Small PCPs With Low Query Complexity

by

Prahladh Harsha

Bachelor of Technology (Computer Science and Engineering),
Indian Institute of Technology, Madras, India. (1998)

Submitted to the
Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY.

June 2000

Author ______________________________
Department of Electrical Engineering and Computer Science
May 19, 2000

Certified by ______________________________
Madhu Sudan
Associate Professor of Computer Science and Engineering
Thesis Supervisor

Accepted by ______________________________
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

Small PCPs With Low Query Complexity

by

Prahladh Harsha

Submitted to the
Department of Electrical Engineering and Computer Science
on May 19, 2000 in partial fulfillment of the
requirements for the degree of Master of Science.

Abstract

Most known constructions of probabilistically checkable proofs (PCPs) either blow up the proof-size by a large polynomial, or have a high (though constant) query complexity. In this thesis we give a transformation with slightly-super-cubic blowup in proof-size and a low query complexity. Specifically, the verifier probes the proof in 16 bits and rejects every proof of a false assertion with probability arbitrarily close to $\frac{1}{2}$, while accepting corrects proofs of theorems with probability one. The proof is obtained by revisiting known constructions and improving numerous components therein. In the process we abstract a number of new modules that may be of use in other PCP constructions.

Thesis Supervisor: Madhu Sudan
Title: Associate Professor of Computer Science and Engineering

Acknowledgments

I would like to thank my advisor Madhu Sudan for his invaluable guidance and support and for always being there to discuss and clarify any matter.

Research supported in part by NSF Grant CCR-9875511.

I owe a lot to the research atmosphere at LCS. I also thank all the members of the Theory Group in LCS for having created a very conducive atmosphere for me to work in. I would like to make a special mention of the LCS Library for their excellent facilities. I would like to thank all my friends for their wonderful company, during my stay in Boston, which helped me get over my occasional blues. I am thankful to all others who have been either directly or indirectly involved in the completion of this thesis.

Last but certainly not the least, I would like to thank amma, appa, Pavithra and Ramesh.

Contents

CHAPTER 1

Introduction

The notion of proof verification is fundamental in Complexity Theory. One of the basic complexity classes NP is defined in terms of proof verification. NP is precisely the class of languages for which there exists a short (polynomially long) proof of membership that can be verified by a polynomial time deterministic algorithm (called a *verifier*). While studying proof verification, one comes across several questions.

- How efficiently can the proof be verified?
- Do all the bits of the proof have to be read to verify it?
- Should the verifier be deterministic?

Attempts to answer these questions lead to the notion of Probabilistically Checkable Proofs (PCP). It is known today that proofs can be written in such a fashion that the verifier can merely check the proof at a few selected places and be assured of its validity in the case of correct proofs and find flaws in case of false proofs. If the verifier were deterministic, it is clear that it must read all the bits of the proof to convince itself of the validity of the proof. However, this need not be the case if the verifier adopts a randomized strategy, i.e., the verifier could possibly throw a few random coins, decide to read the proof at a few selected places and still find flaws in a false proof with fairly high probability.

1.1 Probabilistically Checkable Proofs

Informally, a PCP system for a language L consists of a verifier which is a probabilistic Turing Machine that has to check the membership of an input string x in the language L. The verifier has oracle access to a (binary) proof π which supposedly contains the proof of the statement "$x \in L$". The verifier checks membership by tossing a certain number of random coins, decides to check the proof at a few bit positions and accepts or rejects the proof π based on a boolean verdict depending on the input string x, the random coins tossed and the bits read by it.

More formally the verifier in a PCP system is defined as follows:

Definition 1.1.1 *For functions $r, q : \mathbb{Z}^+ \to \mathbb{Z}^+$, a probabilistic oracle machine (or verifier) V is called a (r, q)-restricted verifier if on input x of length n, the verifier V tosses at most $r(n)$ random coins to obtain the random string R and queries an oracle π for at most $q(n)$ bits. It then computes a Boolean verdict based on x, R and the bits read from the proof π and accepts or rejects the proof according to the Boolean verdict. We denote this decision by $V^\pi(x; R)$.*

The parameters $r(n), q(n)$ quantify the complexity of the verification procedure and we hence expect them to be small (compared to n) so that the verification is efficient. We then have to characterize the performance of the verifier by two other parameters (a) *Completeness:* This is the probability with which the verifier accepts correct proofs, when $x \in L$. (b) *Soundness:* This is the maximum probability with which the verifier accepts purported proofs when $x \notin L$. The choice of these parameters decides the class of languages accepted by restricted verifiers behaving according to those parameters. Or more formally,

Definition 1.1.2 *A language L is said to be in the class PCP$_{c,s}[r, q]$ if there exists an (r, q)-restricted verifier that satisfies the following properties on input x.*

Completeness *If $x \in L$ then there exists π such that V on oracle access to π accepts with probability at least c (i.e., $\exists \pi$ such that $\Pr_R[V^\pi(x; R) = \text{accept}] \geq c$.)*

Soundness *If $x \notin L$ then for every oracle π, the verifier V accepts with probability strictly less than s (i.e., $\forall \pi$, $\Pr_R[V^\pi(x; R) = \text{accept}] < s$.)*

In the case when $c = 1$ and $s = 1/2$, we omit the subscripts c, s and refer to the corresponding class as just PCP$[r, q]$.

1.2 PCPs - A Brief History

In this section, we shall give a brief history of the results in the area of PCPs.[1]

Interactive Proof systems (IP) were introduced independently by Goldwasser, Micali and Rackoff [17] and Babai [4]. Ben-Or, Goldwasser, Kilian and Wigderson [10] then extended the notion of interactive proof system to multiple provers and defined the concept of multiprover interactive

[1]The study of PCPs is very involved and we stress that there are far too many beautiful results for us to do do justice by citing all of them.

proofs (MIP)[2]. Fortnow, Rompel and Sipser [15] then showed that MIP so defined is exactly equal to PCP$[\mathrm{poly}, \mathrm{poly}]$ (using the terminology of PCPs). In fact the model underlying today's PCP systems is the "oracle model" introduced by Fortnow, Rompel and Sipser [15].

The central result early in this area is that of Babai, Fortnow and Lund [6]. They showed that MIP $=$ NEXP (i.e., the class of languages recognizable in non-deterministic exponential time) This result combined with that of Fortnow, Rompel and Sipser [15] shows that NEXP $\subseteq$ PCP$[\mathrm{poly}, \mathrm{poly}]$. This important connection between NEXP and PCPs was scaled down to the NP-level by two separate groups. Babai, Fortnow, Levin and Szegedy [5] showed that there exist PCPs (called holographic proofs in their result) for NP in which it is possible to verify the correctness of the proof in poly-logarithmic time. The seminal result indicating the intricate relationship between PCP systems and hardness of approximations was made by Feige, Goldwasser, Lovász, Safra and Szegedy [12]. The definition of PCPs is implicit in their result and they show that $\mathrm{NP} \subseteq$ PCP$[O(\log n \log\log n), O(\log n \log\log n)]$. They also make the remarkable observation that $\mathrm{NP} \subseteq$ PCP$_{1,s}[r, q]$ implies that approximating the maximum clique in a $2^{r(n)+q(n)}$-vertices graph to within a factor of $1/s$ factor is infeasible[3] unless $\mathrm{NP} \subseteq$DTIME$(2^{O(r+q)})$.

This hardness connection of PCPs spurred a lot of research into strengthening the parameters of PCPs to prove stronger hardness assumptions. Arora and Safra [2] set the stage for results in this direction. They proved that $\mathrm{NP} =$ PCP$[O(\log n), O(\sqrt{\log n})]$. They also made explicit the definition of PCPs, the hierarchy of classes PCP$_{c,s}[r, q]$ and their dependence on the parameters $r(n), q(n)$. Their proof introduced the notion of *recursive proof checking* (also called *proof composition*) Proof Composition has played a vital role in all subsequent constructions of PCPs. They also provide the first strong NP-hardness result for MaxClique (a factor of $2^{\sqrt{\log n}}$). Arora, Lund, Motwani, Sudan and Szegedy [1] then showed how to reduce the query complexity to constant while preserving the logarithmic randomness, i.e., they showed that $\mathrm{NP} =$ PCP$[O(\log n), O(1)]$. This result implies that Max3Sat is NP-hard to approximate within some constant factor and so is any MaxSNP hard problem. It also implied the NP-hardness of approximating MaxClique within n^{ϵ}, for some $\epsilon > 0$.

Constant prover proof systems have been very useful both in the construction of PCPs as well as in the derivation of inapproximability results. There are used in the penultimate step of recursive proof composition. Informally, a constant prover proof system of one round consists of a verifier which makes a few random coin tosses, queries a constant number of provers, each of which respond with answers of a certain size (not necessarily a bit long). The verifier then accepts or rejects based on the responses of the provers. Two-prover proof systems with poly-logarithmic randomness and answer size were introduced by Lapidot and Shamir [20] and Feige and Lovász [14]. Arora, Lund, Motwani, Sudan and Szegedy [1] reduced the randomness to logarithmic at the ex-

[2]We shall formally define MIPs in Chapter 2

[3]infeasible here means not doable in polynomial time.

pense of number of provers (still a constant). Bellare, Goldwasser, Lund and Russell [8] attain the same randomness but reduce answer size to sublogarithmic with just 4 provers. Feige and Kilian [13] then constructed 2-prover proof systems with logarithmic randomness and constant answer size. A breakthrough result in this area is Raz' parallel repetition theorem which shows the existence of 2-prover proof systems with logarithmic randomness and constant answer size [24].

It is to be noted that in the above results, proof-size is not a parameter that has been optimized. Proof-sizes were considered in Babai, Fortnow, Levin and Szegedy [5] and Polishchuk and Spielman [23]. With respect to proof-size, Polishchuk and Spielman [23, 28] attain the optimal result; they show how a PCP can be constructed with just a blowup of $n^{1+\epsilon}$ in the proof-size for any $\epsilon > 0$. *Long Code* as an important error-correcting code to be used in the ultimate step of proof composition was first introduced in Bellare, Goldreich and Sudan [7]. With respect to query complexity, a sequence of results [6, 5, 12, 2, 1, 8, 13, 9, 24, 7, 18] finally culminated in Håstad's beautiful result [19] that every language in NP has a PCP with query complexity 3 and soundness arbitrarily close to 1/2. This query complexity is tight with respect to soundness 1/2. Håstad in his results also describes a "Fourier Analysis" technique which can potentially be used to give the tight analysis of any verifier.

1.3 Our Main Results

Constructions of efficient probabilistically checkable proofs (PCP) have been the subject of active research in the last ten years. As mentioned in the earlier section, Arora, Lund, Motwani, Sudan and Szegedy [1] showed that it is possible to transform any proof into a probabilistically checkable one of polynomial size, such that it is verifiable with a constant number of queries. Valid proofs are accepted with probability one, while any purported proof of an invalid assertion is rejected with probability $1/2$. Neither the proof-size, nor the query complexity is explicitly described there; however the latter is estimated to be around 10^6.

Subsequently much success has been achieved in improving the parameters of PCPs, constructing highly efficient proof systems either in terms of their size or their query complexity. The best result in terms of the former is a result of Polishchuk and Spielman [23]. They show how any proof can be transformed into a probabilistically checkable proof with only a mild blowup in the proof-size, of $n^{1+\epsilon}$ for arbitrarily small $\epsilon > 0$ and that is checkable with only a constant number of queries. This number of queries however is of the order of $O(1/\epsilon^2)$, with the constant hidden by the big-Oh being some multiple of the query complexity of [1]. On the other hand, Håstad [19] has constructed PCPs for arbitrary NP statements where the query complexity is a mere three bits (for completeness almost 1 and soundness $1/2$). However the blowup in the proof-size of Håstad's PCPs has an exponent proportional to the query complexity of the PCP of [1]. Thus neither of these "nearly-optimal" results provides simultaneous optimality of the two parameters. It is reasonable to wonder if this inefficiency in the combination of the two parameters is inherent; and this thesis is motivated by this question.

We examine the size and query complexity of PCPs jointly and obtain a construction with reasonable performance in both parameters. The only previous work that mentions the joint size vs. query complexity of PCPs is a work of Friedl and Sudan [16], who indicate that NP has PCPs with nearly quadratic size complexity and in which the verifier queries the proof for 165 bits. The main technical ingredient in their proof was an improved analysis of the "low-degree test". Subsequent to this work, the analysis of low-degree tests has been substantially improved. Raz and Safra [25] and Arora and Sudan [3] have given highly efficient analysis of different low-degree tests. Furthermore, techniques available for "proof composition" have improved, as also have the construction for terminal "inner verifiers". In particular, the work of Håstad [19], has significantly strengthened the ability to analyze inner verifiers used at the final composition step of PCP constructions.

In view of these improvements, it is natural to expect the performance of PCP constructions to improve. Our work confirms this expectation. However, our work exposes an enormous number of complications in the natural path of improvement. We resolve most of these, with little loss in performance and thereby obtain the following result: Satisfiability has a PCP verifier that makes at most 16 oracle queries to a proof of size at most $n^{3+o(1)}$, where n is the size of the instance of satisfiability. Satisfiable instances have proofs that are accepted with probability one, while unsatisfiable instances are accepted with probability arbitrarily close to $1/2$. (See Theorem 2.3.1.)

We also raise several technical questions whose positive resolution may lead to a PCP of nearly quadratic size and query complexity of 6. Surprisingly, no non-trivial limitations are known on the joint size + query complexity of PCPs. In particular, it is open as to whether nearly linear sized PCPs with query complexity of 3 exist for NP statements.

While our principal interest is in the size of a PCP and not in the randomness, it is well-known that the size of a probabilistically checkable proof (or more precisely, the number of distinct queries to the oracle π) is at most $2^{r(n)+q(n)}$. Thus the size is implicitly governed by the randomness and query complexity of a PCP. The main result of this thesis is the following.

Theorem 1.3.1 *For every $\varepsilon, \mu > 0$,*

$$\mathrm{SAT} \in PCP_{1,\frac{1}{2}+\mu}\left[(3+\varepsilon)\log n, 16\right].$$

Remark: Actually the constants ε and μ above can be replaced by some $o(1)$ functions; but we don't derive them explicitly.

It follows from the parameters that the associated proof is of size at most $O(n^{3+\varepsilon})$.

Cook [11] showed that any language in NTIME$(t(n))$ could be reduced to SAT in $O(t(n)\log t(n))$ time such that instances of size n are mapped to boolean formulae of size at most $O(t(n)\log t(n))$.

Combining this with Theorem 2.3.1, we have that every language in NP has a PCP with at most a slightly super-cubic blowup in proof-size and a query complexity as low as 16 bits.

In the process of proving the above theorem, we present a algebraic problem called the *polynomial constraint satisfaction* and exhibit the NP-hardness of this problem (see Lemma 3.2.2). We show that for every $\epsilon > 0$, it is NP-hard to distinguish between instances of this problem in which all the constraints are satisfiable and those in which at most ϵ-fraction of the constraints are satisfiable. We show that SAT is reducible to this problem with a minimal blowup in the proof-size. This problem and the accompanying result are neat algebraic formulations and are easily amenable to future PCP constructions.

Håstad's PCPs ([19]) have a terminal inner-verifier which convert 2-prover canonical MIPs to PCPs with 3 queries. We build a similar inner verifier for p prover non-canonical MIPs. This is the first time that such a verifier has been constructed for MIPs with more than 2 provers. Our analysis shows surprising complications and forces us to use a large number (seven) of extra bits to effect the final truncation in the PCP construction.

1.4 Organization of the Thesis

We present the Main Theorem (Theorem 2.3.1) and its proof in Chapter 2. We divide the task of proving the theorem into 3 lemmas, which we prove in the subsequent chapters. We present the Polynomial Constraint Satisfaction problem and analyze its hardness in Chapter 3. In Chapter 4, we work the Low-Degree Test of Raz and Safra [25] into a form that is convenient for us to work with. In Chapter 5, we construct a 3-prover MIP for SAT which is efficient in terms of randomness. In Chapter 6, we present a constant bit verifier for MIPs, which is used in the final step of the recursion in the proof of the Main Theorem. Finally, in chapter 7, we make a few concluding remarks and suggest possible approaches for improvements in the joint size-query complexity of PCPs.

CHAPTER 2

Main Theorem

In this chapter, we present the main theorem and its proof, modulo the proof of two lemmas which we shall prove in the subsequent chapters. As mentioned in Chapter 1, the parameters we seek are such that no existing proof system achieves them. Hence we work our way through the PCP construction of Arora, Lund, Motwani, Sudan and Szegedy [1] and make every step as efficient as possible. The key ingredient in their construction (as well as most subsequent constructions) is the notion of recursive composition of proofs. Informally, recursive proof composition takes an "outer verifier" that is efficient in its use of randomness, but inefficient in query complexity; combines it with an "inner verifier" that is inefficient in its use of randomness but efficient in its query complexity; and obtains a composed verifier that is efficient in both the query and the randomness complexity. The paradigm of recursive composition is best described in terms of multi-prover interactive proof systems (MIPs).

2.1 MIP and Recursive Proof Composition

Definition 2.1.1 *For integer p, and functions $r, a : \mathbb{Z}^+ \to \mathbb{Z}^+$, an MIP verifier (probabilistic oracle machine) V is called a (p, r, a)-restricted if it interacts with p mutually-non-interacting provers $\pi_1, \ldots, \pi_p$ in the following restricted manner. On input x of length n, V picks a random $r(n)$-bit string R and generates p queries $q_1, \ldots, q_p$ and a linear sized circuit C. The verifier then issues query q_i to prover π_i. The provers respond with answers $a_1, \ldots, a_p$ each of length at most $a(n)$ and the verifier accepts x iff $C(a_1, \ldots, a_p) = 1$. We denote this verdict of the verifier by $V^{\pi_1, \cdots, \pi_p}(x; R)$.*

The class of languages accepted by these verifiers is defined as follows:

Definition 2.1.2 *Language L belongs to MIP$_{c,s}[p,r,a]$ if there exists a (p,r,a)-restricted MIP verifier V such that on input x:*

Completeness *If $x \in L$ then there exist $\pi_1, \ldots, \pi_p$ such that V accepts with probability at least c (i.e., $\exists \pi_1, \ldots, \pi_p$ such that $\Pr_R[V^{\pi_1,\cdots,\pi_p}(x;R) = \text{accept}] \geq c$).*

Soundness *If $x \notin L$ then for every $\pi_1, \ldots, \pi_p$, V accepts with probability less than s (i.e., $\forall \pi_1, \ldots, \pi_p$, $\Pr_R[V^{\pi_1,\cdots,\pi_p}(x;R) = \text{accept}] < s$).*

It is easy to see that $\mathrm{MIP}_{c,s}[p,r,a]$ is a subclass of $\mathrm{PCP}_{c,s}[r,pa]$ and thus it is beneficial to show that SAT is contained in MIP with nice parameters. However, much stronger benefits are obtained if the containment has a small number of provers, even if the answer size complexity (a) is not very small. This is because the verifier's actions can usually be simulated by a much more efficient verification procedure, one with much smaller answer size complexity, at the cost of a few more provers. Results of this nature are termed proof composition lemmas; and the efficient simulators of the MIP verification procedure are usually called "inner verification procedures".

2.2 Main Lemmas

The next three lemmas divide the task of proving the Main Theorem into smaller subtasks. The first gives a starting MIP for satisfiability, with 3 provers, but poly-logarithmic answer size. We next give the composition lemma that is used in the intermediate stages. The final lemma gives our terminal composition lemma – the one that reduces answer sizes from some slowly growing function to a constant.

Lemma 2.2.1 *For every $\varepsilon, \mu > 0$, $\mathrm{SAT} \in MIP_{1,\mu}[3, (3+\varepsilon)\log n, \mathrm{poly}\log n]$.*

Lemma 2.2.1 is proven in Chapter 5. This lemma is critical to bounding the proof-size. This lemma follows the proof of a similar one (the "parallelization" step) in [1]; however various aspects are improved. We show how to incorporate advances made by Polishchuk and Spielman [23], and how to take advantage of the low-degree test of Raz and Safra [25]. Most importantly, we show how to save a quadratic blowup in this phase that would be incurred by a direct use of the parallelization step in [1].

The first composition lemma we use is an off-the-shelf product due to Arora and Sudan [3]. Similar lemmas are implicit in the works of Bellare, Goldwasser, Lund and Russell [8] and Raz and Safra [25].

Lemma 2.2.2 ([3]) *There exist absolute constants c_1, c_2, c_3 such that for every $\epsilon > 0$, every p, and every $r, a : \mathbb{Z}^+ \to \mathbb{Z}^+$,*

$$MIP_{1,\epsilon}[p,r,a] \subseteq MIP_{1,\epsilon^{1/(2p+2)}}[p+3, r + c_1 \log a, c_2(\log a)^{c_3}].$$

The next lemma shows how to truncate the recursion. This lemma is proved in Chapter 6 using a "Fourier-analysis" based proof, as in [19]. This is the first time that this style of analysis has been applied to MIPs with more than 2 provers. All previous analyses seem to have focused on composition with canonical 2-prover proof systems at the outer level. Our analysis reveals surprising complications and forces us to use a large number (seven) of extra bits to effect the truncation.

Lemma 2.2.3 *For every $\epsilon > 0$ and $p < \infty$, there exists a $\gamma > 0$ such that for every $r, a : \mathbb{Z}^+ \to \mathbb{Z}^+$,*

$$MIP_{1,\gamma}[p, r, a] \subseteq PCP_{1,\frac{1}{2}+\epsilon}[r + O(2^{pa}), p + 7].$$

2.3 Main Theorem and Proof

Theorem 2.3.1 *For every $\varepsilon, \mu > 0$,*

$$\mathrm{SAT} \in PCP_{1,\frac{1}{2}+\mu}\left[(3+\varepsilon)\log n, 16\right].$$

Proof The proof is straightforward given the three lemmas mentioned in Section 2.2. We first apply Lemma 2.2.1 to get a 3-prover MIP for SAT, then apply Lemma 2.2.2 twice to get a 6- and then a 9-prover MIP for SAT. The answer size in the final stage is $\mathrm{poly}\log\log\log n$. Applying Lemma 2.2.3 at this stage we obtain a 16-query PCP for SAT; and the total randomness in all stages remains $(3+\varepsilon)\log n$. □

It follows from the parameters that the associated proof is of size at most $O(n^{3+\varepsilon})$.

Cook [11] showed that any language in NTIME$(t(n))$ could be reduced to SAT such that instances of size n are mapped to boolean formulae of size at most $O(t(n)\log t(n))$.

Lemma 2.3.2 ([11]) *Let $L \in \mathrm{NTIME}(t(n))$. Then there is a $O(t(n)\log t(n))$ time and $O(\log t(n))$ space algorithm that maps inputs x of length n to boolean formulae ϕ of size $O(t(n)\log t(n))$ such that*

$$x \in L \Longleftrightarrow \phi \in \mathrm{SAT}$$

Combining this Lemma with Theorem 2.3.1, we have that every language in NP has a PCP with at most a slightly super-cubic blowup in proof-size and a query complexity as low as 16 bits.

Chapter 3

Polynomial Constraint Satisfiability

In this chapter, we prepare the necessary ground for building a randomness efficient MIP for SAT. For this purpose, we reduce SAT to another NP-hard problem, that is amenable for MIP constructions. In our choice of a NP-hard problem, we are guided by considerations, similar to those in [23, 28]. We would like our NP-hard problem to satisfy the following properties.

- Problems like SAT, circuit-satisfiability can be "efficiently"[1] reduced to this problem.
- It is easy to construct an MIP for this problem.

At this point, it is worth mentioning that problems like SAT, circuit-satisfiability do not directly satisfy our second requirement. The known algebraic descriptions of these problems usually involve a cubic blowup in the proof-size. The main handicap in these problems that leads to such a blowup in the proof-size is that to check whether a particular constraint is satisfied, we have to look in the proof for the values of the variables that participate in the constraint and these values could appear anywhere in the proof. We design a problem (see Definition 3.2.1), in such a manner that to check whether a particular constraint is satisfied, we would instantly know where the values of the variables that participate in the constraint can be found.

Henceforth, we shall use the term "length-preserving reductions", to refer to reductions in which the length of the target instance of the reduction is nearly-linear ($O(n^{1+\epsilon})$ for arbitrarily small ϵ) in the length of the source instance. We shall use the term "length-efficient reductions", to refer to reductions in which the length of the target instance of the reduction is at most an extra logarithmic factor off the length of the source instance (i.e., $O(n \log n)$).

[1]By efficiently, we mean that the reductions are *length preserving*, a notion we will formalize shortly.

To prove membership in SAT, we first transform SAT into an algebraic problem. This transformation comes in two phases. First we transform it to an algebraic problem (that we call AP for lack of a better name) in which the constraints can be enumerated compactly. Then we transform it to a promise problem on polynomials, called Polynomial Constraint Satisfaction (PCS), with a large associated gap. We then show how to provide an MIP verifier for the PCS problem in Chapter 5.

Though most of these results are implicit in the literature, we find that abstracting them cleanly significantly improves the exposition of PCPs. The first problem, AP, could be proved to be NP-hard almost immediately, if one did not require length-preserving reductions. We show how the results of Polishchuk and Spielman [23] imply a length preserving reduction from SAT to this problem. We then reduce this problem to PCS. This step mimics the sum-check protocol of Lund, Fortnow, Karloff and Nisan [22]. The technical importance of this intermediate step is the fact that it does *not* refer to "low-degree" tests in its analysis. Low-degree tests are primitives used to test if the function described by a given oracle is close to some (unknown) multivariate polynomial of low-degree. Low-degree tests have played a central role in the constructions of PCPs. Here we separate (to a large extent) their role from other algebraic manipulations used to obtain PCPs/MIPs for SAT .

3.1 A Compactly Described Algebraic NP-hard Problem

Definition 3.1.1 *For functions $m, h : \mathbb{Z}^+ \to \mathbb{Z}^+$, the problem $\mathrm{AP}_{m,h}$ has as its instances $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$ where: H is a field of size $h(n)$, $\psi : H^7 \to H$ is a constant degree polynomial, T is an arbitrary function from H^m to H and the ρ_i's are linear maps from H^m to H^m, for $m = m(n)$. (T is specified by a table of values, and ρ_i's by $m \times m$ matrices.) $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6) \in \mathrm{AP}_{m,h}$ if there exists an assignment $A : H^m \to H$ such that for every $x \in H^m$, $\psi(T(x), A(\rho_1(x)), \ldots, A(\rho_6(x))) = 0$. (In case such an assignment A exists, we then say that A satisfies $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$).*

The above problem is just a simple variant of standard constraint satisfaction problems, the only difference being that its variables and constraints are now indexed by elements of H^m. The only algebra in the above problem is in the fact that the functions ρ_i, which dictate which variables participate in which constraint, are linear functions. The following statement, abstracted from [23], gives the desired hardness of AP.

Lemma 3.1.2 *There exists a constant c such that for any pair of functions $m, h : \mathbb{Z}^+ \to \mathbb{Z}^+$ satisfying $h(n)^{m(n)-c} \geq n$ and $h(n)^{m(n)} = O\left(n^{1+o(1)}\right)$, SAT reduces to $\mathrm{AP}_{m,h}$ under length preserving reductions.*

Lemma 3.1.2 is a reformulation of the result proved in [23, 28] in a manner that is convenient for us to work with. The proof, we present, is along the lines of [23, 28]. In the following two subsections, we (re)present the machinery required to prove the lemma and finally provide a proof of the lemma in Section 3.1.3.

3.1.1 De Bruijn Graph Coloring Problem

Definition 3.1.3 *The de Bruijn graph B_n is a directed graph on 2^n vertices in which each vertex is represented by a n-bit binary string. The vertex represented by $(x_1, \ldots, x_n)$ has edges pointing to the vertices represented by $(x_2, \ldots, x_n, x_1)$ and $(x_2, \ldots, x_n, x_1 \oplus 1)$, where $a \oplus b$ denotes the sum of a and b modulo 2.*

We then define a *wrapped de Bruijn graph* to be the product of a de Bruijn graph and a cycle.

Definition 3.1.4 *The wrapped de Bruijn graph $\mathcal{B}_n$ is a directed graph on $5n \cdot 2^n$ vertices in which each vertex is represented by a pair consisting of an n-bit binary string and a number modulo $5n$. The vertex represented by $((x_1, \ldots, x_n), a)$ has edges pointing to the vertices $((x_2, \ldots, x_n, x_1), a + 1)$ and $((x_2, \ldots, x_n, x_1 \oplus 1), a + 1)$, where the addition $a + 1$ is performed modulo $5n$.*

Similarly, one can define the *extended de Bruijn graph* (on $(5n + 1) \cdot 2^n$ vertices) to be the product of the de Bruijn graph (on 2^n vertices) and a line graph (on $5n + 1$ vertices). For ease of notation, let us define for any vertex v, $\varrho_1(v)$ and $\varrho_2(v)$ to be the two neighbors of v in the wrapped de Bruijn graph. [23, 28] show how to reduce SAT to the following coloring problem on the wrapped de Bruijn graph using standard packet routing techniques (see [21]).

Definition 3.1.5 *The problem* DE-BRUIJN-GRAPH-COLOR *has as its instances $(\mathcal{B}_n, T)$ where $\mathcal{B}_n$ is a wrapped de Bruijn graph on $5n \cdot 2^n$ vertices and $T : V(\mathcal{B}_n) \to C_1$ is a coloring of the vertices of $\mathcal{B}_n$ (T is specified by a table of values). $(\mathcal{B}_n, T) \in$* DE-BRUIJN-GRAPH-COLOR *if there exists another coloring $A : V(\mathcal{B}_n) \to C_2$ such that for all vertices $v \in V(\mathcal{B}_n)$,*

$$\varphi(T(v), A(v), A(\varrho_1(v)), A(\varrho_2(v))) = 0$$

where C_1, C_2 are two sets of colors independent of n and $\varphi : C_1 \times C_2^3 \to \mathbb{Z}^+$ is a function independent of n.

[23, 28] prove the following statement regarding the hardness of the above problem.

Proposition 3.1.6 ([23, 28]) SAT *reduces to* DE-BRUIJN-GRAPH-COLOR *under length-efficient reductions.*

3.1.2 Algebraic Description of De Bruijn Graphs

In this section, we shall give a very simple algebraic description of the de Bruijn graphs.

Definition 3.1.7 *A Galois graph G_n is a directed graph on 2^n vertices in which each vertex is node is identified with an element of $GF(2^n)$. Let α be a generator*[2] *of $GF(2^n)$. The vertex represented by $\gamma \in GF(2^n)$ has edges pointing to the vertices represented by $\alpha\gamma$ and $\alpha\gamma + 1$.*

Claim 3.1.8 *The Galois graph G_n is isomorphic to the de Bruijn graph B_n.*

Proof Recall the standard definition of $GF(2^n)$. Let $p(\alpha) = \alpha^n + c_1\alpha^{n-1} + \ldots + c_{n-1}\alpha + c_n$ be any irreducible monic polynomial over $GF(2)$ of degree n. Then $GF(2^n)$ can be identified with $GF(2)[\alpha]/(p(\alpha))$. Addition and multiplication in $GF(2^n)$ are simple, they are performed exactly similar to polynomial addition and multiplication and the result is then reduced modulo $p(\alpha)$.

We shall show that G_n and B_n are isomorphic by exhibiting an isomorphism $\phi : V(B_n) \to V(G_n)$, between the vertices of the two graphs, as follows:

$$\phi(b_1, \ldots, b_n) = \alpha^{n-1}b_1 + \alpha^{n-2}(b_2 + cb_1) + \ldots + \left(b_n + \sum_{i=1}^{n-1} c_i b_{n-1})\right)$$

To verify that this is an isomorphism, we need to check that $(u, v) \in E(B_n) \iff (\phi(u), \phi(v)) \in E(G_n)$. Note that in the graph B_n, the edges from the vertex $(b_1, \ldots, b_n)$ are pointed towards the vertices $(b_2, \ldots, b_n, b_1)$ and $(b_2, \ldots, b_n, b_1 \oplus 1)$; while in G_n, the edges from

$$\phi(b_1, \ldots, b_n) = \alpha^{n-1}b_1 + \alpha^{n-1}(b_2 + cb_1) + \ldots + \left(b_n + \sum_{i=1}^{n-1} c_i b_{n-1})\right)$$

are towards the vertices

$$\begin{aligned}
& \alpha\left(\alpha^{n-1}b_1 + \alpha^{n-2}(b_2 + cb_1) + \ldots + \left(b_n + \sum_{i=1}^{n-1} c_i b_{n-1})\right)\right) \\
= \quad & b_1(c_1\alpha^{n-1} + c_{n-1}\alpha + c_n) + \alpha\left(\alpha^{n-2}(b_2 + cb_1) + \ldots + \left(b_n + \sum_{i=1}^{n-1} c_i b_{n-1})\right)\right) \\
= \quad & \alpha^{n-1}b_2 + \alpha^{n-2}(b_3 + c_1b_2) + \ldots + \alpha\left(b_n + \sum_{i=1}^{n-1} c_i b_{n-i}\right) + c_n b_1
\end{aligned}$$

and

$$\alpha^{n-1}b_2 + \alpha^{n-2}(b_3 + c_1b_2) + \ldots + \alpha\left(b_n + \sum_{i=1}^{n-1} c_i b_{n-i}\right) + c_n b_1 + 1$$

which we can easily check to be $\phi(b_2, \ldots, b_n, b_1)$ and $\phi(b_2, \ldots, b_n, b_1 \oplus 1)$ (not necessarily in that order). $\square$

Claim 3.1.9 *Let m divide n and α be a generator of $GF(2^{n/m})$. Then the graph on*

$$\underbrace{GF(2^{n/m}) \times GF(2^{n/m}) \times \ldots \times GF(2^{n/m})}_{m \text{ times}}$$

[2] A generator of $GF(2^n)$ is an element $\alpha \in GF(2^n)$ such that $\alpha^{2^n-1} = 1$ and $\alpha^k \neq 1$ for any $1 \leq k < 2^n - 1$. Every element in $GF(2^n)$ can be represented by a unique polynomial in α of degree at most $n-1$ with coefficients from $\{0, 1\}$.

in which the vertex represented by $(\sigma_1, \ldots, \sigma_m)$ *has edges pointing to the vertices represented by*

$$(\sigma_2, \ldots, \sigma_m, \alpha\sigma_1) \text{ and } (\sigma_2, \ldots, \sigma_m, \alpha\sigma_1 + 1)$$

is isomorphic to the de Bruijn graph B_n*.*

Proof By Claim 3.1.8, the given graph is isomorphic to the graph on binary strings of length n in which the vertex

$$(b_1, \ldots, b_{\frac{n}{m}}, b_{\frac{n}{m}+1}, \ldots, b_{2\frac{n}{m}}, \ldots, b_{(m-1)\frac{n}{m}+1}, \ldots, b_n)$$

has edges pointing to the vertices given by

$$(b_{\frac{n}{m}+1}, \ldots, b_{2\frac{n}{m}}, \ldots, b_{(m-1)\frac{n}{m}+1}, \ldots, b_n, b_2, \ldots, b_{\frac{n}{m}}, b_1)$$

and

$$(b_{\frac{n}{m}+1}, \ldots, b_{2\frac{n}{m}}, \ldots, b_{(m-1)\frac{n}{m}+1}, \ldots, b_n, b_2, \ldots, b_{\frac{n}{m}}, b_1 \oplus 1)$$

Shuffling the order of b_i's, we observe that this graph is isomorphic to the graph in which the vertex represented by

$$(b_1, b_{\frac{n}{m}+1}, \ldots, b_{(m-1)\frac{n}{m}+1}, b_2, b_{\frac{n}{m}+2}, \ldots, b_{(m-1)\frac{n}{m}+2}, \ldots, b_m, b_{2m}, \ldots, b_n)$$

has edges pointed towards the vertices

$$(b_{\frac{n}{m}+1}, \ldots, b_{(m-1)\frac{n}{m}+1}, b_2, b_{\frac{n}{m}+2}, \ldots, b_{(m-1)\frac{n}{m}+2}, \ldots, b_m, b_{2m}, \ldots, b_n, b_1)$$

and

$$(b_{\frac{n}{m}+1}, \ldots, b_{(m-1)\frac{n}{m}+1}, b_2, b_{\frac{n}{m}+2}, \ldots, b_{(m-1)\frac{n}{m}+2}, \ldots, b_m, b_{2m}, \ldots, b_n, b_1 \oplus 1)$$

which is identical to the de Bruijn graph. □

Using the above result, we can now give a simple algebraic description of the extended de Bruijn graphs.

Proposition 3.1.10 *Let* m *divide* n *and* α *be a generator of* $H = GF(2^{n/m})$*. Let* $\mathcal{C} = \{1, \alpha, \ldots, \alpha^{5n}\}$ *and* $\mathcal{C}' = \{1, \alpha, \ldots, \alpha^{5n-1}\}$*. Then the extended de Bruijn graph on* $(5n+1) \cdot 2^n$ *vertices is isomorphic to the graph on* $H^m \times \mathcal{C}$ *in which each vertex in* $(x_1, \ldots, x_m, y) \in H^m \times \mathcal{C}'$ *has edges pointed towards the vertices*

$$(x_2, \ldots, x_m, \alpha x_1, \alpha y)$$

and

$$(x_2, \ldots, x_m, \alpha x_1 + 1, \alpha y)$$

For ease of notation, if $v \in H^m \times \mathcal{C}$, then let $\varrho_1(v)$ and $\varrho_2(v)$ denote the two neighbors of v. Or even more generally, for any $v \in H^{m+1}$, define

$$\varrho_1(x_1, \ldots, x_m, y) \mapsto (x_2, \ldots, x_m, \alpha x_1, \alpha y) \quad (3.1)$$

$$\varrho_2(x_1, \ldots, x_m, y) \mapsto (x_2, \ldots, x_m, \alpha x_1 + 1, \alpha y) \quad (3.2)$$

3.1.3 *Proof of Lemma 3.1.2*

Instead of showing that SAT is reducible to $\mathrm{AP}_{m,h}$, we shall show that SAT is reducible under length preserving reductions to another problem $\mathrm{AP'}_{m,h}$. It would then follow from the definition of AP and AP$'$ that SAT is reducible to $AP_{m,h}$ under length preserving reductions.

Definition 3.1.11 *For functions $m, h : \mathbb{Z}^+ \to \mathbb{Z}^+$, the problem $\mathrm{AP'}_{m,h}$ has as its instances $(1^n, H, T, \psi, \rho_1, \ldots, \rho_5, \rho)$ where: H is a field of size $h(n)$, $\psi : H^7 \to H$ is a constant degree polynomial, T is an arbitrary function from H^{m-1} to H, the ρ_i's are linear maps from H^m to H^{m-1} and $\rho : H^m \to H$ is a linear map for $m = m(n)$. (T is specified by a table of values, ρ_i's by $m \times (m-1)$ matrices and ρ by a $m \times 1$ matrix.) $(1^n, H, T, \psi, \rho_1, \ldots, \rho) \in \mathrm{AP'}_{m,h}$ if there exists an assignment $A : H^{m-1} \to H$ such that for every $x \in H^m$, $\psi(T(\rho_1(x)), A(\rho_1(x)), \ldots, A(\rho_5(x)), \rho(x)) = 0$.*

Proposition 3.1.12 *For any pair of functions $m, h : \mathbb{Z}^+ \to \mathbb{Z}^+$ satisfying $h(n)^{m(n)-2} \geq n$ and $h(n)^{m(n)} = O\left(n^{1+o(1)}\right)$, SAT reduces to $\mathrm{AP'}_{m,h}$ under length preserving reductions.*

Proof Let ϕ be any instance of SAT of size n. By Proposition 3.1.6, we have that ϕ can be reduced to an instance $(\mathcal{B}_{n'}, T)$ of DE-BRUIJN-GRAPH-COLOR. As the reduction is perfect length-efficient, we have that $5n' \cdot 2^{n'} = O(n \log n)$ or $N \approx n$ where $N = 2^{n'}$. Let m and h be any two functions satisfying the requisites of Proposition 3.1.12. Let $m'(n) = m(n) - 2$. Let α be a generator of the field $GF(2^{n/m'})$. Now as $h(n)^{m(n)-2} \geq n$, there exists a field H of size $h(n)$ such that the field $GF(2^{n/m'})$ can be embedded in H. Now, as seen from Section 3.1.2, we can view the graph B_n as a graph on $H^{m'}$ and the graph $\mathcal{B}_n$ as a graph on $H^{m'} \times \mathcal{C}$ where $\mathcal{C} = \{1, \alpha, \ldots, \alpha^{5n}\}$. As $\mathcal{C} \subseteq GF(2^{n/m'}) \subseteq H$, we can further view $\mathcal{B}_n$ as a graph on $H^{m'+1}$, where the neighborhood functions ϱ_1, ϱ_2 are as defined in (3.1) and (3.2). We can also view the set of colors C_1 and C_2 as embedded in the field H. With such an embedding, we can consider the map $T : V(\mathcal{B}_{n'}) \to C_1$ as a map $T : H^{m'+1} \to H$.

Consider the following choice of linear transformations $\rho_i : H^m \to H^{m'+1}$ (recall $m' = m - 2$) For any $(\bar{x}, y, z) \in H^m$ where $\bar{x} \in H^{m'}, y, z \in H$

- $\rho_1 : (\bar{x}, y, z) \mapsto (\bar{x}, y)$.
- $\rho_2 : (\bar{x}, y, z) \mapsto \varrho_1(\bar{x}, y)$.
- $\rho_3 : (\bar{x}, y, z) \mapsto \varrho_2(\bar{x}, y)$.
- $\rho_4 : (\bar{x}, y, z) \mapsto (\bar{x}, 1)$.
- $\rho_5 : (\bar{x}, y, z) \mapsto (\bar{x}, \alpha^{5n})$.

Also define $\rho : H^m \to H$ such that $\rho_6 : (\bar{x}, y, z) \mapsto z$. Note each of the ρ_i's are linear transformations. Now consider the polynomials defined as follows:

- $\varphi_1 : H^4 \to H$ satisfying $\varphi_1\Big|_{C_1 \times C_2^3} = \varphi$. i.e., the restriction of φ_1 on the subset $C_1 \times C_2^3$ of the domain is the same as the function φ in the definition of DE-BRUIJN-GRAPH-COLOR.
- $\varphi_2 : H^2 \to H$ such that $\varphi_2(a, b) = 0$ iff $a = b$. (i.e., φ_2 checks if its two inputs are equal.)
- $\varphi_3 : H \to H$ satisfying $\varphi_3\Big|_{C_2} \equiv 0$. (i.e., φ_3 evaluates to true if its input belongs to the set C_2)
- $\varphi_4 : H \to H$ satisfying $\varphi_4\Big|_{C_1} \equiv 0$. (i.e., φ_4 evaluates to true if its input belongs to the set C_1)

It can easily be seen that the φ_i's can be defined such that they are all of constant degree where the degree depends only on the cardinality of the sets C_1 and C_2.

Now consider the polynomial $\psi : H^7 \to H$ defined as follows

$$\psi(a, b, c, d, e, f, t) = \begin{cases} \varphi_1(a, b, c, d) & \text{if } t = 1, \\ \varphi_2(e, f) & \text{if } t = 2, \\ \varphi_3(b) & \text{if } t = 3, \\ \varphi_4(a) & \text{if } t = 4, \\ \text{arbitrary} & \text{otherwise.} \end{cases}$$

It can easily be checked that ψ is also a constant degree polynomial. By construction of ψ, we have that $\psi(T(\rho_1(z)), A(\rho_1(z)), A(\rho_2(z)), A(\rho_3(z)), A(\rho_4(z)), A(\rho_5(z)), \rho(z)) = 0, \forall z \in H^m$ iff the corresponding instance $(\mathcal{B}_{n'}, T) \in$ DE-BRUIJN-GRAPH-COLOR, which happens iff $\phi \in \text{SAT}$. Note

(1) φ_1 checks if the condition φ is satisfied by vertices of the graph.

(2) φ_2 checks if the first and last column of the extended graph is the same (and hence the graph can be viewed as a wrapped graph).

(3) Finally, φ_3 and φ_4 checks iff the colors assigned by the function A and T are indeed valid colors. (i.e., $T(v) \in C_1$ and $A(v) \in C_2$.)

We have thus shown that $(1^n, H, T, \psi, \rho_1, \ldots, \rho_5, \rho) \in \text{AP}'_{m,h} \iff \phi \in \text{SAT}$. Moreover all the reductions mentioned are length preserving (since $h^m = O\left(n^{1+o(n)}\right)$). Thus, proved. □

3.2 *Polynomial Constraint Satisfaction*

We next present an instance of an algebraic constraint satisfaction problem. This differs from the previous one in that its constraints are "wider", the relationship between constraints and variables that appear in it is arbitrary (and not linear), and the hardness is not established for arbitrary assignment functions, but only for low-degree functions. All the above changes only make the problem harder, so we ought to gain something – and we gain in the gap of the hardness. The problem is shown to be hard even if the goal is only to separate satisfiable instances from instances in which only ϵ fraction of the constraints are satisfiable. We define this gap version of the problem first.

Definition 3.2.1 *For $\epsilon : \mathbb{Z}^+ \to \mathbb{R}^+$, and $m, b, q : \mathbb{Z}^+ \to \mathbb{Z}^+$ the promise problem $\mathrm{GapPCS}_{\epsilon,m,b,q}$ has as instances $(1^n, d, k, s, \mathbb{F}; C_1, \ldots, C_t)$, where $d, k, s \leq b(n)$ are integers and $\mathbb{F}$ is a field of size $q(n)$ and $C_j = (A_j; x_1^{(j)}, \ldots, x_k^{(j)})$ is an algebraic constraint, given by an algebraic circuit A_j of size s on k inputs and $x_1^{(j)}, \ldots, x_k^{(j)} \in \mathbb{F}^m$, for $m = m(n)$. $(1^n, d, k, s, \mathbb{F}; C_1, \ldots, C_t)$ is a YES instance if there exists a polynomial $p : \mathbb{F}^m \to \mathbb{F}$ of degree at most d such that for every $j \in \{1, \ldots, t\}$, the constraint C_j is satisfied by p, i.e., $A_j(p(x_1^{(j)}), \ldots, p(x_k^{(j)})) = 0$. $(1^n, d, k, s, \mathbb{F}; C_1, \ldots, C_t)$ is a NO instance if for every polynomial $p : \mathbb{F}^m \to \mathbb{F}$ of degree at most d it is the case that at most $\epsilon(n) \cdot t$ of the constraints C_j are satisfied.*

Lemma 3.2.2 *There exist constants c_1, c_2 such that for every choice of functions ϵ, m, b, q satisfying $(b(n)/m(n))^{m(n)-c_1} \geq n$, $q(n) \geq c_2 b(n)/\epsilon(n)$ and $q(n) = O\left(n^{1+o(1)}\right)$, SAT reduces to $\mathrm{GapPCS}_{\epsilon,m,b,q}$ under length preserving reductions.*

(The problem $\mathrm{AP}_{m,h}$ is used as an intermediate problem in the reduction. However we don't mention this in the lemma, since the choice of parameters m, h may confuse the statement further.)

We shall prove the hardness of $\mathrm{GapPCS}_{\epsilon,m,b,q}$ using another related problem *Polynomial Evolution* (PE) as an intermediary problem between AP and GapPCS. In Section 3.2.1, we describe the problem Polynomial Evolution and analyze its hardness. In Section 3.2.2, we prove Lemma 3.2.2.

3.2.1 Polynomial Evolution

Definition 3.2.3 *A polynomial construction rule R over a field $\mathbb{F}$ on m variables is a circuit which takes an oracle for a polynomial $p : \mathbb{F}^m \to \mathbb{F}$ and returns a new polynomial $q : \mathbb{F}^m \to \mathbb{F}$, defined by $q \triangleq R^p(x)$.*

Polynomial Evolution involves checking whether there exists a polynomial $p : \mathbb{F}^m \to \mathbb{F}$ such that when a given sequence of construction rules are composed on this polynomial, the resulting polynomial is identically zero. More formally,

Definition 3.2.4 *For functions $b, m, q : \mathbb{Z}^+ \to \mathbb{Z}^+$, the problem $\mathrm{PE}_{m,b,q}$ has as instances $(1^n, d, \mathbb{F}; R_1, \ldots, R_l)$ where $d \leq b(n)$ are integers, $\mathbb{F}$ is a finite field of size $q(n)$ and the R_i's are polynomial construction rules over $\mathbb{F}$ on m variables. $(1^n, d, \mathbb{F}; R_1, \ldots, R_l) \in \mathrm{PE}_{m,b,q}$ if there exists a polynomial $p_0 : \mathbb{F}^m \to \mathbb{F}$ of degree at most d such that the sequence of polynomials p_i defined by $p_i \triangleq R^{p_{i-1}}$ for $i = 1 \ldots l$ satisfies $p_l \equiv 0$ (i.e., p_l is identically zero.)*

If q^m is polynomial in the description of the instance, then clearly $\mathrm{PE}_{m,b,q} \in \mathrm{NP}$. We shall prove the following statement regarding the hardness of $\mathrm{PE}_{m,b,q}$.

Lemma 3.2.5 *There exists a constant $c \in \mathbb{Z}^+$ such that for functions $m, h, q : \mathbb{Z}^+ \to \mathbb{Z}^+$ satisfying $q \geq cmh$ and $q^m = O(n^{1+o(1)})$, $\mathrm{AP}_{m,h}$ reduces to $\mathrm{PE}_{m,mh,q}$ under length-preserving reductions.*

Let $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$ be an instance of $\mathrm{AP}_{m,h}$. Let $\mathbb{F}$ be a field of size $q(n)$ where q satisfies the requirements of Lemma 3.2.5 such that $H \subseteq \mathbb{F}$. Let c be the degree of the polynomial $\psi : H^7 \to H$. (Recall that by definition of $\mathrm{AP}_{m,h}$, c is a constant.)

Any assignment $S : H^m \to H$ can be interpolated to obtain a polynomial $\hat{S} : \mathbb{F}^m \to \mathbb{F}$ of degree at most $|H|$ in each variable (and hence a total degree of at most $m|H|$) such that $\hat{S}\Big|_{H^m} = S$. (i.e., the restriction of $\hat{S}$ to H^m coincides with the function S.) Conversely, any polynomial $\hat{S} : \mathbb{F}^m \to \mathbb{F}$ can be interpreted as an assignment from H^m to $\mathbb{F}$ by considering the function restricted to the sub-domain H^m.

Based on the instance $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$, we will construct a sequence of $(m+1)$ polynomial construction rules which transform a polynomial p_0 to the zero polynomial iff the assignment given by $A = p_0\Big|_{H^m}$ satisfies the instance $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$. The first rule takes as input a polynomial $p_o : \mathbb{F}^m \to \mathbb{F}$ of degree mh and outputs a polynomial $p_1 : \mathbb{F}^m \to \mathbb{F}$ of degree cmh which is 0 on H^m iff the corresponding assignment $p_0\Big|_{H^m}$ satisfies the instance $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$. The remaining m rules follow the sum-check protocol of Lund, Fortnow, Karloff and Nisan [22] and "amplify" the zero-set of the polynomial p_1 so that the resulting polynomials are zero on larger and larger sets. The final polynomial $p_{m+1} : \mathbb{F}^m \to \mathbb{F}$ will be identically zero iff the original polynomial p_1 was zero on H^m and hence, iff $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6) \in \mathrm{AP}_{m,h}$.

The first polynomial construction rule R_1 encodes the polynomial $\psi : H^7 \to H$ of constant degree c, the function $T : H^m \to H$ and the linear transformations $\rho_i : H^m \to H$. Let $\hat{T} : \mathbb{F}^m \to \mathbb{F}$ be interpolation of T such that the restriction coincides with the function T. Also let $\hat{\psi} : \mathbb{F}^7 \to \mathbb{F}$ be the extension of the polynomial ψ to the domain $\mathbb{F}^m$. (i.e., If $\psi : H^m \to H$ is given by $\psi(x_1, \ldots, x_m) = \sum a_{i_1,\ldots,i_m} x_1^{i_1} \ldots x_m^{i_m}$, then $\hat{\psi} : \mathbb{F}^m \to \mathbb{F}$ is the same polynomial $\psi(x_1, \ldots, x_m) = \sum a_{i_1,\ldots,i_m} x_1^{i_1} \ldots x_m^{i_m}$.) Note $\hat{\psi}$ is also of degree c. Also let $\hat{\rho_i} : \mathbb{F}^m \to \mathbb{F}^m$ represent the extension of the linear transformation $\rho_i : H^m \to H^m$ to the domain $\mathbb{F}^m$ (i.e., if ρ_i is the linear map given by $\bar{x} \mapsto A\bar{x}$ where $\bar{x} \in H^m$ and A is a $m \times m$ matrix with elements from H, then $\hat{\rho_i}$ is the linear map given by $x \mapsto A\bar{x}$ where $\bar{x} \in \mathbb{F}^m$) The rule R_1 is defined as follows:

$$p_1(x_1, \ldots, x_m) \quad \triangleq \quad \hat{\psi}(\hat{T}(x_1, \ldots, x_m), p_0(\hat{\rho_1}(x_1, \ldots, x_m)), \ldots, p_0(\hat{\rho_6}(x_1, \ldots, x_m)))$$

When $p_0 = \hat{A}$ for some assignment $A : H^m \to H$, then for $(x_1, \ldots, x_m) \in H^m$,

$$p_1(x_1, \ldots, x_m) \quad = \quad \psi(T(x_1, \ldots, x_m), A(\rho_1(x_1, \ldots, x_m)), \ldots, A(\rho_6(x_1, \ldots, x_m)))$$

Thus, $p_1\Big|_{H^m} \equiv 0$ iff the polynomial p_0 represents an assignment A that satisfies the instance $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$. Note that if p_0 is a polynomial of degree mh, then p_1 is a polynomial of degree at most cmh where c is the degree of the polynomial ψ.

Now to the remaining rules. It is to be noted that only rule R_1 actually depends on the instance, the other rules are generic rules which follow the sum-check protocol in [22]. As mentioned earlier, these rules make the zero-set of the polynomials larger and larger.

For starters, let us first work on a univariate polynomial, $p : \mathbb{F} \to \mathbb{F}$. Let $H = \{h_1, \dots, h_{|H|}\}$ be an enumeration of the elements in H. Consider the construction rule that works as follows:

$$q(r) \triangleq \sum_{j=1}^{|H|} p(h_j) r^j$$

Clearly, if $p(h) = 0$ for all $h \in H$, then $q \equiv 0$ on $\mathbb{F}$. Conversely, if $\exists h \in H, p(h) \neq 0$, then q is a non-zero polynomial and hence is not identically zero.

Now, for multivariate polynomials, we shall mimic the above construction. Consider the sequence of polynomials construction rules defined as follows. For $i = 1, \dots, m$, rule R_{i+1} works as follows:

$$R_{i+1} : p_{i+1}\left(\underbrace{\longleftarrow \bar{r} \longrightarrow}_{i-1 \text{ variables}}, r_i, \underbrace{\longleftarrow \bar{x} \longrightarrow}_{m-i \text{ variables}}\right) \triangleq \sum_{j=1}^{|H|} p_i\left(\underbrace{\longleftarrow \bar{r} \longrightarrow}, h_j, \underbrace{\longleftarrow \bar{x} \longrightarrow}\right) r_i^j$$

By the same reasoning as in the univariate case, we have that

$$p_{i+1}\Big|_{\mathbb{F}^i \times H^{m-i}} \equiv 0 \Longleftrightarrow p_i\Big|_{\mathbb{F}^{i-1} \times H^{m-i+1}} \equiv 0$$

Thus, $p_{m+1} \equiv 0$ iff $p_1\Big|_{H^m}$. But $p_1\Big|_{H^m} \equiv 0$ iff $p_0\Big|_{H^m}$ satisfies $(1^n, H, T, \psi, \rho_1, \dots, \rho_6)$. Thus, the rules we have constructed satisfy

$$(1^n, mh, \mathbb{F}; R_1, \dots, R_{m+1}) \in \mathrm{PE}_{m,mh,q} \Longleftrightarrow (1^n, H, T, \psi, \rho_1, \dots, \rho_6) \in \mathrm{AP}_{m,h}$$

It can easily be checked that the reduction is length preserving. Thus, Lemma 3.2.5 is proved.

We can in fact prove a stronger statement regarding the hardness of the PE instance, we have created.

Proposition 3.2.6 *Suppose, we have an instance $(1^n, d, \mathbb{F}; R_1, \dots, R_{m+1})$ of $\mathrm{PE}_{m,mh,q}$ constructed from an instance $(1^n, H, T, \psi, \rho_1, \dots, \rho_6)$ of $\mathrm{AP}_{m,h}$ as mentioned above.*

- *[Completeness] If $(1^n, H, T, \psi, \rho_1, \dots, \rho_6) \in \mathrm{AP}_{m,h}$, then there exists a polynomial $p_0 : \mathbb{F}^m \to \mathbb{F}$ of degree at most mh such that the sequence of polynomials constructed by applying the rules $R_1, \dots, R_{m+1}$ (i.e., $p_i = R^{p_{i-1}}$ for $i = 1 \dots m+1$) satisfy $p_{m+1} \equiv 0$. Moreover, each of the polynomials $p_1, \dots, p_{m+1}$ are of degree at most cmh.*

- *[Soundness] If there exist polynomials $p_0 : \mathbb{F}^m \to \mathbb{F}$ of degree at most mh and polynomials $p_1, \dots, p_{m+1}$ of degree at most cmh each, such that*

$$\Pr_{\bar{x} \in \mathbb{F}^m}[p_i(\bar{x}) = R^{p_{i-1}}] > \frac{(c+1)mh}{q}, i = 1, \dots, m+1$$
$$\Pr_{\bar{x} \in \mathbb{F}^m}[p_{m+1}(\bar{x}) = 0] > \frac{(c+1)mh}{q}$$

then, $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6) \in \mathrm{AP}_{m,h}$.

For the proof of this proposition, we shall need Schwartz's Lemma.

Lemma 3.2.7 (Schwartz Lemma [27]) *For any finite field* $\mathbb{F}$*, if* $p, q : \mathbb{F}^m \to \mathbb{F}$ *are two distinct polynomials of degree at most* d *each, then*

$$\Pr_{\bar{x} \in \mathbb{F}^m}[p(\bar{x}) = q(\bar{x})] < \frac{d}{|\mathbb{F}|}$$

Proof of Proposition 3.2.6:

The proof for the Completeness part of the proposition directly follows from the manner in which the rules are constructed.

For the soundness part, we note that the rule R_1 increases the degree of the polynomial by at most a factor of c and each of the other rules R_i has the effect of changing the degree with respect to the $(i-1)^{th}$ variable to at most h and not increasing the degree with respect to any of the other variables. This implies that each of the polynomials $R_i^{p_{i-1}}$ have degree at most $(c+1)mh$. By Schwartz's Lemma, it now follows that $p_i \equiv R_i^{p_{i-1}}$ for $i = 1, \ldots, m+1$ and $p_{m+1} \equiv 0$. But this implies that $p_0\Big|_{H^m}$ satisfies $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$. Thus, proved. □

3.2.2 *Hardness of* Gap PCS

We first reduce AP to GapPCS

Lemma 3.2.8 *There exists a constant* c *such that for all functions* $q, m, h, b, \epsilon : \mathbb{Z}^+ \to \mathbb{Z}^+$ *satisfying* $q(n) \geq b(n)/\epsilon(n)$ *and* $b(n) \geq 2cm(n)h(n)$ *,* $\mathrm{AP}_{m,h}$ *reduces to* $\mathrm{GapPCS}_{\epsilon,m+1,b,q}$ *under length preserving reductions.*

Proof Let $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$ be any instance of $\mathrm{AP}_{m,h}$. Using the reduction in the proof of Lemma 3.2.5, obtain the instance $(1^n, d, \mathbb{F}; R_1, \ldots, R_{m+1})$. We shall build an instance $(1^n, d, k, s, \mathbb{F}; C_1, \ldots, C_t)$ of $\mathrm{GapPCS}_{\epsilon,m+1,b,q}$ as specified below.

Let c be the same constant that appears in Lemma 3.2.5. Let p_0 be the polynomial of degree at most mh that occurs in the proof of the statement "$(1^n, d, \mathbb{F}; R_1, \ldots, R_{m+1}) \in \mathrm{PE}_{m,b,q}$". Also let $p_1, \ldots, p_{m+1}$ be the polynomials defined by the rules $R_1, \ldots, R_{m+1}$ (i.e, $p_i = R_i^{p_{i-1}}$). Note p_i's are of degree at most cmh. We first bundle together the polynomials $p_0, \ldots, p_{m+1}$ into a single polynomial $p : \mathbb{F}^{m+1} \to \mathbb{F}$. Let $\{f_0, \ldots, f_{q-1}\}$ be an enumeration of the elements in $\mathbb{F}$. Let $F_l = \{f_0, \ldots, f_{m+1}\}$. For each $i = 0, \ldots, m+1$, let $\delta_i : \mathbb{F} \to \mathbb{F}$ be the unique polynomial of degree at most $m+1$ satisfying

$$\delta_i(x) = \begin{cases} 1 & \text{if } x = f_i \\ 0 & if \text{ if } x \in F_{m+1} - f_i \end{cases}$$

Polynomial $p : \mathbb{F}^{m+1} \to \mathbb{F}$ is defined as follows: For $(v, \bar{x}) \in \mathbb{F}^{m+1}$ where $v \in \mathbb{F}$ and $\bar{x} \in \mathbb{F}^m$,

$$p(v, \bar{x}) = \sum_{i=0}^{m+1} \delta_i(v) p_i(\bar{x})$$

Since each of the polynomials $p_0, \ldots, p_{m+1}$ is of degree at most cmh, the polynomial p is of degree at most $cmh + m \leq 2cmh \leq b$.

For each $x \in \mathbb{F}^m$, construct constraint C_x as follows:

$$C_x = \Big(p_{m+1}(x) = 0\Big) \wedge \bigwedge_{i=1}^{m+1} \left(p_i(x) = R_i^{p_{i-1}}(x)\right)$$

(This constraint is to be thought of as a constraint on the single polynomial p.)

The circuit associated with each constraint C_x checks the polynomial p at $k \approx (m+2)(h+1) \leq b$ points and has size s which is of the same order as k. Since p is of degree d which is at most b, we have constructed an instance $(1^n, d, k, s, \mathbb{F}; C_1, \ldots, C_t)$ of $\mathrm{GapPCS}_{\epsilon, m+1, b, q}$ where $d, k, s \leq b$ and $t = q^m$. It follows from Proposition 3.2.6, that this instance $(1^n, d, k, s, \mathbb{F}; C_1, \ldots, C_t)$ satisfies the following lemma.

Proposition 3.2.9 *Suppose, we have an instance* $(1^n, d, k, s, \mathbb{F}; C_1, \ldots, C_t)$ *of* $\mathrm{GapPCS}_{\epsilon, m+1, b, q}$ *constructed from an instance* $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6)$ *of* $\mathrm{AP}_{m,h}$ *as mentioned above.*

- *[Completeness] If* $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6) \in \mathrm{AP}_{m,h}$*, then there exists a polynomial* $p : \mathbb{F}^{m+1} \to \mathbb{F}$ *of degree at most* d *such that* p *satisfies all the constraints* C_i *(i.e.,* $A_i(p(x_1^{(i)}, \ldots, p(x_k^{(i)}) = 0$*)*

- *[Soundness] If there exist polynomial* $p : \mathbb{F}^{m+1} \to \mathbb{F}$ *of degree at most* d *which satisfies at least* ϵ *fraction of the constraints, then* $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6) \in \mathrm{AP}_{m,h}$.

The completeness part of this proposition is clear by construction. For the soundness part, it is to be noted that if at least $(c+1)mh/q$ fraction of the constraints are satisfied, then the soundness condition in Proposition 3.2.6 implies that $(1^n, H, T, \psi, \rho_1, \ldots, \rho_6) \in \mathrm{AP}_{m,h}$. The only observation to be made is that $\epsilon \geq b/q \geq 2cmh/q \geq (c+1)mh/q$.

This proposition completes the proof of the lemma.

□

Lemma 3.2.2 now follows from Lemma 3.1.2 and Lemma 3.2.8.

CHAPTER 4

Low Degree Test

Low-degree tests have been a subject of much research in the context of program checking and PCPs. We use the reduction of SAT to GapPCS described in Chapter 3 to construct MIPs that are efficient in randomness. The MIPs for GapPCS consists of a proof (or more correctly a prover) which is a polynomial provided as a table of values. The MIP verifier before checking whether the polynomial provided by the prover satisfies the constraints of the GapPCS problem, needs to verify that the table of values supplied by the prover is indeed close to a polynomial. Low-degree tests are procedures designed to address this verification step ,i.e., to verify that an arbitrary function $f : \mathbb{F}^m \to \mathbb{F}$ is close to some (unknown) polynomial p of degree d. For our purposes, we need tests that have very low probability of error. Two such tests with analyses are known, one due to Raz and Safra [25] and another due to Rubinfeld and Sudan [26] (with low-error analysis by Arora and Sudan [3]) For our purposes the test of Raz and Safra [25] is more efficient than that of Arora and Sudan [3] for reasons which we will explain shortly.

4.1 The Plane-Point Test

A plane in $\mathbb{F}^m$ is a collection of points parametrized by two variables. Specifically, given $a, b, c \in \mathbb{F}^m$ the plane $\wp_{a,b,c} = \{\wp_{a,b,c}(t_1, t_2) = a + t_1 b + t_2 c | t_1, t_2 \in \mathbb{F}\}$. Several parameterizations are possible for a given plane. We assume some canonical one is fixed for every plane, and thus the plane is equivalent to the set of points it contains. The low-degree test uses the fact that for any polynomial $p : \mathbb{F}^m \to \mathbb{F}$ of degree at most d, the function $p_\wp : \mathbb{F}^2 \to \mathbb{F}$ given by $p_\wp(t_1, t_2) = p(\wp(t_1, t_2))$ is a bivariate polynomial of degree at most d. The verifier tests this property for a function f by picking a random plane through $\mathbb{F}^m$ and verifying that there *exists* a bivariate polynomial that has good agreement with f restricted to this plane. The verifier expects an auxiliary oracle f_{planes} that gives such a bivariate polynomial for every plane. This motivates the test below.

Low-Degree Test (Plane-Point Test)

Input: A function $f : \mathbb{F}^m \to \mathbb{F}$ and an oracle f_{planes}, which for each plane in $\mathbb{F}^m$ gives a bivariate degree d polynomial.

1. Choose a random point in the space $x \in_R \mathbb{F}^m$.
2. Choose a random plane $\wp$ passing through x in $\mathbb{F}^m$.
3. Query f_{planes} on $\wp$ to obtain the polynomial $h_\wp$. Query f on x.
4. Accept iff the value of the polynomial $h_\wp$ at x agrees with $f(x)$.

It is clear that if f is a degree d polynomial, then there exists an oracle f_{planes} such that the above test accepts with probability 1. It is non-trivial to prove any converse and Raz and Safra give a strikingly strong converse. Below we work their statement into a form that is convenient for us.

First some more notation. Let $\mathrm{LDT}^{f,f_{\text{planes}}}(x,\wp)$ denote the outcome of the above test on oracle access to f and f_{planes}. Let $f,g : \mathbb{F}^m \to \mathbb{F}$ have agreement δ if $\Pr_{x\in\mathbb{F}^m}[f(x)=g(x)] = \delta$.

Theorem 4.1.1 ([25]) *There exist constants c_0, c_1, c_2, c_3 such that for every positive real δ, integers m, d and field $\mathbb{F}$ satisfying $|\mathbb{F}| \geq c_0 d(m/\delta)^{c_1}$, the following holds: Let $f : \mathbb{F}^m \to \mathbb{F}$ be any function. If there exists an oracle f_{planes} satisfying $\Pr_{x,\wp}[\mathrm{LDT}^{f,f_{\text{planes}}}(x,\wp) = \text{accept}] \geq \delta$, then there exists a polynomial $p : \mathbb{F}^m \to \mathbb{F}$ of degree at most d such that p and f agree on at least δ^{c_2}/c_3 fraction of the points.*

The above theorem statement of Raz and Safra [25] relates the probability of a function f passing the low degree test with the agreement of f with some polynomial of low degree. The form of the statement which will be most convenient for us to work with is one which states that the probability of the low degree test passing on points at which f does not agree with any of the polynomials it has high agreement with is very low. We work the above statement of Raz and Safra into the following form. We present the proof of Theorem 4.1.2 starting from the statement of Raz and Safra (Theorem 4.1.1) in the subsequent section.

Theorem 4.1.2 *There exist constants c, c' such that for every positive real δ, integers m, d and field $\mathbb{F}$ satisfying $|\mathbb{F}| \geq cd(m/\delta)^{c'}$, the following holds: Fix $f : \mathbb{F}^m \to \mathbb{F}$ and f_{planes}. Let $\{P_1, \ldots, P_l\}$ be the set of all m-variate polynomials of degree d that have agreement at least $\delta/2$ with the function $f : \mathbb{F}^m \to \mathbb{F}$. Then*

$$\Pr_{x,\wp}[f(x) \notin \{P_1(x), \ldots, P_l(x)\} \textit{ and } \mathrm{LDT}^{f,f_{\text{planes}}}(x,\wp) = \text{accept}] \leq \delta.$$

A cubic blowup in the proof-size of the MIPs we will be constructing (in Chapter 5) occurs from the oracle f_{planes} which has size cubic in the size of the oracle f. A possible way to make the proof shorter would be to use an oracle for f restricted only to lines. (i.e., an analogous line-point test to the above test) The analysis of [3] does apply to such a test. However they require the field size to be (at least) a fourth power of the degree; and this results in a blowup in the proof to (at least) an eighth power. Note that the above theorem only needs a linear relationship between the degree and the field size.

4.2 *Stronger Forms of the LDT*

In this section, we shall prove stronger forms of Theorem 4.1.1 and finally prove the form of the theorem (Theorem 4.1.2) which is most convenient to us. The first strong from of the theorem is as follows:

Theorem 4.2.1 *Let c_0, c_1, c_2, c_3 be the constants that appear in Theorem 4.1.1. For every positive real δ, integers m, d and field $\mathbb{F}$ satisfying $|\mathbb{F}| \geq c_0 d(m/\delta)^{c_1}$, the following holds: Fix $f : \mathbb{F}^m \to \mathbb{F}$ and f_{planes}. Let $\{P_1, \ldots, P_l\}$ be the set of all m-variate polynomials of degree d that have agreement at least $\delta^{c_2}/2c_3$ with the function $f : \mathbb{F}^m \to \mathbb{F}$. Then*

$$\Pr_{x,\wp}[f(x) \notin \{P_1(x), \ldots, P_l(x)\} \textit{ and } \mathrm{LDT}^{f, f_{\text{planes}}}(x, \wp) = \text{accept}] \leq \delta.$$

Proof Suppose, $\Pr_{x,\wp}[f(x) \notin \{P_1(x), \ldots, P_l(x)\}$ and $\mathrm{LDT}^{f, f_{\text{planes}}}(x, \wp) = \text{accept}] > \delta$. Let $S \subseteq \mathbb{F}^m$ be the set of all points in $\mathbb{F}^m$ at which f does not agree with any of $P_1, \ldots, P_l$. Then by our hypothesis, $f\Big|_S$ passes the low-degree test (Plane-point test) with probability at least δ. We can now extend $f\Big|_S$ to a function $g : \mathbb{F}^m \to \mathbb{F}$ on the entire domain $\mathbb{F}^m$ by setting the value of g at points not in S randomly. As g passes the test low degree test with probability at least δ, by Theorem 4.1.1, we have that there exists a polynomial $P : \mathbb{F}^m \to \mathbb{F}$ of degree at most d that agrees with g on at least δ^{c_2}/c_3 fraction of the points in $\mathbb{F}^m$. The points of agreement of P with g must be concentrated in S as the value of g at points in $\mathbb{F}^m - S$ is random. Note the a random function has agreement approximately $1/|\mathbb{F}|$ with every degree d polynomial. Thus, P agrees with $f\Big|_S$ on at least $\frac{\delta^{c_2}}{2c_3}|\mathbb{F}^m|$ points in S. As f is different from each of $P_1, \ldots, P_l$ in S, this polynomial P must be different from $P_1, \ldots, P_l$. Thus, we have a polynomial other than $P_1, \ldots, P_l$ that agrees with f on $\delta^{c_2}/2c_3$ fraction of points in $\mathbb{F}^m$. But this is a contradiction as $\{P_1, \ldots, P_l\}$ is the set of all polynomial that have at least $\delta^{c_2}/2c_3$ agreement with f. $\square$

Now, for some more notation. Fix $f : \mathbb{F}^m \to \mathbb{F}$ and an oracle f_{planes}. Let the success probability of a point $x \in \mathbb{F}^m$ be defined as the fraction of planes $\wp$ passing through x such that the value of the polynomial $f_{\text{planes}}(\wp)$ at x agrees with $f(x)$. The success probability of a plane $\wp$ is defined to be the fraction of points x on the plane $\wp$ such that $f_{\text{planes}}(\wp)$ at x agrees with $f(x)$. Note, by this definition

$$E_{x \in \mathbb{F}^m} [\text{ Success probability of } x\ ] = E_{\wp - \text{ plane}} [\text{ Success probability of } \wp\ ] = \Pr_{x,\wp}[\mathrm{LDT}^{f,f_{\text{planes}}} = \text{ accept }]$$

We are now ready to prove the next stronger form of Theorem 4.1.1.

Theorem 4.2.2 *There exist constants c, c' such that for every positive real δ, integers m, d and field $\mathbb{F}$ satisfying $|\mathbb{F}| \geq cd(m/\delta)^{c'}$, the following holds: Let $f : \mathbb{F}^m \to \mathbb{F}$ be any function. If there exists a oracle f_{planes} satisfying $\Pr_{x,\wp}[\mathrm{LDT}^{f,f_{\text{planes}}}(x,\wp) = \text{accept}] \geq \delta$, then there exists a polynomial $p : \mathbb{F}^m \to \mathbb{F}$ of degree at most d such that p and f agree on at least $3\delta/4$ fraction of the points.*

Proof

Let $\wp$ be a random plane. Since $E_{\wp - \text{ plane}} [\text{ Success probability of }] \geq \delta$, it follows by an averaging argument that with probability at least $\delta/8$, the success probability of $\wp$ is at least $7\delta/8$. In other words, if for a random plane $\wp$, $E(\wp)$ denotes the event that there exists a bivariate polynomial $g_\wp : \mathbb{F}^2 \to \mathbb{F}$ of degree at most d that agrees with f on at least $7\delta/8$ fraction of the points on $\wp$, then

$$\Pr_{\wp}[E(\wp)] \geq \frac{\delta}{8} \tag{4.1}$$

Let c_0, c_1, c_2, c_3 be the constants that appear in Theorem 4.1.1. Let $P_1, \ldots, P_l$ be all the polynomials of degree at most d that agree with f on at least $\frac{1}{2c_3}\left(\frac{\delta^2}{20}\right)^{c_2}$ fraction of the points of $\mathbb{F}^m$. Note that $l \leq 4c_3\left(\frac{20}{\delta^2}\right)^{c_2}$. Define $\rho_1, \ldots, \rho_l$ such that $\rho_i = \Pr_{x \in \mathbb{F}^m}[P_i(x) = f(x)]$ (i.e., agreement of P_i and f). If we show that there exists an i such that $\rho_i \geq 3\delta/4$, we would be done. We will assume the contrary and obtain a contradiction to (4.1).

Suppose for all $i = 1, \ldots, l$, $\rho_i < 3\delta/4$. Let $\wp$ be any plane such that the event $E(\wp)$ occurs. Then, the bivariate polynomial $g_\wp$ that is described in the event $E(\wp)$ should satisfy one of the following.

Case (i) $g_\wp \notin \{P_1\big|_\wp, \ldots, P_l\big|_\wp\}$. (i.e., $g_\wp$ is not the restriction of any of the P_i's to the plane $\wp$.)

Case (ii) $g_\wp \in \{P_1\big|_\wp, \ldots, P_l\big|_\wp\}$. (i.e., $g_\wp$ is the restriction of one of the P_i's to the plane $\wp$.)

In case (i), we have that $\wp$ is a plane whose success probability is at least $7\delta/8$ and moreover, on at least $7\delta/8 - ld/|\mathbb{F}|$ fraction of the points on $\wp$, the polynomial $g_\wp$ agrees with f but not with any of $P_1, \ldots, P_l$. By Theorem 4.2.1, if $|\mathbb{F}| \geq c_0 d(20m/\delta^2)^{c_1}$, then at most $\delta^2/20$ fraction of the points in $\mathbb{F}^m$ are such that f does not agree with $P_1, \ldots, P_l$ but the low degree test passes at that point. Thus, by an averaging argument it follows that

$$\Pr_{\wp}[\text{ Case } (i) \text{ occurs }] \leq \frac{\delta^2}{20(\frac{7\delta}{8} - \frac{ld}{|\mathbb{F}|})}$$

If $|\mathbb{F}| > 2^{2c_2+5}5^{c_2+1}c_3 d/3\delta^{c_2+1}$, then $|\mathbb{F}| > 40ld/3\delta$ and the above probability is less than $\delta/16$. Thus, if $\mathbb{F}$ is chosen in such a manner, the probability of case(i) happening is less than $\delta/16$.

In case (ii), for $i = 1, \dots, l$, define the random variable γ_i to denote the fraction of points on the random plane $\wp$ at which P_i agrees with f. We have that for each i, $E_\wp[\gamma_i] = \rho_i$. An application of Chebyshev's inequality tells us that for each $i = 1, \dots, l$,

$$\Pr_\wp\left[\gamma_i - \rho_i > \frac{\delta}{8}\right] \leq \frac{64\rho_i}{\delta^2|\mathbb{F}|^2}$$

As we have by our assumption that $\rho_i < 3\delta/4$, we have that

$$\Pr_\wp\left[\exists i, \gamma_i > \frac{7\delta}{8}\right] \leq l \times \frac{64\rho_i}{\delta^2|\mathbb{F}|^2} \leq \frac{2^{2c_2+8}5^{c_2}c_3}{|\mathbb{F}|^2\delta^{2c_2+1}}$$

If we choose $\mathbb{F}$ such that $|\mathbb{F}| \geq 2^{c_2+6}5^{c_2/2}\sqrt{c_3}/\delta^{c_2+1}$, then the above probability is less than $\delta/16$. Note that the probability on the LHS is an upper bound on the $\Pr_\wp[\text{ Case } (ii) \text{ occurs }]$. Thus, case (ii) happens with probability less than $\delta/16$.

Let c, c' be sufficiently large constants such that $|\mathbb{F}| \geq cd(m/\delta)^{c'}$ implies the three inequalities $|\mathbb{F}| \geq c_0 d(20m/\delta^2)^{c_1}$, $|\mathbb{F}| > 2^{2c_2+5}5^{c_2+1}c_3 d/3\delta^{c_2+1}$ and $|\mathbb{F}| \geq 2^{c_2+6}5^{c_2/2}\sqrt{c_3}/\delta^{c_2+1}$. In this case we have that $\Pr_\wp[E(\wp)] = \Pr_\wp[\text{ Case (i) }] + \Pr_\wp[\text{ Case (i) }] < \delta/16 + \delta/16 = \delta/8$. This contradicts (4.1). Hence, there does exist a i such that $\rho_i \geq 3\delta/4$. Thus, for this i, the polynomial P_i and f agree on at least $3\delta/4$ fraction of the points in $\mathbb{F}^m$. □

Theorem 4.1.2 is obtained from Theorem 4.2.2 by mimicking the proof of Theorem 4.2.1 from Theorem 4.1.1.

Chapter 5

Randomness Efficient MIP for SAT

In this chapter, we show how to translate the use of state-of-the-art low-degree tests, in particular the test of Raz and Safra [25], in conjunction with the hardness of PCS to obtain a 3-prover MIP for SAT.

Using GapPCS it is easy to produce a simple probabilistically checkable proof for SAT. Given an instance of SAT, reduce it to an instance $\mathcal{I}$ of GapPCS ; and provide as proof the polynomial $p : \mathbb{F}^m \to \mathbb{F}$ as a table of values. To verify correctness a verifier first "checks" that p is close to some polynomial using the low degree test and then verifies that a random constraint C_j is satisfied by p.

A proof for such a PCP (MIP) system would be an oracle f representing the polynomial and the auxiliary oracle f_{planes}. The verifier performs a low-degree test on f and then picks a random constraint C_j and verifies that C_j is satisfied by the assignment f. But the naive implementation would make k queries to the oracle f and this is too many queries. The same problem was faced by Arora, Lund, Motwani, Sudan and Szegedy [1] who solved it by running a curve through the k points and then asking a new oracle f_{curves} to return the value of f restricted to this curve. This solution cuts down the number of queries to 3, but the analysis of correctness works only if $|\mathbb{F}| \geq kd$. In our case, this would impose an additional quadratic blowup in the proof-size and we would like to avoid this. We do so by picking r-dimensional varieties (algebraic surfaces) that pass through the given k points. This cuts down the degree to $rk^{1/r}$. However some additional complications arise: The variety needs to pass through many random points, but not at the expense of too much randomness. We deal with these issues in the following section.

5.1 MIP Verifier

A variety $\mathcal{V} : \mathbb{F}^r \to \mathbb{F}^m$ is a collection of m functions, $\mathcal{V} = \langle \mathcal{V}_1, \dots, \mathcal{V}_m \rangle$, $\mathcal{V}_i : \mathbb{F}^r \to \mathbb{F}$. A variety is of degree D if all the functions $\mathcal{V}_1, \dots, \mathcal{V}_m$ are polynomials of degree D. For a variety $\mathcal{V}$ and function $f : \mathbb{F}^m \to \mathbb{F}$, the restriction of f to $\mathcal{V}$ is the function $f\Big|_{\mathcal{V}} : \mathbb{F}^r \to \mathbb{F}$ given by $f\Big|_{\mathcal{V}}(a_1, \dots, a_r) = f(\mathcal{V}(a_1, \dots, a_r))$. Note that the restriction of a degree d polynomial $p : \mathbb{F}^m \to \mathbb{F}$ to an r-dimensional variety $\mathcal{V}$ of degree D is an r-variate polynomial of degree Dd.

Let $S \subseteq \mathbb{F}$ be of cardinality $k^{1/r}$. Let $z_1, \dots, z_k$ be some canonical ordering of the points in S^r. Let $\mathcal{V}^{(0)}_{S,x_1,\dots,x_k} : \mathbb{F}^r \to \mathbb{F}^m$ denote a canonical variety of degree $r|S|$ that satisfies $\mathcal{V}^{(0)}_{S,x_1,\dots,x_k}(z_i) = x_i$ for every $i \in \{1, \dots, k\}$. Let $Z_S : \mathbb{F}^r \to \mathbb{F}$ be the function given by $Z_S(y_1, \dots, y_r) = \prod_{i=1}^{r} \prod_{a \in S}(y_i - a)$; i.e. $Z_S(z_i) = 0$. Let $\alpha = \langle \alpha_1, \dots, \alpha_m \rangle \in \mathbb{F}^m$. Let $\mathcal{V}^{(1)}_{S,\alpha}$ be the variety $\langle \alpha_1 Z_S, \dots, \alpha_m Z_S \rangle$. We will let $\mathcal{V}_{S,\alpha,x_1,\dots,x_k}$ be the variety $\mathcal{V}^{(0)}_{S,x_1,\dots,x_k} + \mathcal{V}^{(1)}_{S,\alpha}$. Note that if α is chosen at random, $\mathcal{V}_{S,\alpha,x_1,\dots,x_k}(z_i) = x_i$ for $z_i \in S^r$ and $\mathcal{V}_{S,\alpha,x_1,\dots,x_k}(z)$ is distributed uniformly over $\mathbb{F}^m$ if $z \in (\mathbb{F} - S)^r$. These varieties will replace the role of the curves of [1].

We are now ready to describe the MIP verifier for $\mathrm{GapPCS}_{\epsilon,m,b,q}$. (Henceforth, we shall assume that t, the number of constraints in $\mathrm{GapPCS}_{\epsilon,m,b,q}$ instance is at most q^{2m}. In fact, for our reduction from SAT (Lemma 3.2.2), t is exactly equal to q^m.)

MIP Verifier$^{f, f_{\text{planes}}, f_{\text{varieties}}}(1^n, d, k, s, \mathbb{F}; C_1, \dots, C_t)$.

Notation: r is a parameter to be specified. Let $S \subseteq \mathbb{F}$ be such that $|S| = k^{1/r}$.

1. Pick $a, b, c \in \mathbb{F}^m$ and $z \in (\mathbb{F} - S)^r$ at random.
2. Let $\wp = \wp_{a,b,c}$. Use b, c to compute $j \in \{1, \dots, t\}$ at random (i.e., j is fixed given b, c, but is distributed uniformly when b and c are random.) Compute α such that $\mathcal{V}(z) = a$ for $\mathcal{V} = \mathcal{V}_{S,\alpha,x_1^{(j)},\dots,x_k^{(j)}}$.
3. Query $f(a)$, $f_{\text{planes}}(\wp)$ and $f_{\text{varieties}}(\mathcal{V})$. Let $g = f_{\text{planes}}(\wp)$ and $h = f_{\text{varieties}}(\mathcal{V})$.
4. Accept if all the conditions below are true:
   (a) g and f agree at a.
   (b) h and f agree at a.
   (c) A_j accepts the inputs $h(z_1), \dots, h(z_k)$.

Complexity: Clearly the verifier V makes exactly 3 queries. Also, exactly $3m \log q + r \log q$ random bits are used by the verifier. The answer sizes are no more than $O((drk^{1/r} + r)^r \log q)$ bits.

5.1.1 Completeness and Soundness of MIP Verifier

Now to prove the correctness of the verifier. Clearly, if the input instance is a YES instance then there exists a polynomial P of degree d that satisfies all the constraints of the input instance. Choosing $f = P$ and constructing f_{planes} and $f_{\text{varieties}}$ to be restrictions of P to the respective planes and varieties, we notice that the MIP verifier accepts with probability one. We now bound the soundness of the verifier.

Claim 5.1.1 *Let δ be any constant that satisfies the conditions of Theorem 4.1.2 and $\delta \geq 2\sqrt{\frac{d}{q}}$. Then the soundness of the MIP Verifier is at most*

$$\delta + \frac{4\epsilon}{\delta} + \frac{4rk^{\frac{1}{r}}d}{\delta(q - k^{\frac{1}{r}})}$$

Proof Let $P_1, \ldots, P_l$ be all the polynomials of degree d that have agreement at least $\delta/2$ with f. (Note $l \leq 4/\delta$ since $\delta \geq 2\sqrt{d/q}$.) Now suppose, the MIP Verifier had accepted a NO instance, then one of the following events must have taken place.

Event 1: $f(a) \notin \{P_1(a), \ldots, P_l(a)\}$ and $\mathrm{LDT}^{f, f_{\text{planes}}}(a, \wp) = \text{accept}$.

We have from Theorem 4.1.2, that Event 1 could have happened with probability at most δ.

Event 2: $\exists i \in \{1, \ldots, l\}$, such that constraint C_j is satisfiable with respect to polynomial P_i. (i.e., $A_j(P_i(x_1^{(j)}), \ldots, P_i(x_k^{(j)})) = 0$).

As the input instance is a NO instance of $\mathrm{GapPCS}_{\epsilon,m,b,q}$, this events happens with probability at most $l\epsilon \leq 4\epsilon/\delta$.

Event 3: $\forall i$, $P_i\Big|_{\mathcal{V}} \neq h$, but the value of h at a is contained in $\{P_1(a), \ldots, P_l(a)\}$.

To see this part, we reinterpret the randomness of the MIP verifier. First pick $b, c, \alpha \in \mathbb{F}^m$. From this we generate the constraint C_j and this defines the variety $\mathcal{V} = \mathcal{V}_{S,\alpha,x_1^{(j)},\ldots,x_k^{(j)}}$. Now we pick $z \in (\mathbb{F} - S)^r$ at random and this defines $a = \mathcal{V}(z)$. We can bound the probability of the event in consideration after we have chosen $\mathcal{V}$, as purely a function of the random variable z as follows. Fix any i and $\mathcal{V}$ such that $P_i\Big|_{\mathcal{V}} \neq h$. Note that the value of h at a equals $h(z)$ (by definition. of a, z and $\mathcal{V}$). Further $P_i(a) = P_i\Big|_{\mathcal{V}}(z)$. But z is chosen at random from $(\mathbb{F} - S)^r$. By the Schwartz's lemma (Lemma 3.2.7), the probability of agreement on this domain is at most $rk^{1/r}d/(|\mathbb{F}| - |S|)$. Using the union bound over the i's we get that this event happens with probability at most $lrk^{1/r}d/(|\mathbb{F}| - |S|) \leq 4rk^{\frac{1}{r}}d/\delta(q - k^{\frac{1}{r}})$.

We thus have that the probability of one of the above events occurring is at most $\delta + 4\epsilon/\delta + 4rk^{\frac{1}{r}}d/\delta(q - k^{\frac{1}{r}})$.

We would be done if we show that if none of the three events occur, then the MIP verifier rejects. Suppose none of the three events took place. In other words, all the following happened

- $f(a) \in \{P_1(a), \ldots, P_l(a)\}$ or $\mathrm{LDT}^{f, f_{\text{planes}}}(a, \wp) = $ reject. We could as well assume that $f(a) \in \{P_1(a), \ldots, P_l(a)\}$ for in the other case (i.e., LDT rejects), the verifier rejects.

- $\forall i$, $A_j(P_i(x_1^{(j)}), \ldots, P_i(x_k^{(j)}) \neq 0$.

- $\exists i$, $P_i\Big|_{\mathcal{V}} = h$ or the value of h at a is not contained in $\{P_1(a), \ldots, P_l(a)\}$.

If h at a is not one of $P_1(a), \ldots, P_l(a)$, then the MIP verifier rejects as $f(a) \in \{P_1(a), \ldots, P_l(a)\}$. So, if the MIP verifier had accepted, it should be the case that $\exists i$, $P_i\Big|_{\mathcal{V}} = h$. But as $\forall i$, $A_j(P_i(x_1^{(j)}, \ldots, P_i(x_k^{(j)}) \neq 0$, the verifier is bound to reject in this case too. Thus, if none of the the three events occurred, then the verifier should have rejected. □

5.2 *Proof of Lemma 2.2.1*

We can now complete the construction of a 3-prover MIP for SAT and give the proof of Lemma 2.2.1.

Proof of Lemma 2.2.1: Choose $\delta = \frac{\mu}{3}$. Let c, c' be the constants that appear in Theorem 4.1.2. Choose $\varepsilon' = \varepsilon/2$ where ε is the soundness of the MIP, we wish to prove. Choose $\epsilon = \min\{\delta\mu/12, \varepsilon'/3(9+c')\}$. Let n be the size of the SAT instance. Let $m = \epsilon \log n / \log\log n$, $b = (\log n)^{3+\frac{1}{\epsilon}}$ and $q = (\log n)^{9+c'+\frac{1}{\epsilon}}$. Note that this choice of parameters satisfies the requirements of Lemma 3.2.2. Hence, SAT reduces to $\mathrm{GapPCS}_{\epsilon,m,b,q}$ under length preserving reductions. Combining this reduction with the MIP verifier for GapPCS, we have a MIP verifier for SAT. Also δ satisfies the requirements of Claim 5.1.1. Thus, this MIP verifier has soundness as given by Claim 5.1.1. Setting $r = \frac{1}{\epsilon}$, we can easily check that for sufficiently large n, $4rk^{\frac{1}{r}}d/\delta(q - k^{\frac{1}{r}}) \leq 8rk^{\frac{1}{r}}d/q\delta \leq \mu/3$. We thus have that the soundness of the MIP verifier is at most $\delta + 4\epsilon/\delta + \mu/3 \leq \mu$. The randomness used is exactly $3m\log q + r\log q$ which with the present choice of parameters is $(3+\varepsilon')\log n + \mathrm{poly}\log n \leq (3+\varepsilon)\log n$. The answer sizes are clearly $\mathrm{poly}\log n$. Thus, $\mathrm{SAT} \in \mathrm{MIP}_{1,\frac{1}{2}+\mu}[(3+\varepsilon)\log n, \mathrm{poly}\log n]$. □

CHAPTER 6

Constant Query Inner Verifier for MIPs

In this chapter, we truncate the recursion by constructing a constant query "inner verifier" for a p-prover interactive proof system.

6.1 Inner Verifier

6.1.1 Introduction

An inner verifier is a subroutine designed to simplify the task of an MIP verifier. Say an MIP verifier V_{out}, on input x and random string R, generated queries $q_1, \ldots, q_p$ and a linear sized circuit C. In the standard protocol the verifier would send query q_i to prover Π_i and receive some answer a_i. The verifier accepts if $C(a_1, \ldots, a_p) = -1$. (In this section, we will assume all Boolean functions map to $\{+1, -1\}$ with -1 representing the logical true.) An inner verifier reduces the answer size complexity of this protocol by accessing oracles $A_1, \ldots, A_p$ supposedly encoding the responses $a_1, \ldots, a_p$, and an auxiliary oracle B; and probabilistically verifying that the A_i's really correspond to some commitment to strings $a_1, \ldots, a_p$ that satisfy the circuit C. The hope is to get the inner verifier to do all this with very few queries to the oracles $A_1, \ldots, A_p$ and B and we do so with one (bit) query each to the A_i's and seven queries to B.

6.1.2 Details of the Inner Verifier

Let $\mathcal{A} = \{+1, -1\}^a$ and $\mathcal{B} = \{(a_1, \ldots, a_p) | C(a_1, \ldots, a_p) = -1\}$. Let π_i be the projection function $\pi_i : \mathcal{B} \to \mathcal{A}$ which maps $(a_1, \ldots, a_p)$ to a_i. By abuse of notation, for $\beta \subseteq \mathcal{B}$, let $\pi_i(\beta)$ denote $\{\pi_i(x) | x \in S\}$. Queries to the oracle A_i will be functions $f : \mathcal{A} \to \{+1, -1\}$. Queries to the oracle B

will be functions $g : \mathcal{B} \to \{+1, -1\}$. The inner verifier expects the oracles to provide the long codes of the strings $a_1, \ldots, a_p$, i.e., $A_i(f) = f(a_i)$ and $B(g) = g(a_1, \ldots, a_p)$. Of course, we can not assume these properties; they need to be verified explicitly by the inner verifier. We will assume however that the tables are "folded", i.e., $A_i(f) = -A_i(-f)$ and $B(g) = -B(-g)$ for every i, f, g. (This is implemented by issuing only one of the queries f or $-f$ for every f and inferring the other value, if needed by complementing it.)[1]We are now ready to specify the inner verifier.

${V_{\text{inner}}}^{A_1,\ldots,A_p,B}(\mathcal{A}, \mathcal{B}, \pi_1, \ldots, \pi_p)$.

1. For each each $i \in \{1, \ldots, p\}$, choose $f_i : \mathcal{A} \to \{+1, -1\}$ at random.
2. Choose $f, g_1, g_2, h_1, h_2 : \mathcal{B} \to \{+1, -1\}$ at random and independently.
3. Let $g = f\,(g_1 \wedge g_2)\,(\Pi f_i \circ \pi_i))$ and $h = f\,(h_1 \wedge h_2)\,(\Pi f_i \circ \pi_i))$.
4. Read the following bits from the oracles $A_1, \ldots, A_p, B$

   $y_i = A_i(f_i)$, for each $i \in \{1, \ldots, p\}$.

   $w = B(f)$.

   $u_1 = B(g_1); u_2 = B(g_2)$

   $v_1 = B(h_1); v_2 = B(h_2)$

   $z_1 = B(g); z_2 = B(h)$
5. Accept iff
$$w \prod_{i=1}^{p} y_i = (u_1 \wedge u_2)z_1 = (v_1 \wedge v_2)z_2$$

Clearly, the number of queries made by V_{inner} is exactly 7 while the randomness needed by it is $p|\mathcal{A}| + 5|\mathcal{B}| \leq p2^a + 52^{pa} = O(2^{pa})$.

6.1.3 *Completeness and Soundness of Inner Verifier*

It is clear that if $a_1, \ldots, a_p$ are such that $C(a_1, \ldots, a_p) = -1$ and for every i and f, $A_i(f) = f(a_i)$ and for every g, $B(g) = g(a_1, \ldots, a_p)$, then the inner verifier accepts with probability one. The following lemma gives a soundness condition for the inner verifier, by showing that if the acceptance probability of the inner verifier is sufficiently high then the oracles $A_1, \ldots, A_p$ are non-trivially close to the encoding of strings $a_1, \ldots, a_p$ that satisfy $C(a_1, \ldots, a_p) = -1$. The proof uses, by now standard, Fourier analysis.

Note that the oracle A_i can be viewed as a function mapping the set $\{\mathcal{A} \to \{+1, -1\}\}$ to the reals. Let the inner product of two oracles A and A' be $\langle A, A' \rangle = 2^{-|\mathcal{A}|} \sum_f A(f)A'(f)$. For $\alpha \subseteq \mathcal{A}$, let $\chi_\alpha(f) = \prod_{a \in \alpha} f(a)$. Then the χ_α's give an orthonormal basis for the space of oracles A. This

[1]The folded condition in terms of Fourier coefficients translates to $\hat{A}_\alpha = 0$ for all α such that $|\alpha|$ is even. More specifically, $\hat{A}_\phi = 0$.

allows us to express $A(\cdot) = \sum_\alpha \hat{A}_\alpha \chi_\alpha(\cdot)$, where $\hat{A}_\alpha = \langle A, \chi_\alpha \rangle$ are the Fourier coefficients of A. In what follows, we let $\hat{A}_{i,\alpha}$ denote the α^{th} Fourier coefficient of the table A_i. Similarly one can define a basis for the space of oracles B and the Fourier coefficients of any one oracle.

Our next lemma lays out the precise soundness condition in terms of the Fourier coefficients of the oracles $A_1, \dots, A_p$.

Claim 6.1.1 *For every $\varepsilon > 0$, there exists a $\delta(= \delta_\varepsilon) > 0$ such that if ${V_{\text{inner}}}^{A_1,\dots,A_p,B}(\mathcal{A}, \mathcal{B}, \pi_1, \dots, \pi_p)$ accepts with probability at least $\frac{1}{2} + \varepsilon$, then there exist $a_1, \dots, a_p \in \mathcal{A}$ such that $C(a_1, \dots, a_p) = -1$ and $|\hat{A}_{i,\{a_i\}}| \geq \delta$ for every $i \in \{1, \dots, p\}$.*

Proof Let δ be some constant (to be decided later.) Assume that there do not exist $a_1, \dots, a_p \in \mathcal{A}$ such that $C(a_1, \dots, a_p) = -1$ and $|\hat{A}_{i,\{a_i\}}| \geq \delta$ for every $i \in \{1, \dots, p\}$. On restating this assumption, we get that for every $\beta \subseteq \mathcal{B}$ such that $|\beta| = 1$, there exists a $i \in \{1, \dots, p\}$ such that $|\hat{A}_{i,\pi_i(\beta)}| < \delta$. To prove the lemma, it is sufficient if we show that for a particular choice of δ, this assumption implies that the acceptance probability of V_{inner} is less than $\frac{1}{2} + \varepsilon$.

The acceptance condition of the verifier V_{inner} can be given by the following expression.

$$ACC = \frac{1}{4}\left(1 + w(u_1 \wedge u_2) z_1 \prod_{i=1}^{p} y_i\right)\left(1 + w(v_1 \wedge v_2) z_2 \prod_{i=1}^{p} y_i\right)$$

Thus, the acceptance probability ($E[ACC]$) of V_{inner} is exactly equal to

$$\frac{1}{4}E\left[\left(1 + B(f)\left(B(g_1) \wedge B(g_2)\right) B(g) \prod_{i=1}^{p} A_i(f_i)\right)\left(1 + B(f)\left(B(h_1) \wedge B(h_2)\right) B(h) \prod_{i=1}^{p} A_i(f_i)\right)\right]$$

where the expectation is taken over the random choices of the functions f_i, f, g_1, g_2, h_1 and h_2. This expression can be simplified to

$$E[ACC] = \frac{1}{4} + \frac{1}{2}E_{f_i,f,g_1,g_2}\left[B(f)\left(B(g_1) \wedge B(g_2)\right) B(g) \prod_{i=1}^{p} A_i(f_i)\right] \quad (6.1)$$

$$+ \frac{1}{4}E_{f_i,f,g_1,g_2,h_1,h_2}\left[\left(B(g_1) \wedge B(g_2)\right)\left(B(h_1) \wedge B(h_2)\right) B(g) B(h)\right] \quad (6.2)$$

Using Fourier expansion and the fact that $a \wedge b = \frac{1+a+b-ab}{2}$, the expectation in (6.1) can be expressed as follows

$$\begin{aligned}
& \frac{1}{2}E\left[B(f)B(g)\prod_{i=1}^{p}A_i(f_i)\right] + E\left[B(f)B(g)B(g_1)\prod_{i=1}^{p}A_i(f_i)\right] \\
- & \frac{1}{2}E\left[B(f)B(g)B(g_1)B(g_2)\prod_{i=1}^{p}A_i(f_i)\right] \\
= & \frac{1}{2}\sum_{\beta}\hat{B}_\beta^2\prod_{i=1}^{p}\hat{A}_{i,\alpha_i}\left(\frac{1}{2}\right)^{|\beta|} + \sum_{\beta}\sum_{\beta_1\subseteq\beta}\hat{B}_\beta^2\hat{B}_{\beta_1}\prod_{i=1}^{p}\hat{A}_{i,\alpha_i}\left(\frac{1}{2}\right)^{|\beta|} \\
- & \frac{1}{2}\sum_{\beta}\sum_{\beta_1,\beta_2\subseteq\beta}\hat{B}_\beta^2\hat{B}_{\beta_1}\hat{B}_{\beta_2}\prod_{i=1}^{p}\hat{A}_{i,\alpha_i}(-1)^{|\beta_1\cap\beta_2|}\left(\frac{1}{2}\right)^{|\beta|} \\
\leq & \frac{1}{2}\sum_{\beta}\hat{B}_\beta^2\prod_{i=1}^{p}|\hat{A}_{i,\alpha_i}|\left(\frac{1}{2}\right)^{|\beta|}\left(1+\sum_{\beta_1\subseteq\beta}|\hat{B}_{\beta_1}|\right)^2
\end{aligned}$$

The other expectation in (6.2) in the acceptance probability can be simplified to

$$\begin{aligned}
\frac{1}{4}E\left[B(g)B(h)\right] + E\left[B(g_1)B(g)B(h)\right] - \frac{1}{2}E\left[B(g_1)B(g_2)B(g)B(h)\right] + E\left[B(g_1)B(h_1)B(g)B(h)\right] \\
-E\left[B(g_1)B(g_1)B(h_1)B(g)B(h)\right] + E\left[B(g_1)B(g_1)B(h_1)B(h_2)B(g)B(h)\right]
\end{aligned}$$

This expression can be further simplified to

$$\begin{aligned}
\frac{1}{4}\sum_{\beta}\hat{B}_\beta^2\left(\frac{1}{4}\right)^{|\beta|} + \sum_{\beta}\sum_{\beta_1\subseteq\beta}\hat{B}_\beta^2\hat{B}_{\beta_1}\left(\frac{1}{4}\right)^{|\beta|} - \frac{1}{2}\sum_{\beta}\sum_{\beta_1,\beta_2\subseteq\beta}\hat{B}_\beta^2\hat{B}_{\beta_1}\hat{B}_{\beta_2}\left(\frac{1}{4}\right)^{|\beta|} \\
+\sum_{\beta}\sum_{\beta_1,\beta_2\subseteq\beta}\hat{B}_\beta^2\hat{B}_{\beta_1}\hat{B}_{\beta_2}\left(\frac{1}{4}\right)^{|\beta|} - \sum_{\beta}\sum_{\beta_1,\beta_2,\beta_3\subseteq\beta}\hat{B}_\beta^2\hat{B}_{\beta_1}\hat{B}_{\beta_2}\hat{B}_{\beta_3}\frac{(-1)^{|\beta_1\cap\beta_2|}}{4^{|\beta|}} \\
+\sum_{\beta}\sum_{\beta_1,\beta_2,\beta_3,\beta_4\subseteq\beta}\hat{B}_\beta^2\hat{B}_{\beta_1}\hat{B}_{\beta_2}\hat{B}_{\beta_3}\hat{B}_{\beta_3}\frac{(-1)^{|\beta_1\cap\beta_2|+|\beta_3\cap\beta_4|}}{4^{|\beta|}}
\end{aligned}$$

This expression can easily be seen to be no more than

$$\frac{1}{4}\sum_{\beta}\hat{B}_\beta^2\left(\frac{1}{4}\right)^{|\beta|}\left(1+\sum_{\beta_1\subseteq\beta}|\hat{B}_{\beta_1}|\right)^2$$

We have thus shown that the acceptance probability is no more than

$$\frac{1}{4}+\frac{1}{4}\sum_{\beta}\hat{B}_\beta^2\left(\prod_{i=1}^{p}|\hat{A}_{i,\pi_i(\beta)}|\frac{(1+\gamma_\beta)^2}{2^{|\beta|}}+\frac{1}{4}\frac{(1+\gamma_\beta)^4}{4^{|\beta|}}\right)$$

where $\gamma_\beta = \sum_{\beta'\subseteq\beta}|\hat{B}_{\beta'}|$.

Define $\eta_1 = \sum_{|\beta|=1}\hat{B}_\beta^2$, $\eta_3 = \sum_{|\beta|=3}\hat{B}_\beta^2$ and $\eta_5 = \sum_{|\beta|\geq 5}\hat{B}_\beta^2$. (Note $\eta_1+\eta_3+\eta_5 = 1$.) With these definitions, the acceptance probability can be shown to be less than

$$\begin{aligned}
\frac{1}{4} \quad + & \quad \frac{1}{4}\left[2\eta_1\delta + \eta_3\frac{(1+\sqrt{1-\eta_1}+\sqrt{3\eta_1})^2}{8} + \frac{25}{32}\eta_5\right] \\
+ & \quad \frac{1}{4}\left[\eta_1\frac{(1+\sqrt{\eta_1})^4}{16} + \eta_3\frac{(1+\sqrt{1-\eta_1}+\sqrt{3\eta_1})^4}{256} + \frac{5^4}{4^6}\eta_5\right]
\end{aligned}$$

This expression is of the form $\lambda_1(\eta_1) + \eta_3\lambda_2(\eta_1) + C\eta_5$ where λ_1, λ_2 are the appropriate functions and C a constant. For a fixed η_1, if $\lambda_2(\eta_1) < C$, then the acceptance probability is at most $\lambda_1(\eta_1) + C(1-\eta_1)$ and otherwise the acceptance probability is at most $\lambda_1(\eta_1)+(1-\eta_1)\lambda_2(\eta_1)$. We shall show that both these expressions are at most $\frac{1}{2} + \frac{\delta}{2}$. The first of these expressions is

$$\frac{1}{4} + \frac{1}{4}\left[2\eta_1\delta + \eta_1\frac{(1+\sqrt{\eta_1})^4}{16} + \left(\frac{25}{32} + \frac{5^4}{4^6}\right)(1-\eta_1)\right]$$

which is at most

$$\frac{1}{4} + \frac{\delta}{2} + \frac{1}{4}\left[\eta_1 + \left(\frac{25}{32} + \frac{5^4}{4^6}\right)(1-\eta_1)\right]$$

Now as $\left(\frac{25}{32} + \frac{5^4}{4^6}\right) < 1$, the expression $\eta_1 + \left(\frac{25}{32} + \frac{5^4}{4^6}\right)(1-\eta_1)$ is at most 1. Hence the above expression is no more than $\frac{1}{2} + \frac{\delta}{2}$ for $\eta_1 \leq 1$. The other expression is

$$\begin{aligned}\frac{1}{4} &+ \frac{1}{4}\left[2\eta_1\delta + \eta_1\frac{(1+\sqrt{\eta_1})^4}{16}\right] \\ &+ \frac{1-\eta_1}{4}\left[\frac{(1+\sqrt{1-\eta_1}+\sqrt{3\eta_1})^2}{8} + \frac{(1+\sqrt{1-\eta_1}+\sqrt{3\eta_1})^4}{256}\right]\end{aligned}$$

From Claim 6.1.2, it follows that this expression is at most $\frac{1}{2} + \frac{\delta}{2}$.

We thus have that the acceptance probability in either case is less than $\frac{1}{2} + \frac{\delta}{2}$. Thus choosing $\delta = 2\varepsilon$, we have that the acceptance probability of V_{inner} is less than $\frac{1}{2}+\varepsilon$, which is what we wanted to prove. $\square$

Claim 6.1.2 *For* $0 \leq \eta \leq 1$,

$$\eta_1\frac{(1+\sqrt{\eta_1})^4}{16} + (1-\eta_1)\left(\frac{(1+\sqrt{1-\eta_1}+\sqrt{3\eta_1})^2}{8} + \frac{(1+\sqrt{1-\eta_1}+\sqrt{3\eta_1})^4}{256}\right)$$

is at most 1

Proof For $\eta_1 \leq 1$, we have that $\sqrt{1-\eta_1} \leq 1 - \eta_1/2$. Using this fact, the above expression is at most

$$\eta_1\frac{(1+\sqrt{\eta_1})^4}{16} + (1-\eta_1)\left(\frac{(2-\frac{\eta_1}{2}+\sqrt{3\eta_1})^2}{8} + \frac{(2-\frac{\eta_1}{2}+\sqrt{3\eta_1})^4}{256}\right)$$

For convenience, let us call the above expression $\mu(\eta_1)$.

Define $\mu'(\eta_1) = \mu((1-\eta_1)^2)$. Note μ' is a polynomial of degree 10 in η_1. In fact $\mu'(\eta_1) =$

$\mu_1(\eta_1) + \mu_2(\eta_1)$, where μ_1 and μ_2 are as defined below.

$$\begin{aligned}
\mu_1(\eta_1) = \quad & 1 + \left(-\frac{4631}{2048} + \frac{255}{256}\sqrt{3}\right)\eta_1 + \left(\frac{18407}{4096} - \frac{497}{512}\sqrt{3}\right)\eta_1^2 + \left(-\frac{567}{128} + \frac{305}{512}\sqrt{3}\right)\eta_1^3 \\
& + \left(\frac{2195}{1024} + \frac{411}{512}\sqrt{3}\right)\eta_1^4 + \left(-\frac{203}{1024} - \frac{169}{512}\sqrt{3}\right)\eta_1^5 \\
\mu_2(\eta_1) = \quad & \left(-\frac{615}{2048} + \frac{77}{512}\sqrt{3}\right)\eta_1^6 + \left(\frac{35}{256} - \frac{35}{512}\sqrt{3}\right)\eta_1^7 + \left(-\frac{25}{1024} + \frac{9}{512}\sqrt{3}\right)\eta_1^8 \\
& + \left(\frac{5}{2048} - \frac{1}{512}\sqrt{3}\right)\eta_1^9 - \frac{1}{4096}\eta_1^{10}
\end{aligned}$$

We can easily check that $\mu_2(\eta_1) \leq 0$ for all $\eta_1 \geq 0$. Thus it suffices, if we show that $\mu_1(\eta_1) \leq 1$ for all $0 \leq \eta_1 \leq 1$. Consider the function $\chi(\eta_1) = (\mu_1(\eta_1) - 1)/\eta_1$. χ is a polynomial of degree 4 in η_1 with a negative leading coefficient. It can easily be checked that the polynomial $\chi(x)$ has no real roots. Hence $\chi(\eta_1) < 0$ for all η_1. Thus, $\mu_1(\eta_1) \leq 1$ for all $0 \leq \eta_1$.

□

6.2 *Composed Verifier*

There is a natural way to compose a p-prover MIP verifier V_{out} with an inner verifier such as V_{inner} above so as to preserve perfect completeness. The number of queries issued by the composed verifier is exactly that of the inner verifier. The randomness is the sum of the randomness. We finally sum up giving the composed verifier and thus prove Lemma 2.2.3.

Proof of Lemma 2.2.3:

Let $\epsilon > 0$ be an arbitrary number. Choose $\varepsilon = \epsilon/2$. By claim 6.1.1, there exists a $\delta = \delta_\varepsilon$ such that the statement of Claim 6.1.1 holds. Choose $\gamma = \varepsilon\delta^{2p}$. For this choice of γ, we shall show that

$$\text{MIP}_{1,\gamma}[p, r, a] \subseteq \text{PCP}_{1,\frac{1}{2}+\epsilon}[r + O(2^{pa}), p + 7]$$

thus, proving Lemma 2.2.3.

Let $L \in \text{MIP}_{1,\gamma}[p, r, a]$. Let V_{out} be the corresponding MIP verifier for L. The action of V_{out} is as described below.

V_{out} interacts with p provers, $\Pi_1, \ldots, \Pi_p$. On an input string x of length n, V_{out} picks a $r(n)$-bit random string R; generates p queries $(1, q_1^{(R)}), \ldots, (p, q_p^{(R)})$ and a linear sized circuit C_R. It then issues query $(i, q_i^{(R)})$ to prover Π_i which responds with the answer $a_{i,q_i^{(R)}}$. V_{out} accepts iff $C_R(a_{1,q_1^{(R)}}, \ldots, a_{p,q_p^{(R)}}) = -1$.

Let Q be the set of all queries issued by V_{out} on input string x and over all random strings R. (Notice that $|Q| \leq p2^r$ since each random string R uniquely determines the query V_{out} issues to prover Π_i) The p provers $\Pi_1, \ldots, \Pi_p$ that V_{out} interacts with can be thought of as p functions $\Pi_i : Q \to \{0,1\}^a$.

We shall now construct a $(r + O(2^{pa}), p + 7)$-restricted verifier V_{comp} for L by composing V_{out} with the inner verifier V_{inner} specified in Section 6.1.2. The proof (or oracle) that V_{comp} expects is of the form $\Gamma : \{0,1\}^* \to \{+1,-1\}$.

${V_{\text{comp}}}^{\Gamma}(x)$

1. Pick a random string $R \in \{0,1\}^{r(n)}$.
2. Generate queries $(1, q_1^{(R)}), \ldots, (p, q_p^{(R)})$ and circuit C_R as V_{out} would do on input x and random string R.
3. For each $i \in \{1, \ldots, p\}$, set $A_i(\cdot) \leftarrow \Gamma(i, q_i^{(R)}, \cdot)$.
4. Set $B \leftarrow \Gamma(p+1, R, \cdot)$.
5. Set $\mathcal{A} \leftarrow \{+1,-1\}^{a(n)}$.
6. Set $\mathcal{B} \leftarrow \{(a_1, \ldots, a_p) | C_R(a_1, \ldots, a_p) = -1\}$.
7. For each $i \in \{1, \ldots, p\}$, set the projection function $\pi_i : \mathcal{B} \to \mathcal{A}$ such that $(a_1, \ldots, a_p) \xrightarrow{\pi_i} a_i$.
8. Accept iff ${V_{\text{inner}}}^{A_1, \ldots, A_p, B}(\mathcal{A}, \mathcal{B}, \pi_1, \ldots, \pi_p)$ accepts.

Clearly the number of queries issued by V_{comp} is that of V_{inner} which is 7, while the total randomness is the sum of the randomness of V_{out} and V_{inner} which is $r + O(2^{pa})$.

It is easy to verify that V_{comp} has completeness 1. Suppose $x \in L$. By the completeness of V_{out}, there exists tables $\Pi_1, \ldots, \Pi_p$ such that $\Pr_R[{V_{\text{out}}}^{\Pi_1, \ldots, \Pi_p}(x, R) = \text{ accept }] = 1$. For each $R \in \{0,1\}^r$, let $(1, q_{j_1}^{(R)}), \ldots, (p, q_{j_p}^{(R)})$ be the queries issued by V_{out} on input string x and random string R. Construct another oracle $\Pi_{p+1} : \{0,1\}^r \to \{0,1\}^{ap}$ such that $\Pi_{p+1}(R) = (a_{1,q_1^{(R)}}, \ldots, a_{p,q_p^{(R)}})$ where $a_{i,q_i^{(R)}} = \Pi_i(q_i^{(R)})$ (i.e., response of oracle Π_i on query $q_i^{(R)}$). Now if we construct Γ such that

- For each $i \in \{1, \ldots, p\}$, and $q \in Q$, $\Gamma(i, q, \cdot)$ is the long code of $\Pi_i(q)$.
- For each $R \in \{0,1\}^r$, $\Gamma(p+1, R, \cdot)$ is the long code of $\Pi_{p+1}(R)$.

we note that V_{comp} accepts on all random strings. Thus, the completeness is 1.

The only thing that is left to be proved is that the soundness of V_{comp} is $\frac{1}{2} + \epsilon$. We prove this by showing that if V_{comp} accepts x with probability at least $\frac{1}{2} + \epsilon$, i.e.,

$$\Pr_{R'}[V^{\Gamma}(x; R') = \text{ accept }] \geq \frac{1}{2} + \epsilon$$

(where R' is the combined randomness of V_{out} and V_{inner}) then $x \in L$. By the soundness condition of the outer MIP verifier V_{out}, it is sufficient if we show that there exist provers $\Pi_1, \ldots, \Pi_p$ such that

$$\Pr_{R}[V^{\Pi_1, \ldots, \Pi_p}(x; R) = \text{ accept }] \geq \gamma$$

And the rest of the proof would be devoted to proving this fact.

Consider the following randomized strategy DECODE that takes as input a folded table A and returns a a-bit string. A is an oracle whose input are functions of the form $f : \mathcal{A} \rightarrow \{+1, -1\}$. Recall $\mathcal{A} = \{+1, -1\}^a$.

DECODE(A)

1. Choose $\alpha \subseteq \mathcal{A}$ with probability $\hat{A}^2_\alpha$.
2. Choose an $x \in \alpha$ uniformly at random.
3. Return x.

We remark that since $\sum_\alpha \hat{A}^2_\alpha = 1$, $\hat{A}^2_\alpha$ does determine a probability distribution and hence Step 1 is legitimate. Moreover, the procedure will never get stuck in Step 2 because of choosing $\alpha = \phi$ since $\hat{A}_\phi = 0$ (as A is folded) We thus have that if $|\hat{A}_{\{a\}}| \geq \delta$, then $\Pr[\text{DECODE}(A) = a] \geq \delta^2$.

Now imagine constructing the p provers $\Pi_1, \ldots, \Pi_p$ using the randomized strategy DECODE (on the proofs Γ of the composed verifier V_{comp}) as follows:

For each $i \in \{1, \ldots, p\}$ do

For each $q \in Q$ do

Set $a_{i,q} \leftarrow \text{DECODE}(\Gamma(i, q, \cdot)$.

Set prover $\Pi_i : Q \rightarrow \{0,1\}^a$ such that $\Pi_i(q) = a_{i,q}, \forall q \in Q$.

We shall now show that if V_{comp} accepts x on proof Γ with probability at least $\frac{1}{2}+\epsilon$, then V_{out} accepts x on interacting with the p provers $\Pi_1, \ldots, \Pi_p$ as constructed above with probability at least γ (over the random coin tosses of V_{out} and the DECODE strategy.)

Let $\mathcal{R}$ denote the set of random choices of the MIP verifier V_{out} that satisfy

$$\Pr_{R''}[{V_{\text{inner}}}^{A_1,\ldots,A_p,B}(x; R'') = \text{ accept }] \geq \frac{1}{2} + \frac{\epsilon}{2}$$

where the probability is taken over the coin tosses R'' of V_{inner} and each of $A_i(\cdot) = \Gamma(i, q_i^{(R)}, \cdot)$ and $B = \Gamma(p+1, R, \cdot)$ as specified in the working of V_{comp}. By an averaging argument, it follows that $\Pr_R[R \in \mathcal{R}] \geq \epsilon/2$. Let $\varepsilon = \epsilon/2$ and $\delta = \delta_\varepsilon$ as mentioned in the beginning of the proof. By the soundness condition for the inner verifier V_{inner} (see Claim 6.1.1), we have that for each $R \in \mathcal{R}$, there exist $a_1^{(R)}, \ldots, a_p^{(R)}$ such that $C_R(a_1^{(R)}, \ldots, a_p^{(R)}) = -1$ and for each $l \in \{1, \ldots, p\}$, $|\hat{A}_{i,\{a_l^{(R)}\}}| \geq \delta$. Translating these conditions into the proof of the composed verifier V_{comp}, we have that for each $R \in \mathcal{R}$, there exist $a_1^{(R)}, \ldots, a_p^{(R)}$ such that $C_R(a_1^{(R)}, \ldots, a_p^{(R)}) = -1$ and for each $l \in \{1, \ldots, p\}$, $|(\hat{\Gamma}(i, q_i^{(R)}, \cdot)_{\{a_l^{(R)}\}}| \geq \delta$. We now use these facts to produce p provers $\Pi_1, \ldots, \Pi_p$ for V_{out} such that V_{out} accepts these p provers with probability at least γ.

Reiterating the soundness condition from the inner verifier V_{inner}, we have that for each $R \in \mathcal{R}$, there exist $a_1^{(R)}, \ldots, a_p^{(R)}$ such that $C_R(a_1^{(R)}, \ldots, a_p^{(R)}) = -1$ and for each $l \in \{1, \ldots, p\}$, $|(\hat{\Gamma}(i, q_i^{(R)}, \cdot))_{\{a_l^{(R)}\}}| \geq \delta$. Now, let us analyze the probability of the outer verifier accepting the provers $\Pi_1, \ldots, \Pi_p$ on input string x, where the provers Π_i are constructed from Γ as mentioned before.

$$
\begin{aligned}
\Pr\left[{V_{\text{out}}}^{\Pi_1,\ldots,\Pi_p}(x;R) = \text{ accept }\right] &= \Pr\left[C_r(a_{1,q_1^{(R)}}, \ldots, a_{p,q_p^{(R)}}) = -1\right] \\
&\geq \Pr\left[\forall i, \Pi_i(q_i^{(R)}) = a_i^{(R)}\right] \\
&\geq \Pr_R[R \in \mathcal{R}] \cdot \Pr\left[\forall i, \Pi_i(q_i^{(R)}) = a_i^{(R)} \middle| R \in \mathcal{R}\right] \\
&= \Pr_R[R \in \mathcal{R}] \cdot \Pr\left[\forall i, \textsc{Decode}\left(\Gamma(i, q_i^{(R)}, \cdot)\right) = a_i^{(R)} \middle| R \in \mathcal{R}\right] \\
&= \Pr_R[R \in \mathcal{R}] \cdot \prod_{i=1}^{p} \Pr\left[\textsc{Decode}\left(\Gamma(i, q_i^{(R)}, \cdot)\right) = a_i^{(R)} \middle| R \in \mathcal{R}\right] \\
&\geq \varepsilon\delta^{2p} \\
&= \gamma
\end{aligned}
$$

(all the probabilities are over the random coins of both V_{out} and the DECODE procedure unless otherwise specified.) Thus, there exists provers $\Pi_1, \ldots, \Pi_p$ such that V_{out} accepts with probability at least γ, which in turn implies that $x \in L$. This completes the proof of the Lemma 2.2.3 $\square$

CHAPTER 7

Conclusion

We considered the problem of finding small PCPs with low query complexity. Both the parameters - proof-size and query complexity have been independently optimized. In this thesis, we considered whether we can have PCPs which have both low query complexity and small proof-size. We demonstrated that for every language in NP there exists a PCP in which there is at most a slightly-super-cubic blowup in the proof-size and with a query complexity as low as 16. In this process, we construct several modules that are amenable to future PCP constructions.

As a starting step, we proved the hardness of the *Polynomial Constraint Satisfaction* problem. This is a neat algebraic problem and easily lends itself to MIP constructions. In the next step, we use the state-of-the-art Low Degree Tests [25] in conjunction with the hardness of the Polynomial constraint satisfaction to obtain a 3-prover MIP for SAT. For this part, we follow a proof of [1] (their parallelization step); however a direct implementation would involve $6 \log n$ randomness, or an n^6 blow up in the size of the proof. Part of this is a cubic blow up due to the use of the low-degree test and we are unable to get around this part. Direct use of the parallelization also results in a quadratic blowup of the resulting proof. We saved on this by creating a variant of the parallelization step of [1] that uses higher dimensional varieties instead of 1-dimensional ones. We then finally truncate the recursion by providing a constant bit verifier. This is the first time that such a constant bit-verifier has been constructed for non-canonical MIPs with more than 2 provers.

7.1 Scope for Further Improvements

It is open as to whether there exist nearly linear sized PCPs with query complexity of 3 for NP statements. Also, no non-trivial limitations are known for the joint query-proofsize complexity of

PCPs.

With respect to our PCP construction, the following are a few approaches which would further reduce the size-query complexity.

1. An improved low-error analysis of the low-degree test of Rubinfeld and Sudan [26] in the case when the field size is linear in the degree of the polynomial. (It is to be noted that the current best analysis [3] requires the field size to be at least a fourth power of the degree.) Such an analysis would reduce the proof blowup to nearly quadratic.

2. Converting the PCP of Håstad [19] into an inner verifier for p-prover MIPs and thus showing that for every $\delta > 0$ and p there exists $\epsilon > 0$ and c such that

$$\mathrm{MIP}_{1,\epsilon}[p, r, a] \subseteq \mathrm{PCP}_{1-\delta,\frac{1}{2}}[r + c \log a, p + 3].$$

   This would reduce the query complexity of the small PCPs constructed in this paper to 6 bits.

Bibliography

[1] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, May 1998.

[2] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, January 1998.

[3] Sanjeev Arora and Madhu Sudan. Improved low degree testing and its applications. In *Proc. 29th ACM Symp. on Theory of Computing*, pages 485–495, El Paso, Texas, 4–6 May 1997.

[4] László Babai. Trading group theory for randomness. In *Proc. 17th ACM Symp. on Theory of Computing*, pages 421–429, Providence, Rhode Island, 6–8 May 1985.

[5] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proc. 23rd ACM Symp. on Theory of Computing*, pages 21–31, New Orleans, Louisiana, 6–8 May 1991.

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

[7] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability—towards tight results. *SIAM Journal of Computing*, 27(3):804–915, June 1998.

[8] Mihir Bellare, Shafi Goldwasser, Carsten Lund, and Alexander Russell. Efficient probabilistically checkable proofs and applications to approximation. In *Proc. 25th ACM Symp. on Theory of Computing*, pages 294–304, San Diego, California, 16–18 May 1993.

[9] Mihir Bellare and Madhu Sudan. Improved non-approximability results. In *Proc. 26th ACM Symp. on Theory of Computing*, pages 184–193, Montréal, Québec, Canada, 23–25 May 1994.

[10] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *Proc. 20th ACM Symp. on Theory of Computing*, pages 113–131, Chicago, Illinois, 2–4 May 1988.

[11] Stephen A. Cook. Short propositional formulas represent nondeterministic computations. *Information Processing Letters*, 26(5):269–270, 11 January 1988.

[12] Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, March 1996.

[13] Uriel Feige and Joe Kilian. Impossibility results for recycling random bits in two-prover proof systems. In *Proc. 27th ACM Symp. on Theory of Computing*, pages 457–468, Las Vegas, Nevada, 29 May–1 June 1995.

[14] Uriel Feige and László Lovász. Two-prover one-round proof systems: Their power and their problems (extended abstract). In *Proc. 24th ACM Symp. on Theory of Computing*, pages 733–744, Victoria, British Columbia, Canada, 4–6 May 1992.

[15] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-prover interactive protocols. *Theoretical Computer Science*, 134(2):545–557, 21 November 1994. Note.

[16] Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *Proc. 3rd Israel Symposium on Theoretical and Computing Systems*, 1995.

[17] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal of Computing*, 18(1):186–208, February 1989.

[18] Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Proc. 37nd IEEE Symp. on Foundations of Comp. Science*, pages 627–636, Burlington, Vermont, 14–16 October 1996.

[19] Johan Håstad. Some optimal inapproximability results. In *Proc. 29th ACM Symp. on Theory of Computing*, pages 1–10, El Paso, Texas, 4–6 May 1997.

[20] Dror Lapidot and Adi Shamir. Fully parallelized multi prover protocols for NEXP-time (extended abstract). In *Proc. 32nd IEEE Symp. on Foundations of Comp. Science*, pages 13–18, San Juan, Puerto Rico, 1–4 October 1991. IEEE.

[21] Frank Thomson Leighton. *Introduction to Parallel Algorithms and Architectures*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1992.

[22] Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. In *Proc. 31st IEEE Symp. on Foundations of Comp. Science*, pages 2–10, St. Louis, Missouri, 22–24 October 1990.

[23] Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *Proc. 26th ACM Symp. on Theory of Computing*, pages 194–203, Montréal, Québec, Canada, 23–25 May 1994.

[24] Ran Raz. A parallel repetition theorem. *SIAM Journal of Computing*, 27(3):763–803, June 1998.

[25] Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proc. 29th ACM Symp. on Theory of Computing*, pages 475–484, El Paso, Texas, 4–6 May 1997.

[26] Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal of Computing*, 25(2):252–271, April 1996.

[27] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, October 1980.

[28] Daniel Spielman. *Computationally Efficient Error-Correcting Codes and Holographic Proofs*. PhD thesis, Massachusetts Institute of Technology, June 1995.