2(q_2)$, where $a \in \{0,1\}^n$ and $b \in \{0,1\}^{3n}$, the verifier accepts iff $b \in S$ and $\pi(b) = a$. Let $\mathcal{P}_n$ denote the set of "projection functions" from $\{0,1\}^{3n}$ to $\{0,1\}^n$ which project exactly one coordinate out of $3i-2, 3i-1, 3i$ for any i, $1 \le i \le n$. Then for each q_1, the distribution of π given that q_1 is chosen is the uniform distribution in $\mathcal{P}_n$. The verifier has perfect completeness and soundness c^n, where $c < 1$ is an absolute constant.

Raz's verifier is used in PCP constructions in the following way. We encode each possible answer of the provers in Raz's protocol using a suitable error-correcting code. Additionally, we define a new verification procedure (which we call the *inner verifier*) that, given two strings A and B, the function π, and the set S, checks whether A is the encoding of an a and B is the encoding of a b such that $b \in S$ and $\pi(b) = a$. As an error correcting code we use the *long code* of Bellare et al. [5].

The long code of a string $a \in \{0,1\}^n$ is a string E_a (we let E stand for encoding) of length 2^{2^n} whose entries are indexed by the functions $f : \{0,1\}^n \to \{0,1\}$, such that $E_a(f) = f(a)$. The long code is extremely wasteful, but since a is of constant length, we can afford such a redundancy, and it will simplify the checking procedures. When representing all the answers of a pair of Raz provers using the long code, we will adopt some useful conventions. First of all, we represent Boolean values as elements of $\{1, -1\}$ rather than $\{0, 1\}$. The association is that -1 stands for 1 (or true) and 1 stands for 0 (or false), so that the xor operator becomes integer multiplication. In the following we will denote by $\mathcal{F}_n$ the set of functions $f : \{-1,1\}^n \to \{-1,1\}$; also n will always be the parameter in Raz's protocol; we will usually let $m = 3n$ be the length of an answer from prover P_2 in Raz's protocol. We say that a string A indexed by functions $f \in \mathcal{F}_n$ is *1-folded* if $A(f) = -A(-f)$ for every f. Codewords of the long code are 1-folded; in the rest of the paper we will restrict to 1-folded functions, this can be done without loss of generality using an access mechanism from [5]. A string B indexed by functions $g \in \mathcal{F}_m$ is S*-consistent*, where $S \subseteq \{0,1\}^m$, if $g_{|S} \equiv h_{|S}$ implies $B(g) = B(h)$. Notice that the long code of an element of S is S-consistent. An access mechanism described in [10] allows us to restrict without loss of generality to S-consistent B.

To sum up, we are considering a modification of Raz's protocol where all the answers of the two provers are encoded using the long code (call the encoded provers LP_1 and LP_2, where L stands for the use of the long code). We want to design a verifier that chooses q_1, q_2, S, π according to the same distribution of Raz's verifier, and then looks at the string $A = LP_1(q_1)$ and $B = LP_2(q_2)$; the access mechanism guarantees that A and B are 1-folded and B is S-consistent. Our goal is to test whether it is the case that A is the long code of some answer a and B is the long code of some b such that $b \in S$ and $\pi(b) = a$. An *inner verifier* is a testing procedure for this task. We clearly want to read as few bits as possible from A and B, and also have a good soundness. The definition of *soundness* for an inner verifier is somewhat tricky. Intuitively, we would like the inner verifier to reject with high probability whenever the conditions that we are testing are not satisfied (i.e. A is not a codeword of the long code, or B is not a codeword, or B is the encoding of some b and A is the encoding of some a but we have $\pi(b) \neq a$ and so on), but this is impossible to achieve without reading the tables entirely.

The right definition of soundness is that there be (possibly randomized) decoding procedures that can decode A and B, independently, into a pair of strings a and $b \in S$ such that $\pi(b) = a$, whenever the inner verifier accepts with sufficiently large probability. We will use a randomized decoding procedure D_n that on input a 1-folded string $A : \mathcal{F}_n \to \{-1,1\}$, returns an element $a = D_n(A) \in \{-1,1\}^n$ (formally, $D_n(A)$ is a random variable depending on the internal coin tosses of D_n). We will not describe D_n later (the description requires some definitions on Fourier analysis), but we anticipate that if B is S-consistent then $D_m(B) \in S$ with probability 1, and this allows us to concentrate solely on the "projection" condition $\pi(b) = a$ in the formal definition of a good inner verifier below.

Definition 1 (Good Inner Verifier) *An inner verifier V that has input $A : \mathcal{F}_n \to \{-1,1\}$, $B : \mathcal{F}_m \to \{-1,1\}$ and $\pi : \{-1,1\}^m \longrightarrow \{-1,1\}^n$ is a (c, s, q)-good inner verifier if the following properties hold: (1) V makes a total number of at most q queries to A and B; (2) If A is the long code of a, B is the long code of b, and $\pi(b) = a$, then $V(A, B, \pi)$ accepts with probability at least c* (completeness)*; (3) for each $\gamma > 0$, there exists a $\gamma' > 0$ that possibly depends on γ but is independent of n and m, such that for all 1-folded strings B, and all sets of 1-folded strings*

$\{A^\pi\}_{\pi\in\mathcal{P}_n}$,

$$\mathbf{Pr}[V(A^\pi, B, \pi) \textit{ accepts}] \geq s + \gamma \Longrightarrow \mathbf{Pr}[D_n(A^\pi) = \pi(D_m(B))] \geq \gamma'$$

where the probabilities are taken over the uniform distribution of π in $\mathcal{P}_n$ and over the internal coin tosses of V and D.

The above definition is based on the standard definition of an inner verifier (e.g. from [5]) but it incorporates the possibility that the decoding procedure be randomized[3] and also allows the soundness condition to be averaged over the choices of π. The latter is a technicality that is necessary to use some new results of Håstad [10], and it makes the definition less elegant as several details of the Raz verifier have to be taken into account explicitly (for example the fact that π is a projection, rather than an arbitrary function mapping $\{-1,1\}^m$ into $\{-1,1\}^n$, and also that π is selected uniformly). The following theorem can be proved using standard techniques.

Theorem 2.1 (Composition Theorem) *If there exists a (c,s,q)-good inner verifier, then for any $\varepsilon > 0$,* $\mathrm{NP} = \mathrm{PCP}_{c,s+\varepsilon}[\log, q]$. *Moreover, if there exists a (c,s,q)-good inner verifier that makes at most q queries non–adaptively, then* $\mathrm{NP} = \mathsf{na}\mathrm{PCP}_{c,s+\varepsilon}[\log, q]$.

3 The Adaptive 3-Query Protocol: A Motivating Discussion

Among the tasks of designing and analyzing PCP verifiers, the latter used to be the more challenging one, but the wide applicability of the new analysis techniques based on Fourier transforms are shifting much of the difficulty on the design phase. We will spend much of this section motivating the intuition behind our optimal protocol and describing the ideas that lead to it.

As defined in the previous section, the inner verification problem is: given π and given oracle access to 1-folded strings A and B, test whether there exists a b such that B is the encoding of b and A is the encoding of $\pi(b)$. Equivalently, we want to test whether for every f, g_1 and g_2, the following properties hold: (1) $A(f) = B(f\circ\pi)$; (2) $B(g_1 \wedge g_2) = B(g_1) \wedge B(g_2)$; (3) $B(g_1 g_2) = B(g_1)B(g_2)$. We will call A and B *consistent* if they satisfy the above properties.

Assume A and B are consistent, and suppose $A(f) = 1$: then $B(f\circ\pi) = 1$ and also for any h $B(f\circ\pi\wedge h) = B(f\circ\pi)\wedge B(h) = 1 \wedge B(h) = 1$. Similarly, if $A(f) = -1$ then $B(-f\circ\pi\wedge h) = 1$ for every h. So far we used only the first two properties of consistent strings A and B. Using also the third, we deduce that if A and B are consistent, then for any $f \in \mathcal{F}_n$ and any $g, h \in \mathcal{F}_m$.

$$\begin{array}{rlll} & A(f) = 1 & \text{implies} & B(g) = B(g(f\circ\pi\wedge h)) \\ \text{and} & A(f) = -1 & \text{implies} & B(g) = B(g(-f\circ\pi\wedge h)) \ . \end{array} \tag{1}$$

Checking this condition is essentially the *Monomial Basis Check* (MBC) of Bellare et al. [5], with the minor twist of looking at both A and B (Bellare et al. [5] would instead look only at B, and then test separately whether A is consistent with B). As a first proposal, we consider the test of Figure 1, which checks the Condition (1) for f, g and h chosen uniformly at random from their domain. It is possible to show that the MBC inner verifier is $(1, 3/4, 3)$-good, i.e., the soundness is

[3] Bellare et al. [5] used a deterministic procedure that returned a list of candidates, and this was conceptually similar to the randomized decoding idea that first appeared in [10]. A definition of inner verifier with respect to a randomized decoding procedure is explicit in [15].

Inner Verifier MBC (A, B, π)
Choose uniformly at random $f \in \mathcal{F}_n$, $g, h \in \mathcal{F}_m$
if $A(f) = 1$ **then accept** iff $B(g) = B(g(f \circ \pi \wedge h))$
if $A(f) = -1$ **then accept** iff $B(g) = B(g(-f \circ \pi \wedge h))$

Figure 1: The Inner Verifier based on the Monomial Basis Check of Bellare et al. [5].

Inner Verifier $\mathrm{BGS}_p(A, B, \pi)$
Choose uniformly at random $f \in \mathcal{F}_n$, $g, h \in \mathcal{F}_m$
With probability p **do**
if $A(f) = 1$ **then accept** iff $B(g) = B(g(f \circ \pi \wedge h))$
if $A(f) = -1$ **then accept** iff $B(g) = B(g(-f \circ \pi \wedge h))$
With probability $1 - p$ **do**
accept iff $A(f) = B(g)B(g(f \circ \pi))$

Figure 2: The Inner Verifier that combines Monomial Basis Check and Projection Test.

3/4. We omit the (not too hard) analysis, that is based on the techniques of [10]. The following argument shows that the analysis is tight: if A and B are inconsistent long codes then the MBC verifier accepts with probability 3/4, and our decoding procedure will have success probability zero when working with two inconsistent Long codes.

Since the worst case for the MBC verifier arises when A and B are individually correct but inconsistent, we try to patch the MBC test by adding another test that handles this case well. A good candidate for this "patching" role is the Projection Test, where f is taken uniformly from $\mathcal{F}_n$, g is taken uniformly from $\mathcal{F}_m$, and the test accepts iff $A(f) = B(g)B(g(f \circ \pi))$. Indeed, one can verify that if A and B are inconsistent long codes, then with probability 1/2 over the choices of f and g the Projection Test rejects. Combining the two verification procedures, we define the BGS_p verifier (see Figure 2) that is very similar to one used in [5]. Omitted calculations (this time they are quite hard) show that it is best to set $p = 1/3$ and that $\mathrm{BGS}_{1/3}$ is a $(1, 2/3, 3)$-good verifier, i.e. the soundness is 2/3. Again, the analysis can be shown to be tight: no setting of p can result in a verifier with soundness less than 2/3.

A second look at the BGS_p verifier reveals that the two possible tests to be executed are very related: if we pick $h \equiv -1$ instead that according to the uniform distribution, then the Projection Test coincides with the MBC test[4]. Therefore, we can view BGS_p in the following equivalent way: it first chooses to pick h according to one out of two possible distributions (i.e. either the uniform distribution on $\mathcal{F}_m$ or the deterministic choice of setting $h(b) = -1$ for every b), then it picks f and g uniformly and performs the MBC test. An alternative approach, that turns out to be much better, is to "shuffle" the distributions, and to skew the distribution of h pointwise. This gives rise to the definition of the B-MBC_p verifier (Figure 3, top). The analysis of the BGS_p verifier suggests

[4]Recall that B is 1-folded — otherwise the claim would not be true.

`Inner Verifier` B-MBC$_p(A, B, \pi)$ Choose uniformly at random $f \in \mathcal{F}_n$, $g \in \mathcal{F}_m$ Choose at random $h \in \mathcal{F}_m$ such that $\forall b \in \{-1, 1\}^m$. $\mathbf{Pr}[h(b) = 1] = p$ **if** $A(f) = 1$ **then accept** iff $B(g) = B(g(f \circ \pi \wedge h))$ **if** $A(f) = -1$ **then accept** iff $B(g) = B(g(-f \circ \pi \wedge h))$

`Inner Verifier` IV3$_\delta(A, B, \pi)$ Set $t = \lceil 1/\delta \rceil$, $\varepsilon_1 = \delta^2$ and $\varepsilon_i = e_{i-1}^c$ Choose $p \in \{\varepsilon_1, \ldots, \varepsilon_t\}$ uniformly at random Run B-MBC$_p(A, B, \pi)$

Figure 3: The B-MBC$_p$ verifier, a version of the MBC verifier where h is biased, and our final Inner Verifier IV3$_\delta$.

that it would be good to set $p = 1/6$ in B-MBC$_p$. Instead, it turns out that it is better to have p much smaller. We now see some details of the analysis.

Let A, B, π and p be fixed, and let $X = X_{A,B,\pi,p}$ be the random variable whose value is 1 when B-MBC$_p(A, B, \pi)$ accepts and 0 otherwise[5]. The acceptance probability of B-MBC$_p(A, B, \pi)$ is

$$\begin{aligned}
\mathbf{E}[X] &= \mathop{\mathbf{E}}_{f,g,h}\Bigg[\left(\frac{(1 + A(f))}{2}\right)\left(\frac{(1 + B(g)B(g(f \circ \pi \wedge h)))}{2}\right) \\
&\quad + \left(\frac{(1 - A(f))}{2}\right)\left(\frac{(1 + B(g)B(g(-f \circ \pi \wedge h)))}{2}\right)\Bigg]
\end{aligned}$$

Since A is folded, we always have $-A(f) = A(-f)$. Using the linearity of expectation and the fact that f and $-f$ are identically distributed in the uniform distribution, we get

$$\begin{aligned}
\mathbf{E}[X] &= 2 \mathop{\mathbf{E}}_{f,g,h}\left[\left(\frac{(1 + A(f))}{2}\right)\left(\frac{(1 + B(g)B(g(f \circ \pi \wedge h)))}{2}\right)\right] \\
&= \frac{1}{2} + \frac{1}{2} \mathop{\mathbf{E}}_{f,g,h}[B(g)B(g(f \circ \pi \wedge h))] + \frac{1}{2} \mathop{\mathbf{E}}_{f,g,h}[A(f)B(g)B(g(f \circ \pi \wedge h))] \qquad (2)
\end{aligned}$$

(there would also be a term $\frac{1}{2} \mathop{\mathbf{E}}_{f}[A(f)]$ that we omit since it is zero).

The expressions arising in (2) are extremely hard to bound, but we are fortunate that their analysis already appeared in the literature! Indeed they are the *same* expressions arising in a verifier that Håstad [10] constructs in order to prove a (tight) non-approximability result for satisfiable instances of MAX 3SAT. An equivalent description of the verifier of Håstad is the following: it asks the queries $A(f)$, $B(g)$ and $B(-g(f \circ \pi \wedge h))$, where f, g, h are generated as in B-MBC$_p$, then

- if $A(f) = 1$ it accepts iff $B(g) = -B(-g(f \circ \pi \wedge h))$;
- if $A(f) = -1$ it accepts no matter what is the value of $B(g)$ and $B(-g(f \circ \pi \wedge h))$.

[5]The sample space of $X_{A,B,\pi,p}$ is given by the possible choices of f, g, and h.

(see steps (2) and (3) in the definition of the $3S_\varepsilon$ verifier [10, Page 23] and the comments in [10, Remark 4.12].)

Håstad proves that the expectation of $B(g)B(g(f\circ\pi\wedge h))$ can be upper bounded by some arbitrarily small constant, for any B, and he also shows that whenever the expectation of $A(f)B(g)B(g(f\circ\pi\wedge h))$ is non-negligible, then also the probability of success of the decoding procedure is non-negligible. In order to bound the expectation of $B(g)B(g(f\circ\pi\wedge h))$, however, one cannot fix a particular p, but one has to pick p according to an appropriate distribution; also the bounds hold only if the expectation is also taken over π. There is a counter-example showing that the expressions of (2) cannot be bounded without going through such additional complications. For our purposes, it suffices to pick p according to the same distribution as in [10, Test $3S$, Page 29] and then apply the results proved therein. Our final verifier IV3_δ, described in Figure 3, is the same as B-MBC$_p$ except for the choice of p. The strange distribution of p is the particular one for which Håstad shows how to bound the expressions of (2). The constant c used in the definition of $\varepsilon_1,\ldots,\varepsilon_t$ is an absolute constant which is left unspecified in [10]. Using results from [10], specifically Lemmas 4.6 and 4.10, we can prove that the soundness of IV3_δ can be made arbitrarily close to 1/2 by choosing δ small.

4 Analysis of the 3-Query Protocol

In this section we will prove the following result.

Theorem 4.1 *For any $\delta > 0$, IV3_δ is a $(1, 1/2+2\delta, 3)$-good inner verifier.*

Theorem 1.1 follows from Theorem 4.1 and the Composition Theorem 2.1.

4.1 Fourier Transforms and the Long Code

We describe the basic definitions and machinery involved in the Fourier analysis of expressions arising out of the use of the long code. Recall that boolean functions take values in $\{-1,1\}$ throughout our discussion. We say that a function $A:\mathcal{F}_n\to\{-1,1\}$ is *linear* iff $A(f)A(g)=A(fg)$ for all $f,g\in\mathcal{F}_n$. There are 2^{2^n} linear functions l_α, one for each set $\alpha\subseteq\{-1,1\}^n$; it is defined as

$$l_\alpha(f)=\prod_{a\in\alpha} f(a)\ .$$

(By convention, we say that a product ranging over the empty set equals 1.)
We will be using the following three standard properties of linear functions:

$$l_\alpha(f)l_\beta(f)=l_{\alpha\Delta\beta}(f)\ ,\ \ l_\alpha(f)l_\alpha(g)=l_\alpha(fg)\ ,\ \ \mathop{\mathbf{E}}_f l_\alpha(f)=\begin{cases}1 & \text{If } \alpha=\emptyset\\ 0 & \text{otherwise.}\end{cases} \tag{3}$$

It is useful to see a function $A:\mathcal{F}_n\to\{-1,1\}$ as a real-valued function $A:\mathcal{F}_n\to\mathcal{R}$. The set of functions $A:\mathcal{F}_n\to\mathcal{R}$ is a vector space over the reals of dimension 2^{2^n}. We can define a scalar product between functions as

$$A\cdot B=\frac{1}{2^{2^n}}\sum_{f\in\mathcal{F}_n}A(f)B(f)=\mathop{\mathbf{E}}_f[A(f)B(f)]\ .$$

```
Decoding Procedure D_n(A);
   /* A : F_n → {−1, 1}, A 1-folded. */
      Choose α ⊆ {−1, 1}^n with probability Â_α^2.
      Pick an x ∈ α uniformly at random and return x.
```

Figure 4: The Decoding Procedure

The set of linear functions is easily seen to form an orthonormal basis for the set of functions $A : \mathcal{F}_n \to \{-1, 1\}$. This implies that for any function such A we have the *Fourier expansion*

$$A(f) = \sum_{\alpha} \hat{A}_\alpha l_\alpha(f), \tag{4}$$

where $\hat{A}_\alpha = A \cdot l_\alpha$ is the *Fourier coefficient* of A w.r.t α. Observe that for a function $A : \mathcal{F}_n \to \{-1, 1\}$, we have $-1 \leq \hat{A}_\alpha \leq 1$ for any α. Moreover, Parseval's identity implies that for every $A : \mathcal{F}_n \to \{-1, 1\}$, we have $\sum_\alpha \hat{A}_\alpha^2 = 1$.

It can be checked that (a proof of this appears in [10]) (i) if A is 1-folded, then $\hat{A}_\alpha = 0$ for all α with $|\alpha|$ even (in particular $A_\emptyset = 0$), and (ii) if A is S-consistent, then $\alpha \subseteq S$ for any α such that $\hat{A}_\alpha \neq 0$.

Given the definitions of Fourier coefficients, we are now ready to describe the decoding procedure that is used in the paper. This is done in Figure 4. We remark that the procedure is well-defined since Parseval's identity implies that $\hat{A}_\alpha^2$ in fact defines a probability distribution, and because the procedure will never get stuck by picking $\alpha = \emptyset$ since $\hat{A}_\emptyset = 0$.

4.2 Technical Lemmas

We now state two technical lemmas that were used in the soundness analysis of our inner verifiers of Section 3. The proofs may be found in [10] as Lemmas 4.6,4,11 and Lemma 4.10 respectively.

Lemma 4.1 ([10]) *If $t = \lceil \delta^{-1} \rceil$, $\varepsilon_1 = \delta^2$ and $\varepsilon_i = \varepsilon_{i-1}^c$ for $1 < i \leq t$, and $p \in \{\varepsilon_1, \cdots, \varepsilon_t\}$ is chosen uniformly at random, then for all 1-folded $B : \mathcal{F}_m \to \{-1, 1\}$*

$$\left| \mathop{\mathbf{E}}_{\pi,p,f,g,h} [B(g) B(g \cdot (f \circ \pi \wedge h))] \right| \leq 3\varepsilon_1^{1/2} + \frac{1}{t} \leq 4\delta$$

Lemma 4.2 ([10]) *For every $\gamma, p > 0$, there exists a constant $\delta = \delta_{\gamma,p} > 0$ that depends only on γ, p, such that for all 1-folded strings B and all sets of 1-folded strings $\{A^\pi\}_{\pi \in \mathcal{P}_n}$, if*

$$\left| \mathop{\mathbf{E}}_{\pi,f,g,h} [A^\pi(f) B(g) B(g \cdot (f \circ \pi \wedge h))] \right| \geq \gamma,$$

then $\mathop{\mathbf{Pr}}_{\pi} [D_n(A^\pi) = \pi(D_m(B))] \geq \delta$.

Inner Verifier B-V5_p (A, B, π)
Choose uniformly at random $f_1, f_2 \in \mathcal{F}_n$, $g \in \mathcal{F}_m$
Choose at random $h \in \mathcal{F}_m$ such that $\forall b \in \{-1,1\}^m. \mathbf{Pr}[h(b) = 1] = p$
Let $G_1 = g \cdot (f_1 \circ \pi \wedge f_2 \circ \pi \wedge h)$, $G_2 = g \cdot (f_1 \circ \pi \wedge -f_2 \circ \pi \wedge h)$
$G_3 = g \cdot (-f_1 \circ \pi \wedge f_2 \circ \pi \wedge h)$, $G_4 = g \cdot (-f_1 \circ \pi \wedge -f_2 \circ \pi \wedge h)$
if $A(f_1) = 1$ and $A(f_2) = 1$ **accept** iff $B(G_1) = B(G_2) = B(G_3)$
if $A(f_1) = 1$ and $A(f_2) = -1$ **accept** iff $B(G_1) = B(G_2) = B(G_4)$
if $A(f_1) = -1$ and $A(f_2) = 1$ **accept** iff $B(G_1) = B(G_3) = B(G_4)$
if $A(f_1) = -1$ and $A(f_2) = -1$ **accept** iff $B(G_2) = B(G_3) = B(G_4)$

Inner Verifier IV$5_\delta(A, B, \pi)$
Set $t = \lceil \delta^{-1} \rceil$, $\varepsilon_1 = \delta^2$ and $\varepsilon_i = \varepsilon_{i-1}^c$ for $1 < i \le t$.
Choose $p \in \{\varepsilon_1, \cdots, \varepsilon_t\}$ uniformly at random.
Run B-V5_p (A, B, π)

Figure 5: The 5 query adaptive inner verifier with a fixed p and the final inner verifier.

4.3 Proof of Theorem 4.1

The Proof: The statements about query complexity and completeness are clear, we verify the soundness claim. Recall that the probability of acceptance of IV3_δ is

$$\frac{1}{2} + \frac{1}{2} \mathop{\mathbf{E}}_{p,f,g,h}[B(g)B(g(f \circ \pi \wedge h))] + \frac{1}{2} \mathop{\mathbf{E}}_{p,f,g,h}[A(f)B(g)B(g(f \circ \pi \wedge h))].$$

By Lemma 4.1,

$$\left| \mathop{\mathbf{E}}_{\pi,p,f,g,h}[B(g)B(g \cdot (f \circ \pi \wedge h))] \right| \le 3\varepsilon_1^{1/2} + \frac{1}{t} \le 4\delta$$

and hence if $\mathbf{Pr}[V(A^\pi, B, \pi) \text{ accepts}] \ge 1/2 + 2\delta + \gamma$, we must have

$$\left| \mathop{\mathbf{E}}_{\pi,p,f,g,h}[A(f)B(g)B(g(f \circ \pi \wedge h))] \right| \ge 2\gamma$$

and invoking Lemma 4.2, we can conclude that $\mathop{\mathbf{Pr}}_{\pi}[D_n(A^\pi) = \pi(D_m(B))] \ge \eta$ where $\eta > 0$ depends upon γ but is *independent* of n, m. This concludes the proof that IV3_δ is a $(1, 1/2 + 2\delta, 3)$-good inner verifier. □

5 The Adaptive 5-Query Protocol

We now proceed to construct a PCP that makes 5 queries. The way to exploit the additional queries at our disposal is to look at two functions $f_1, f_2 \in \mathcal{F}_n$ instead of just a single $f \in \mathcal{F}_n$. In the 3 query protocol, the rationale used was that if $f(a) = 1$ then $(f \wedge h)(a) = 1$ for any h and similarly for the case when $f(a) = -1$. Similarly if $f_1(a) = f_2(a) = 1$, then we must have $(f_1 \wedge h)(a) = 1$ and $(f_2 \wedge h)(a) = 1$ for any h, and these two tests can be performed (instead of just one test

$(f \wedge h)(a) = 1$ as was done in the 3 query protocol). As one might expect, this method yields soundness of $(1/2)^2 = 1/4$ while making 5 (adaptive) queries (this construction follows the same idea of recycling one bit as in [15]). We now give a different test that gives no improvement over this test for the case of 5 adaptive queries, but which, however, has other applications (which shall be sketched in section 6) that the originally suggested one does not. We will use as basis for our test the following fact: if $f_1(a) = f_2(a) = 1$, then $(f_1 \wedge f_2 \wedge h)(a) = (f_1 \wedge -f_2 \wedge h)(a) = (-f_1 \wedge f_2 \wedge h)(a)$ for any h. Thus, we will able to perform two equality tests while reading five bits, so one expects that the soundness should be $(1/2)^2 = 1/4$ and indeed we shall prove that to be the case. In the actual protocol, however, f and h will belong to different spaces and this will be handled using the *projection function* π, and moreover, since we would like all queried bits (functions) to be unbiased, we will also xor these functions with an unbiased function g. This yields the inner verifier of Figure 5. As in the case of the 3 query protocol, we once again need to pick the bias p at random from a set of different possibilities for the analysis to work. This gives us the final inner verifier $\mathrm{IV5}_\delta$ of Figure 5.

We now analyze the soundness of $\mathrm{IV5}_\delta$. Let Y denote the indicator random variable for the acceptance of $\mathrm{IV5}_\delta$, so that the probability that $\mathrm{IV5}_\delta$ accepts is just the expectation of Y over the choices of pf_1, f_2, g, h. Arithmetization of the test yields terms of the form $\mathbf{E}[B(G_{j_1})B(G_{j_2})]$, $\mathbf{E}[A(f_i)B(G_{j_1})B(G_{j_2})]$, $\mathbf{E}[A(f_1)A(f_2)B(G_1)B(G_4)]$ and $\mathbf{E}[A(f_1)A(f_2)B(G_2)B(G_3)]$. After straightforward calculations and arguments of the form "$\mathop{\mathbf{E}}_{f_1,f_2,g,h}[B(G_1)B(G_2)] = \mathop{\mathbf{E}}_{f,g,h}[B(g)B(g \cdot (f \circ \pi \wedge h)]$ since the distribution of G_1, G_2 is identical to the one of $g, g \cdot (f \circ \pi \wedge h)$", we get

$$\begin{aligned}\mathbf{E}[Y] \;&=\; \frac{1}{4} + \frac{3}{4} \mathop{\mathbf{E}}_{p,f,g,h}[B(g)B(g \cdot (f \circ \pi \wedge h))] + \frac{1}{2} \mathop{\mathbf{E}}_{p,f,g,h}[A(f)B(g)B(g \cdot (f \circ \pi \wedge h))] \qquad (5)\\ &\quad - \frac{1}{4} \mathop{\mathbf{E}}_{p,f_1,f_2,g,h}[A(f_1)A(f_2)B(g \cdot (f_1 \circ \pi \wedge f_2 \circ \pi \wedge h))B(g \cdot (-f_1 \circ \pi \wedge -f_2 \circ \pi \wedge h))]\end{aligned}$$

The second term in the RHS above can be upper-bounded by 4δ using Lemma 4.1 as was done for the 3 query protocol, and if either of the last two terms has non-negligible absolute value, then the success probability of the randomized decoding strategies will also be non-negligible. The last claim follows from Lemma 4.2 and from Lemma 5.1 below.

Lemma 5.1 *For every $\gamma, p > 0$, there exists a constant $\delta = \delta_{\gamma,p} > 0$ that depends only on γ, p, such that for all 1-folded strings B and all sets of 1-folded strings $\{A^\pi\}_{\pi \in \mathcal{P}_n}$, if*

$$\left| \mathop{\mathbf{E}}_{\pi,f_1,f_2,g,h}[A^\pi(f_1)A^\pi(f_2)B(G_1)B(G_4)] \right| \geq \gamma,$$

then $\mathop{\mathbf{Pr}}_{\pi}[D_n(A^\pi) = \pi(D_m(B))] \geq \delta$.

Proof: Using the Fourier expansions of $A(f_i)$ and $B(G_j)$ as in Equation 4, the properties of linear functions (3), and using linearity of expectation, we transform the given expectation, for each fixed $\pi \in \mathcal{P}_n$, into

$$\sum_{\alpha_1,\alpha_2,\beta_1,\beta_2} \hat{A}_{\alpha_1}\hat{A}_{\alpha_2}\hat{B}_{\beta_1 abs}\hat{B}_{\beta_2 abs} \mathop{\mathbf{E}}_{f_1,f_2,g,h}\left[l_{\alpha_1}(f_1)l_{\alpha_2}(f_2)l_{\beta_1}(f_1 \circ \pi \wedge f_2 \circ \pi \wedge h)l_{\beta_2}(-f_1 \circ \pi \wedge -f_2 \circ \pi \wedge h)l_{\beta_1 \Delta \beta_2}(g)\right].$$

Since g is picked uniformly and independently at random, the inner expectation is 0 unless $\beta_1 = \beta_2 = \beta$ (say). Since f_1 and f_2 are also picked uniformly and independently at random, we can also conclude $\alpha_1, \alpha_2 \subseteq \pi(\beta)$. Some boolean algebra yields $(f_1 \circ \pi \wedge f_2 \circ \pi \wedge h) \cdot (-f_1 \circ \pi \wedge -f_2 \circ \pi \wedge h) = (-f_1 f_2 \circ \pi \wedge h)$. Incorporating all this, our expression simplifies to

$$\sum_{\substack{\beta \\ \alpha_1, \alpha_2 \subseteq \pi(\beta)}} \hat{A}_{\alpha_1} \hat{A}_{\alpha_2} \hat{B}^2_{\beta abs}\, \mathbf{E}\left[l_{\alpha_1}(f_1) l_{\alpha_2}(f_2) l_\beta(-f_1 f_2 \circ \pi \wedge h)\right].$$

Since $f_1(x), f_2(x)$ is chosen independently for different values of x, the inner expectation can be written as a product of expectations, one for each $x \in \pi(\beta)$. For $x \in \pi(\beta)$, define $\beta_x = \{y \in \beta : \pi(y) = x\}$. If there exists $x_0 \in \alpha_1 - \alpha_2$, then the expectation

$$E_{x_0} = \mathbf{E}\left[f_1(x_0) \prod_{y \in \beta_{x_0}} (-f_1(x_0) f_2(x_0) \wedge h(y))\right]$$

occurs as one of the factors of the inner expectation, and since this expectation E_{x_0} can easily be seen to be 0, the original expectation will be 0 as well. This implies $\alpha_1 \subseteq \alpha_2$, and similarly we get $\alpha_2 \subseteq \alpha_1$, yielding $\alpha_1 = \alpha_2$ whenever the inner expectation is non-zero. This further simplifies our expression to:

$$\begin{aligned}
&\sum_{\substack{\beta, \alpha \\ \alpha \subseteq \pi(\beta)}} \hat{A}^2_\alpha \hat{B}^2_{\beta abs} \prod_{x \in \alpha} \mathbf{E}\left[f_1(x) f_2(x) \prod_{y \in \beta_x} (-f_1(x) f_2(x) \wedge h(y))\right] \prod_{x \in \pi(\beta) - \alpha} \mathbf{E}\left[\prod_{y \in \beta_x} (-f_1(x) f_2(x) \wedge h(y))\right] \\
&\qquad = \sum_{\substack{\beta, \alpha \\ \alpha \subseteq \pi(\beta)}} \hat{A}^2_\alpha \hat{B}^2_{\beta abs} \prod_{x \in \alpha} \left(\frac{-1}{2} + \frac{(2\varepsilon - 1)^{\beta_x}}{2}\right) \prod_{x \in \pi(\beta) - \alpha} \left(\frac{1}{2} + \frac{(2\varepsilon - 1)^{\beta_x}}{2}\right) \\
&\leq \sum_{\substack{\beta, \alpha \\ \alpha \subseteq \pi(\beta)}} |\hat{A}_\alpha| \hat{B}^2_{\beta abs} \prod_{x \in \alpha} (-1)^{\beta_x + 1} \left(\frac{(-1)^{\beta_x}}{2} - \frac{(1 - 2\varepsilon)^{\beta_x}}{2}\right) \prod_{x \in \pi(\beta) - \alpha} (-1)^{\beta_x} \left(\frac{(-1)^{\beta_x}}{2} + \frac{(1 - 2\varepsilon)^{\beta_x}}{2}\right) \\
&\qquad = \sum_{\substack{\beta, \alpha \\ \alpha \subseteq \pi(\beta)}} |\hat{A}_\alpha| \hat{B}^2_{\beta abs} \prod_{x \in \alpha} \left(\frac{(-1)^{\beta_x}}{2} - \frac{(1 - 2\varepsilon)^{\beta_x}}{2}\right) \prod_{x \in \pi(\beta) - \alpha} \left(\frac{(-1)^{\beta_x}}{2} + \frac{(1 - 2\varepsilon)^{\beta_x}}{2}\right)
\end{aligned}$$

where the last step follows from the fact that all non-zero terms have $|\alpha|, |\beta|$ both odd (since A, B are 1-folded), and the power of (-1) multiplying the inner products can easily be seen to be $|\beta| + |\alpha|$. The last expression is the same as the one arising in Lemma 4.10 of [10], and if the expectation of this expression over the choice of π is at least γ, it has been shown in [10] that the decoding procedure has probability of success at least $\delta = \delta_{\gamma,p}$. □

The following theorem has a proof similar to the one of Theorem 4.1.

Theorem 5.1 *For any* $\delta > 0$*,* AIV5_δ *is a* $(1, 1/4 + 3\delta, 5)$*-good inner verifier.*

Theorem 1.2, Part (2), now follows from Theorem 5.1 and the Composition Theorem 2.1.

6 Non-adaptive PCP constructions

This section sketches how improved non-adaptive PCPs may be derived from our adaptive PCP constructions of previous sections. Employing the naive conversion of adaptive PCPs into non-adaptive ones by reading all possible bits already yields that IV3_δ (respectively IV5_δ) is a $(1, 1/2+$

Inner Verifier B-NAV5_p (A, B, π)

Choose uniformly at random $f_1, f_2 \in \mathcal{F}_n$, $g \in \mathcal{F}_m$

Choose at random $h \in \mathcal{F}_m$ such that $\forall b \in \{-1,1\}^m . \mathbf{Pr}[h(b) = 1] = p$

Let $G_1 = g \cdot (f_1 \circ \pi \wedge f_2 \circ \pi \wedge h)$

$G_2 = g \cdot (f_1 \circ \pi \wedge -f_2 \circ \pi \wedge h)$, $G_3 = g \cdot (-f_1 \circ \pi \wedge f_2 \circ \pi \wedge h)$

if $A(f_1) = 1$ and $A(f_2) = 1$ **accept** iff $B(G_1) = B(G_2) = B(G_3)$.

if $A(f_1) = 1$ and $A(f_2) = -1$ **accept** iff $B(G_1) = B(G_2)$.

if $A(f_1) = -1$ and $A(f_2) = 1$ **accept** iff $B(G_1) = B(G_3)$.

if $A(f_1) = -1$ and $A(f_2) = -1$ **accept** iff $B(G_2) = B(G_3)$.

Figure 6: The Inner Verifier B-NAV5_p

$2\delta, 4)$-good (respectively $(1, 1/4 + 3\delta, 6)$-good) non-adaptive inner verifier, which together with the Composition Theorem yields Parts (1) and (4) of Theorem 1.2.

To obtain a good 5 query non-adaptive inner verifier, we modify the non-adaptive version of B-V5_p (that reads 6 bits) to give the verifier B-NAV5_p (see Figure 6) that, instead of reading all $B(G_i)$ for $1 \leq i \leq 4$, does not read bit $B(G_4)$, and does not perform any of the tests involving $B(G_4)$. As usual, our final inner verifier NAIV5_δ runs B-NAV5_p after picking p at random from a suitable set. In the three cases when either $A(f_1) \neq 1$ or $A(f_2) \neq 1$ (or both), NAIV5_δ performs only one test, while when $A(f_1) = A(f_2) = 1$ it performs two tests. This implies that the soundness of NAIV5_δ is at least $3/4 \times 1/2 + 1/4 \times 1/4 = 7/16$ (which is the probability of accepting random, but inconsistent, long codes). One can, however, show, using the same techniques as that of Section 5, that this bound is achieved, and that NAIV5_δ is a $(1, \frac{7}{16} + \frac{9\delta}{4}, 5)$-good non-adaptive inner verifier, thereby proving Part (3) of Theorem 1.2.

7 Weakness Results for PCPs

In this section, we prove that PCP classes with certain query complexity and error probability are *weak* in the sense that they can only capture languages in P. We achieve this goal by providing approximation algorithms for constraint satisfaction problems and then invoking the following result.

Fact 1 ([13]) *If there exists a polynomial time factor r approximation algorithm for MAX kCSP, then* $\mathrm{PCP}_{c,s}[\log, k] \subseteq \mathrm{P}$ *for any $s/c < r$.*

Existing approximation algorithms in [13, 14, 16] for various MAX kCSP problems therefore imply that $\mathrm{PCP}_{c,s}[\log, 3] \subseteq \mathrm{P}$ for any $s/c < 1/2$, $\mathsf{na}\mathrm{PCP}_{1,s}[\log, 3] \subseteq \mathrm{P}$ for any $s < 5/8$, $\mathrm{PCP}_{c,s}[\log, k] \subseteq \mathrm{P}$ for any $s/c < 2^{1-k}$, and lastly $\mathsf{na}\mathrm{PCP}_{1,s}[\log, k] \subseteq \mathrm{P}$ for any $s < (k+1)/2^k$.

Our first result relies on a semidefinite programming relaxation of MAX 4CSP using a methodology of Karloff and Zwick [11, 16] and a numerical analysis of the approximation ratio guaranteed by rounding the SDP solution using a random hyperplane with certain probability and using a random assignment to the variables with the remaining probability.

Theorem 7.1 *There exists a polynomial time algorithm for approximating MAX 4CSP within a factor of 0.33 of the optimum.*

Corollary 7.1 *For any $s/c < 0.33$, we have* $\mathrm{PCP}_{c,s}[\log, 4] \subseteq \mathrm{P}$.

By the above corollary adaptive PCPs making 4 queries cannot achieve a soundness of 0.33 if they have near-perfect completeness; however, a 5–query non–adaptive PCP construction with near-perfect completeness and soundness 0.25 is known [15]. Thus, we exhibit the first instance where adaptive PCPs are strictly less powerful than non-adaptive PCPs that just make one extra query (unless $\mathrm{P} \neq \mathrm{NP}$).

Finally, we obtain limitations on the power of k query PCPs for larger values of k, for the special cases of perfect and near-perfect completeness.

Theorem 7.2 *For an $\varepsilon > 0$,* $\mathrm{PCP}_{1,3/(2^k+1)-\varepsilon}[\log, k] \subseteq \mathrm{P}$.

Proof: Consider a (possibly adaptive) PCP verifier V that makes k queries, has perfect completeness, has soundness less than $3/(2^k+1)$, and recognizes a language L. We show how to, in polynomial time, construct a proof that will be accepted by V with probability at least $3/(2^k+1)$ for inputs $x \in L$. This is achieved by considering a random proof and a proof constructed by finding a satisfying assignment to an appropriately constructed satisfiable 2SAT instance, and taking the "better" of the two proofs. Note that this yields the following polynomial time procedure to decide membership of x in L: construct the afore-mentioned proof in polynomial time (if construction of the proof fails say because the 2SAT instance was not satisfiable, then we know that $x \notin L$, so reject x immediately) and then accept x iff the proof will be accepted with probability at least $3/(2^k+1)$ (since V has logarithmic randomness this can be checked in polynomial time).

For any particular choice R of the random bits of V, the computation of V can be modeled as a decision tree T_R of depth at most k. Let m_1 (respectively m_2) denote the fraction of R's for which T_R has exactly one accepting "leaf" (respectively exactly two accepting leaves). Let $m_3 = 1 - m_1 - m_2$ be the fraction of R's for which T_R has at least three accepting leaves.

It is easy to see that the probability that a decision tree T_R with exactly l accepting leaves accepts a random proof equals $\frac{l}{2^k}$, and hence the probability that V accepts a random proof is at least $\frac{(m_1+2m_2+3m_3)}{2^k}$. (This is because each decision tree of V has the property that configurations corresponding to two distinct leaves can never occur simultaneously: and hence the events of the various accepting leafs of a particular tree occurring are mutually exclusive.)

For the case when $x \in L$, consider the following proof: For each tree T_R with just one accepting leaf, create unary clauses consisting of the (at most k) literals which when set to true will make T_R accept (these literals will correspond to the variables on the path from the root to the unique accepting leaf in T_R). For each tree T_R with exactly two accepting leaves, there must exist a variable x such that both the left and right subtrees of the tree rooted at x have exactly one accepting leaf. Create unary clauses as before for the variables (other than x) which occur in the path from the root of T_R to x, and if $l_1, l_2, \ldots, l_p$ and $r_1, r_2, \ldots, r_q$ are the literals which when set to true will make T_R accept in the cases when $x = 0$ and $x = 1$ respectively, create the (at most $2k$) clauses $\bar{x} \Rightarrow l_1, \ldots, \bar{x} \Rightarrow l_p$ and $x \Rightarrow r_1, \ldots, x \Rightarrow r_q$.

Since V has perfect completeness and $x \in L$, the 2CNF formula $\mathcal{C}$ comprising all clauses so created must be satisfiable, and using the polynomial time algorithm for 2SAT, a satisfying assignment σ for $\mathcal{C}$ can be found. Now, let Π be the proof that agrees with σ on all variables (bits) occurring in the clauses of $\mathcal{C}$, and arbitrary elsewhere. Clearly, the probability that V accepts Π is at least the probability that V chooses an R such that T_R has at most 2 accepting leaves, which is $m_1 + m_2$.

Combining the bounds on acceptance probability for a random proof and Π, we get that, in the case when $x \in L$, we can, in polynomial time, construct a proof that is accepted by V with probability at least

$$\min_{m_1,m_2,m_3:m_1+m_2+m_3=1} \max\left\{m_1+m_2, \frac{m_1+2m_2+3m_3}{2^k}\right\} = \frac{3}{2^k+1}.$$

Theorem 7.3 *We have, for any* $\varepsilon > 0$, $\mathrm{PCP}_{1-\varepsilon,s}[\log, k] \subseteq \mathrm{P}$, *where* $s = \frac{3}{2^k+2} - O(\frac{k\varepsilon^{1/3}}{2^k+2})$.

Proof:

The idea is the same as the above proof, the only difference is that the 2CNF formula $\mathcal{C}$ created as above will not in general be satisfiable since V no longer has perfect completeness. If V has completeness $1-\varepsilon$, however, it is easy to check that at least a fraction $1-\varepsilon/(m_1+m_2)$ of the clauses in $\mathcal{C}$ will be satisfiable, and hence we can run the polynomial time algorithm of Zwick [16] for nearly satisfiable instances of 2SAT to find an assignment σ satisfying at least $1 - O((\varepsilon/(m_1+m_2))^{1/3})$ of the clauses in $\mathcal{C}$. As before, considering a random proof and a proof Π constructed so that it agrees with σ on all its variables and is arbitrary elsewhere, will imply, after a few simple calculations, that the soundness of the verifier cannot be strictly less than

$$\min_{m_1,m_2,m_3:m_1+m_2+m_3=1} \max\left\{m_1+m_2 - O(k\varepsilon^{1/3}), \frac{m_1+2m_2+3m_3}{2^k}\right\} = \frac{3}{2^k+2} - O(\frac{k\varepsilon^{1/3}}{2^k+2}).$$

8 Free bits and our PCP constructions

Informally, a PCP system with query complexity q is said to have *free bit complexity* f for some $f \le q$ if after reading some f bits, the verifier can uniquely determine the values of the remaining $q - f$ query bits that will make it accept. The notation $\mathrm{FPCP}_{c,s}[\log, f]$ stands for a PCP system with logarithmic randomness, completeness c, soundness s and free bit complexity f. The free bit complexity of a PCP is an extremely important one since it has direct applications to proving the hardness of approximating Vertex Cover as is formalized in the following lemma:

Lemma 8.1 ([5]) *If* $\mathrm{NP} \subseteq \mathrm{FPCP}_{c,s}[\log, f]$, *then approximating Vertex Cover up to a factor of* $1 + \frac{c-s}{2^f - c} - \varepsilon$ *is* NP*-hard for any* $\varepsilon > 0$.

The best hardness result known for vertex cover is $7/6 - \varepsilon$, for any $\varepsilon > 0$, and is obtained using the above lemma with the two free bit PCP construction of Håstad [10], which has completeness $1 - \varepsilon$ and soundness $1/2$, for any $\varepsilon > 0$. We now prove that we can achieve the same soundness while also guaranteeing perfect completeness, thereby answering in the affirmative a question raised in [5].

Theorem 8.1 *For any* $\varepsilon > 0$, *we have* $\mathrm{NP} \subseteq \mathrm{FPCP}_{1,1/2+\varepsilon}[\log, 2]$.

Proof: We just observe that the inner verifier $\mathrm{IV3}_\delta$ uses just two free bits. This is because, once $A(f)$ and $B(g)$ are read, the verifier "knows" the values it expects for the other bits it reads. The theorem now follows employing the soundness bound noted in Theorem 4.1. $\Box$

By lemma 8.1, we are able to match the best known hardness result for approximating Vertex Cover while only using perfect completeness, indicating that non–perfect completeness is not inherent at least for the best known result today on Vertex Cover. We also note the following result:

Theorem 8.2 *For any $\varepsilon > 0$, we have* $\mathrm{NP} \subseteq \mathrm{FPCP}_{1,1/4+\varepsilon}[\log, 3]$.

Proof: Follows from the fact that the inner verifier $\mathrm{IV5}_\delta$ uses only 3 free bits. $\Box$

Acknowledgments

We would like to thank Uri Zwick for having provided us with the code he used in [11, 16]. We are grateful to Oded Goldreich for several valuable discussions, and for comments on a preliminary version of this paper.

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. Preliminary version in *Proc. of FOCS'92.*

[2] S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. Preliminary version in *Proc. of FOCS'92.*

[3] L. Babai, L. Fortnow, L. Levin, and M. Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd ACM Symposium on Theory of Computing*, pages 21–31, 1991.

[4] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991. Preliminary version in *Proc. of FOCS'90.*

[5] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCP's and non-approximability – towards tight results. *SIAM Journal on Computing*, 27(3):804–915, 1998. Preliminary version in *Proc. of FOCS'95.*

[6] M Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient probabilistically checkable proofs and applications to approximation. In *Proceedings of the 25th ACM Symposium on Theory of Computing*, pages 294–304, 1993. See also the errata sheet in *Proc of STOC'94.*

[7] M. Bellare and M. Sudan. Improved non-approximability results. In *Proceedings of the 26th ACM Symposium on Theory of Computing*, pages 184–193, 1994.

[8] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996. Preliminary version in *Proc. of FOCS91*.

[9] U. Feige and J. Kilian. Two prover protocols - low error at affordable rates. In *Proceedings of the 26th ACM Symposium on Theory of Computing*, pages 172–183, 1994.

[10] J. Håstad. Some optimal inapproximability results. In *Proceedings of the 29th ACM Symposium on Theory of Computing*, pages 1–10, 1997.

[11] B. Karloff and U. Zwick. A $(7/8-\varepsilon)$-approximation algorithm for MAX 3SAT? In *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science*, 1997.

[12] R. Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998. Preliminary version in *Proc. of STOC'95*.

[13] L. Trevisan. Positive linear programming, parallel approximation, and PCP's. In *Proceedings of the 4th European Symposium on Algorithms*, pages 62–75. LNCS 1136, Springer-Verlag, 1996.

[14] L. Trevisan. Approximating satisfiable satisfiability problems. In *Proceedings of the 5th European Symposium on Algorithms*, pages 472–485. LNCS 1284, Springer-Verlag, 1997.

[15] L. Trevisan. Recycling queries in PCPs and in linearity tests. In *Proceedings of the 30th ACM Symposium on Theory of Computing*, 1998.

[16] U. Zwick. Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, 1998.