PCP CHARACTERIZATIONS OF NP: TOWARDS A POLYNOMIALLY-SMALL ERROR-PROBABILITY

Irit Dinur, Eldar Fischer, Guy Kindler, Ran Raz, and Shmuel Safra

Abstract. This paper strengthens the low-error PCP characterization of NP, coming closer to the upper limit of the BGLR conjecture. Consider the task of verifying a written proof for the membership of a given input in an NP language. In this paper this is acheived by making a constant number of accesses to the proof, obtaining error probability that is exponentially small in the total number of bits that are read.

We show that the number of bits which are read in each access to the proof can be made as high as $\log^{\beta} n$, for *any* constant $\beta < 1$, where n is the length of the proof. The BGLR conjecture asserts the same for any constant β, for β smaller *or equal* to 1.

Our results are in fact stronger, implying that the Gap-Quadratic-Solvability problem with a constant number of variables in each equation is NP-hard. That is, given a system of n quadratic-equations over a field $\mathcal{F}$ of size up to $2^{\log^{\beta} n}$, where each equation depends on a constant number of variables, it is NP-hard to distinguish between the case where there is a common solution to all of the equations, and the case where any assignment satisfies at most a $\frac{2}{|\mathcal{F}|}$ fraction of them.

At the same time, our proof presents a *direct* construction of a low-degree-test whose error-probability is exponentially small in the number of bits accessed. Such a result was previously known only relying on recursive applications of the entire PCP theorem.

Keywords. NP, PCP, sum-check, consistent-reader, low-degree-extension, representation-procedure

Subject classification. PCP

1. Introduction

Cook-Levin's characterization of NP implies that every $L \in$ NP is reducible to 3-SAT. The reduction from L to 3-SAT is a polynomial-time algorithm that receives an input string I, and produces a set Ψ of boolean functions (called local-tests), each depending on a constant number of variables. Ψ represents the membership of I in L, in the sense that there exists an assignment satisfying all local-tests if and only if $I \in L$.

A PCP characterization of NP differs from Cook-Levin's characterization in regards to what is guaranteed in the case where the input is not in L: In Cook's characterization, one can only be sure that the reduction will produce a system that cannot be entirely satisfied. To characterize NP in terms of PCP, it must be guaranteed that the reduction algorithm produces a system Ψ such that no assignment can satisfy even a small fraction ϵ of its local-tests.

In both cases, a satisfying assignment to Ψ can be viewed as a witness for I's membership in L (and hence Ψ can be viewed as a membership-verification system). In a PCP framework, however, this witness can be efficiently verified by randomly picking a local-test of Ψ and verifying that it holds (hence the term PCP – Probabilistic Checking of Proofs). In this case, the *error probability* parameter, ϵ, of the PCP, bounds the probability of accepting I even though $I \notin L$. Other parameters of Ψ, such as the variable range and the number of variables accessed by each local-test, are also part of the PCP characterization.

For many applications of PCP, the characterization of NP with a constant error-probability and variables of a constant range Arora *et al.* (1998); Arora & Safra (1998) suffices. In order to prove NP-hardness of other problems, however, sub-constant error-probability has turned out to be essential. For example, Lund & Yannakakis (1994) and Bellare *et al.* (1993) were able to prove that approximating SET-COVER to within logarithmic factors is *almost* NP-hard, using the constant error-probability PCP characterization of NP. To improve this result to strict NP-hardness, Bellare *et al.* (1993) had suggested the "sliding scale" conjecture.

The sliding scale conjecture states that it is possible to keep the number of variables accessed by each local-reader constant, and to make the variables' range non-constant, obtaining an error probability polynomially small in the size of the variable-range. In other words, it is possible to achieve a membership verification system for any NP-language where each local-test accesses a constant number of 'words' (variables), and where the error-probability is exponentially small in the 'word-length' (number of bits in each variable).

One cannot expect the error-probability to be less than polynomially small in the size of the variables' range, since a random assignment will satisfy any satisfiable local-test with such a probability (recall that each test depends on a constant number of variables). Hence the sliding scale conjecture is optimal in the sense of error-probability.

According to the conjecture, the variables' range may be increased up to a size polynomial in the length of the original input (note that each local-test can be given as a truth-table). Reaching larger range-size while keeping the error polynomially small in the range would imply sub-exponential algorithms for NP (the error-probability would then become less than $1/|\Psi|$, i.e. zero). In the case where the input is not in L, this implies that no local-test succeeds, so the problem of deciding whether the input is in L reduces to that of deciding whether any of the local-tests is satisfiable.

The sliding scale conjecture was shown to hold for a sizable portion of the applicable range-size in Arora & Sudan (1997); Raz & Safra (1997), where a PCP characterization of NP was shown that achieves error-probability polynomially small in the size of the variable range for a variable range of size up to $2^{\log^{\beta} n}$, where $\beta < 1$ is a certain positive constant.

Our Main Results. In this paper, we prove the sliding scale conjecture for variable range sizes of up to $2^{\log^\beta n}$ where β is *any* constant smaller than one (as opposed to "*some* constant" achieved by Raz & Safra (1997)), thus coming closer to proving the sliding scale conjecture for the full applicable range.

In fact our result is somewhat stronger, proving the conjecture for the aforementioned range using proof verification systems of a specific structure. In these systems the local-tests have the form of quadratic-equations instead of being general boolean valued functions, with the variables' range representing a finite field. This result implies that for a quadratic equation-system of n equations over a field $\mathcal{F}$ (with $|\mathcal{F}| \approx 2^{\log^\beta n}$ for any fixed constant $\beta < 1$), where each equation depends on a constant number of variables, it is NP-hard to decide whether there exists a common solution to all equations, or whether any assignment to the variables satisfies no more than a $\frac{2}{|\mathcal{F}|}$ fraction of them.

One of the main tools used to obtain the above result, which is interesting in its own right, is that of a low-degree function reader, LDF-reader for short. This is a version of what is known in the literature as a low-degree test (see Arora & Sudan (1997); Raz & Safra (1997)). A direct construction of an LDF-reader is shown herein, that achieves an exponentially-small error-probability with respect to the number of bits it accesses. Such LDF-readers could previously be attained only by recursive applications of the entire PCP theorem.

Related Results. We note that there is no known PCP characterization of NP, where the size of the variable-range is polynomial in the size of the membership-verification system (or equivalently, the length of each variable is logarithmic in it), and the error probability is exponentially small in the number of accessed bits. This is true even when allowing a super-polynomial time reduction. The repetition lemma of Raz (1998) shows that by two accesses to $\Theta(\log n)$ bits, the error-probability can be made polynomially small in n, where n is the size of the original input, while the size of the generated system is $n^{\log n}$. Similarly, the multi-linear extension of Babai *et al.* (1991) yields a system with a $\frac{1}{n}$ error-probability, whose size is $n^{\log n}$. In fact, in any known reduction there is always a factor of at least $\log^\epsilon n$ in the exponent that separates the error-probability from the size of the generated instance.

Achieving an error-probability polynomially small in the size of the generated instance is an important open problem. Such a characterization of NP would improve hardness results for several problems. For example, approximating the 'Monotone-Minimum-Satisfying-Assignment' problem (which is closely related to approximating the length of propositional proofs Alekhnovich *et al.* (1998)) has been shown to be NP-hard in Dinur & Safra (1998) via a reduction from PCP, such that the hardness of approximation ratio is preserved. Hence a polynomially small error-probability PCP characterization of NP would immediately imply that it is NP-hard to approximate the length of propositional proofs to within an n^ϵ factor for some constant $\epsilon > 0$.

Raz & Safra (1997) managed to keep the exponential relation between the number of bits accessed and the error-probability, thus showing the sliding scale conjecture true for a variable range of size up to $2^{\log^\beta n}$ for *some* constant $\beta < 1$. For larger β (any constant $\beta < 1$) Raz & Safra (1997) showed a system whose error probability is $2^{-\log^\beta n}$, yet without the exponential relation between the number of accessed bits and the error-probability, since the number of bits accessed was $O(\log^{\beta} n \cdot \mathrm{poly}\log\log n)$. This factor of $\mathrm{poly}\log\log n$ is significant when viewing, for example, the result in terms of Gap-Quadratic-Solvability. The result of Raz & Safra (1997), if it were to be translated to Gap-Quadratic-Solvability terms, would at best give an equation system with each equation depending on $O(\mathrm{poly}\log\log n)$ variables. In comparison, our result translates to a quadratic equation-system with the same error-probability, but where every equation depends on a *constant* number of variables, namely $\Theta(\frac{1}{(1-\beta)^2})$.

Techniques. We use the general framework of Arora *et al.* (1998); Arora & Safra (1998); Raz & Safra (1997) for our proof. However, instead of the generalized form of the composition paradigm utilized in previous PCP proofs, we use a more concrete representation. Our result could have been obtained using the previous structure, but this representation simplifies our proof, and some of its techniques may be of independent interest.

In Hastad *et al.* (1993), it was shown that given a system of quadratic equations over a finite field, it is NP-hard to distinguish between the case that the system can be completely satisfied, and the case that not even a small fraction of the equations can be satisfied by a single assignment. The crucial difference between this and our main result is that in the Hastad *et al.* (1993) reduction each equation depends on almost all the variables in the system, while our main result claims the same for the case where the equations are restricted to having a constant number of variables each.

Our proof begins with a system Ψ of quadratic equations as in Hastad *et al.* (1993), and reduces it to a system Ψ' of quadratic equations with a constant number of variables in each. The key property of our proof is that throughout the reduction we use systems of equations over the same field $\mathcal{F}$, the field over which Ψ is defined. The field structure is utilized through various steps of composition, thus enabling us to cross the barrier that limits the proof technique of Raz & Safra (1997).

To simplify the exposition, the reduction partitions the variables of Ψ' into subsets called domains. In each such domain a mapping is defined, associating each variable with a point in a linear space of the form $\mathcal{F}^d$ over $\mathcal{F}$. An assignment to these variables can thus be regarded as a function over the linear space.

The reduction has two main steps. At first, it transforms Ψ into a system Ψ_{sc} where the number of variables in each equation is constant. This is accomplished by an iterative application of the sum-check technique from Babai *et al.* (1991). The system Ψ_{sc} has the required properties only if the assignment to the variables in each domain, when viewed as a function, is a low-degree polynomial. In order to get rid of this restriction, the reduction then generates LDF-readers and plugs them into the equations of Ψ_{sc}, thereby obtaining the final system Ψ'.

LDF-readers. LDF-readers are used to obtain evaluations of polynomial functions of low-degree that are represented by a set of variables, by accessing only a very small part of their representation. An LDF-reader should either reject or return values that are consistent with some low-degree polynomial, even if the assignment to the representation variables is not totally consistent with the representation of one polynomial. The probability that, given an incorrect representation, the LDF-reader does not reject but still the returned evaluations are not consistent with a low-degree polynomial, is its error probability. For a more accurate definition of an LDF-reader, the reader is referred to Section Section 2.

An LDF-reader of sub-constant error-probability seems necessary in order to attain PCP characterizations of NP with sub-constant error-probability. The plane-vs.-plane LDF-reader introduced by Raz & Safra (1997), where a polynomial is represented by its restriction to planes, achieves a sub-constant probability. The previously used line-vs.-point LDF-reader was shown by Arora & Sudan (1997) to have a small error-probability as well. However, the error probability of these LDF-readers is not exponentially small in the number of bits they access.

It seems to be difficult to achieve error-probability smaller than polynomial in the number of accessed bits, using a direct LDF-reader comparing subspaces (lines, planes, etc.) for consistency. This is since many bits are required to represent the restriction of a polynomial to a subspace. One way to attain exponentially small error-probability from these LDF-readers is by utilizing the composition technique, applying the entire PCP theorem to them. Our proof, in contrast, makes this recursion concrete, utilizing an explicit representation of low-degree polynomial functions that yields LDF-readers with an exponentially small error probability.

The composition-recursion LDF-reader. Our LDF-reader uses a representation of low-degree polynomials as follows. We begin with a representation where a multi-dimensional polynomial is represented by all of its point evaluations, and also by its restriction to certain constant dimensional subspaces. We use a new *power-substitution* technique to then replace each constant dimensional restriction of the polynomial by a multi-dimensional polynomial of a much smaller degree. This is done, roughly, by replacing monomials of high degree with new variables. The polynomials whose degree was reduced are then represented by their point evaluations and their restriction to constant dimensional subspaces, and the process is repeated.

After a constant number of such iterations we obtain polynomials of linear degree over constant dimensional spaces. Each of these polynomials is then represented by a constant number of variables that range over the field $\mathcal{F}$. Hence to obtain evaluations our LDF-reader is not required to completely read a low-degree polynomial over some subspace – instead it only accesses a constant number of variables that range over $\mathcal{F}$.

Organization of the paper. Our main result and the main definitions required for its proof are stated in Section Section 2. The proof of the main result, based on lemmas that are proven in the following sections, appears in Section Section 3. The construction of the LDF-reader that is utilized in the proof of the main result appears in Section Section 4. In particular, the power-substitution technique, used in the construction of the LDF-reader to represent polynomials using other polynomials with more variables but with considerably smaller degrees, is described in Subsection Section 4.4. Finally, Section Section 5 describes the recursive application of the sum-check (and other) techniques, which are used in the reduction to obtain from the original system Ψ a system Ψ_{sc} with a constant number of variables in each equation.

2. Preliminaries

In this section we describe the basic ideas and definitions utilized in the proof of our main result.

Gap-Quadratic-Solvability. The Gap-Quadratic-Solvability problem is that of determining whether all the equations in a given system of quadratic-equations can be simultaneously satisfied, or whether only a small fraction of the equations can be satisfied. Viewing the quadratic equations as local-tests of a PCP system, showing this problem to be NP-hard yields a PCP characterization of NP.

Definition 2.1 (Gap-Quadratic-Solvability). *The Gap-Quadratic-Solvability problem with parameters D, σ and ϵ (which may be, implicitly, functions of the system size n), is denoted by gap-QS$[D, \sigma, \epsilon]$. An instance of the problem is a field $\mathcal{F}$ of size*[*] *σ, and a set of n quadratic-equations over $\mathcal{F}$, where each equation has at most D variables (D is called the dependency parameter). The problem is to distinguish between the following two cases:*

Yes. There is an assignment to the variables that satisfies all of the equations.

No. Every assignment to the variables satisfies at most an ϵ fraction of the equations – ϵ is called the error parameter.

An instance which falls under one of the above criteria is said to have the gap property. Any outcome is acceptable for instances that do not have the gap property.

Our main theorem shows NP-hardness of gap-QS with a constant dependency parameter, for a field of size $\sigma \approx 2^{c \log^\beta n}$ and an error parameter $\epsilon = \frac{2}{\sigma}$, where $\beta < 1$ is any constant smaller than 1 and $c > 0$ is some constant (in fact we first prove the result for $\epsilon = \frac{1}{\sigma^{\Omega(1)}}$ which is polynomially small in the size σ of the field, and amplify the hardness to error $\frac{2}{\sigma}$ by a simple amplification technique, which is introduced and proved in Subsection Section 3.3 and Subsection Section 3.4). We therefore abbreviate gap-QS$[D, \sigma]$ for the gap-QS problem where ϵ is fixed to be $\frac{2}{\sigma}$. Note that this error probability is polynomially small in the size of the field, and therefore exponentially small in the length, measured in bits, of each variable.

Theorem 2.2 (main theorem). *For every constant $\beta < 1$ there exists a constant $c_2 > c_1 > 0$ such that gap-QS$[O(1), \sigma]$ is NP-hard for σ in the range $2^{c_1 \log^\beta n} \le \sigma \le 2^{c_2 \log^\beta n}$, where n is the number of equations in the system.*

We actually prove Theorem Theorem 2.2 via a many-to-one reduction. Informally speaking, this means that gap-QS$[O(1), \sigma]$ is proven to be NP-complete.

[*]In fact there can be at most one field of any given cardinality, however we would like to be able to look at gap-QS problems where the size can vary in some range. In that case it makes sense to request that the actual field be given as part of the input.

Gap-QS$[n, \sigma]$, where the number of variables in each equation is not bounded, is proven to be NP-hard in Hastad *et al.* (1993) for any σ that is polynomially bounded in n, using simple linear codes:

THEOREM 2.3 (Hastad *et al.* 1993). *Let $\sigma(n)$ be any positive poly-time computable function, and suppose that for some constant $\gamma > 0$, $\sigma(n) = O(n^\gamma)$. Then there exist constants $d_2 > d_1 > 0$ such that Gap-QS$[n, \sigma]$ is NP-hard for σ in the range $(\sigma(n))^{d_1} \leq \sigma \leq (\sigma(n))^{d_2}$.*

This theorem is proven by a relatively simple reduction from the Cook-Levin characterization of NP, so the entire difference between this characterization of NP and the PCP characterization boils down to the constant bound on the number of variables that each equation accesses.

Theorem Theorem 2.2 is proven by showing a reduction algorithm, taking as input a system Ψ of n quadratic-equations and producing a new system Ψ' where the number of variables in each equation is bounded by a constant. The number of variables in each equation is reduced while roughly preserving the fraction of satisfiable equations. Specifically, if Ψ is completely satisfiable then Ψ' is completely satisfiable as well; and if no more than a $\frac{2}{|\mathcal{F}|}$ fraction of the equations of Ψ can be satisfied then the same occurs for Ψ'.

2.1. LDFs and Domains. Let us set a notation for polynomial functions of low degree – an object used extensively in this paper.

DEFINITION 2.4 (LDF - low degree function). *An $[r, d]$-LDF is a polynomial function from $\mathcal{F}^d$ to $\mathcal{F}$, of total-degree at most r.*

For the exposition of the reduction algorithm and for the proof of correctness, it is useful to consider certain subsets of the variables as separate *domains*. Each variables ranges over $\mathcal{F}$, and the variables in each such domain are associated with the points of a vector field $\mathcal{F}^d$ over $\mathcal{F}$. Throughout our reduction, the domains are always disjoint, so in the final system Ψ' and also in each intermediate construction, no variable can belong to more than one domain.

DEFINITION 2.5 (domain). *A* domain *F is a set of $|\mathcal{F}|^d$ variables ranging over $\mathcal{F}$, that has one variable $F[x]$ for every point $x \in \mathcal{F}^d$, where $d = d(F)$ is called the* dimension *of the domain. F is said to be* assigned *a function $f : \mathcal{F}^d \to \mathcal{F}$ if for every x, the variable $F[x]$ is assigned $f(x)$.*

Two more parameters are associated with each domain in addition to the dimension – the lower-degree, *denoted $s(F)$, and the* upper-degree *$r(F)$ ($r(F)$ will always be larger or equal to $s(F)$).*

Let us stress that the lower-degree and upper-degree parameters of domains, as well as the whole partition of variables into domains, are only figments of our proof and construction. In the final system Ψ' the domain-structure is discarded, and we are left with variables that range over $\mathcal{F}$. These variables can be assigned every value in $\mathcal{F}$, and therefore a domain F containing these variables can be assigned any function $f : \mathcal{F}^{d(F)} \to \mathcal{F}$.

In the proof we give special consideration to assignments of domains which correspond to LDFs. In particular, it will be shown that if the system Ψ' generated by the reduction is satisfiable, there is a satisfying assignment where every domain F is assigned an $s(F)$-degree LDF. In the no case we need to show that Ψ' cannot be more than $\frac{2}{|\mathcal{F}|}$ satisfiable by *any* assignment. However we prove later that it suffices to only show this for assignments where each domain F is assigned an $r(F)$-degree LDF.

Definition 2.6 (assignment of a domain). *The assignment f of a domain F is said to be* good*, if f is an $[s(F), d(F)]$-LDF, and it is said to be* feasible *if f is an $[r(F), d(F)]$-LDF. An assignment of a set of variables containing one or more domains is said to be good (feasible) if the assignment to each domain is good (feasible).*

The reduction which transforms Ψ into Ψ' goes through an intermediate system Ψ_{sc} where the number of variables in each equation is constant, but which does not yet have the desired properties. In particular, Ψ_{sc} might be completely satisfiable even when there exists no assignment satisfying more than a $\frac{2}{|\mathcal{F}|}$ fraction of the equations of Ψ. However, Ψ_{sc} behaves much better if we restrict the set of assignments considered: On one hand if Ψ is completely satisfiable then not only Ψ_{sc} is completely satisfiable, but there exists a good satisfying assignment for it. On the other hand, if Ψ is no more than $\frac{2}{|\mathcal{F}|}$-satisfiable, then there is no good or even feasible assignment for Ψ_{sc} satisfying more than a $\frac{2}{|\mathcal{F}|}$ fraction of its equations.

2.2. Defining LDF-Readers. To transform Ψ_{sc} into the final system Ψ', we should prevent it from being satisfiable by assignments where domains are not assigned LDFs. In fact what we manage is a bit weaker (although it suffices). Consider an equation $\psi \in \Psi_{sc}$ that has the variables $F[x_1], \ldots, F[x_k]$ in a domain F. To prevent it from being satisfiable by unwanted assignments we use a mechanism called an *LDF-reader*, which is plugged into ψ in place of these variables. The LDF-reader ensures that ψ either reads evaluations at $(x_1, \ldots, x_k)$ of an LDF (even if the assignment to F is not feasible) or it is not satisfied.

Example. Fix an assignment to the variables in F, and suppose $k = 1$ and we want to read the value at a point $x \in \mathcal{F}^d$. Ideally, the LDF reader should either output $F[x]$, or reject if the assignment for F is not a low degree function (of degree at most $r(F)$). For example, in the case where $r(F) = 1$ this can be roughly achieved as follows. Pass a random line through x, and read the value of $F[x]$, $F[y]$, and $F[z]$, where y and z are random points on the chosen line. Reject if the values read are not consistent with a linear function, or otherwise return $F[x]$. It is possible to show that almost always the LDF-reader either rejects or outputs a value from an LDF somewhat correlated with the assignment for F. In this example we only used variables from the domain F, however in the general case auxiliary variables may be needed.

LDF readers: formal definition. An LDF-reader evaluating the tuple of points $(x_1, \ldots, x_k)$ in a domain F has two parts – the *representation*, and the set of *local-readers* which produce the evaluations.

The representation. The representation is a set V that contains F and maybe other variables and domains. The variables in V, including those associated with domains in it, are called *representation variables.* The LDF-reader uses the representation variables to produce evaluations. Every good assignment f for F (namely one consistent with an $s(F)$-degree polynomial) must be extendible to a good assignment for all the variables in V, called the *encoding-assignment* of f. When given the encoding-assignment of a good LDF f, the LDF-reader will always return the evaluations of f.

The local-readers. The evaluations at $(x_1, \ldots, x_k)$ are produced by a set of local-readers, where each local-reader accesses only a constant number of representation variables – this property is essential since the local-readers are plugged into Ψ_{sc} to produce Ψ' and the number of variables in each equation must remain constant. Each local-reader may either produce evaluations, or it may reject if it finds that the assignment is not an encoding-assignment.

Local-tests and evaluators. Each local-reader is a pair containing a *local-test* – a conjunction of linear equations over representation variables, and a tuple of k *evaluators.* Each evaluator is a linear-combination of representation variables. A local-reader is said to *accept* an assignment for the representation variables if the local-test is satisfied by it, and otherwise it is said to *reject* it. For an assignment $\mathcal{A}$ which is an encoding-assignment of a good LDF f, it is required that all local-readers accept, and also that the i'th evaluator in each local-reader evaluates to $f(x_i)$.

In case the representation variables are not given a correct encoding-assignment, we would like the local-readers to always reject. This is not possible to achieve, however, with local-readers that access a constant number of representation variables, not even if we allow a small fraction of the local-readers to falsely accept. It is in fact not even possible to ensure that local-readers which do not reject return evaluations of a single LDF. What we can achieve (and turns out to be enough), is that apart from a small fraction, the local-readers either reject or return evaluations of one of a short list of LDFs. This is the list of LDFs which are *permissible* with respect to the assignment of F.

Definition 2.7 (permissible assignment). *An $[r(F), d(F)]$-LDF f is said to be ρ-permissible with respect to an assignment of F if for at least a ρ-fraction of the points x, $F[x]$ is assigned $f(x)$.*

We show later that for a wide range of permissibility parameters ρ, the list of permissible LDFs is bounded by $O(\rho^{-1})$. Since the list is only determined by the assignment to F and is independent of the rest of the representation variables, it will be the same for all LDF-readers evaluating tuples in F. This means that all equations that have variables in F will read evaluations that are consistent with one of the LDFs on the relatively short list.

We now give the formal definition of the parameters of an LDF-reader.

Definition 2.8 ((ρ, ϵ)-LDF-reader). *Let $\mathcal{R}$ be an LDF-reader evaluating a tuple $(x_1, \dots, x_k)$ in a domain F, and fix an assignment to its representation-variables. A local-reader L is said to be* ρ-erroneous *if it accepts,* **and** *there exists no ρ-permissible LDF f (with respect to the assignment of F), such that for all i the i'th evaluator evaluates to $f(x_i)$.*

$\mathcal{R}$ is said to be a (ρ, ϵ)-LDF-Reader, if for any assignment to the representation-variables, the fraction of ρ-erroneous local-readers is at most ϵ.

3. Proof of the Main Theorem

In this section we prove Theorem Theorem 2.2. We show for any constant $\beta < 1$, a polynomial time reduction from the problem Gap-QS$[n, \sigma]$ with $\sigma = 2^{\Theta(\log^\beta n)}$, to the problem Gap-QS$[O(1), \sigma]$.

The reduction starts with a given system Ψ of n quadratic equations over a field $\mathcal{F}$ of size $2^{d_1 \log^\beta n} \leq |\mathcal{F}| \leq 2^{d_2 \log^\beta n}$, d_1, d_2 constants, with up to n variables in each equation. It then generates, in time polynomial in n, a system Ψ' *over the same field* with m equations and at most a constant number of variables in each equation. Since the reduction takes polynomial time (and since the number of equations in not decreased by the reduction), m is polynomially equivalent to n, and therefore the size of the field satisfies $2^{c_1 \log^\beta m} \leq |\mathcal{F}| \leq 2^{c_2 \log^\beta m}$ for appropriate constants $c_2 > c_1 > 0$, as required by Theorem Theorem 2.2.

Ψ' will have the *completeness* property, that is if the given system Ψ can be completely satisfied then Ψ' will be completely satisfiable as well; and the *soundness* property – if Ψ is no more than $\frac{2}{|\mathcal{F}|}$-satisfiable (namely no assignment can satisfy more than a $\frac{2}{|\mathcal{F}|}$ fraction of its equations), then Ψ' is at most $\frac{2}{|\mathcal{F}|}$-satisfiable as well.

The reduction begins by transforming Ψ into a system Ψ_{sc} of quadratic-equations where the number of variables in each equation is bounded by a constant. The variables of Ψ_{sc} are partitioned into domains, and it has the desired properties only with respect to feasible assignments (note that the variables of an equation in Ψ_{sc} can come from several domains). The transformation of Ψ into Ψ_{sc} is done by the *sum-check* algorithm, which consists of an iterative application of the sum-check technique from Babai *et al.* (1991). The properties of the sum-check algorithm are stated in the following lemma, and proven in Section Section 5.

Lemma 3.1 (sum-check). *There exists a polynomial-time algorithm as follows. It takes as input a system Ψ of n quadratic equations over a field $\mathcal{F}$, $|\mathcal{F}| = 2^{\log^\beta n}$, where there are up to n variables in each equation. Given Ψ, the algorithm generates a system Ψ_{sc} of quadratic-equations over $\mathcal{F}$ where each equation has a constant number of variables, and that has the following properties:*

- *Completeness: If Ψ is completely satisfiable then Ψ_{sc} is completely satisfiable by a good assignment.*
- *Soundness: If Ψ is no more than $\frac{2}{|\mathcal{F}|}$-satisfiable then Ψ_{sc} cannot be more than $\frac{2}{|\mathcal{F}|}$-satisfied by a feasible assignment.*

Moreover, all the domains of Ψ_{sc} have the same dimension $d = \Theta(\log^{1-\beta} n)$, lower-degree s, and upper-degree r, where $s \leq |\mathcal{F}|^{c_1}$ and $r \geq |\mathcal{F}|^{c_2}$ for some global constants $c_1 < c_2 < 1$.

The next main steps of the reduction of Ψ_{sc} into the final system Ψ', which are described in detail in the following subsections, are as follows. The reduction generates LDF-readers and plugs them into Ψ_{sc}. For each equation ψ of Ψ_{sc}, that has the variables $F(x_1), \ldots, F(x_k)$ of a domain F, it generates an LDF-reader evaluating $(x_1, \ldots, x_k)$ in F. To plug the LDF-reader into ψ, many copies of ψ are made, and one of the local-readers is plugged into each copy.

The local-tests are added in conjunction with each copy, and hence a system of *conjunctions* is formed. In addition, some of the gap is lost when the LDF-readers are plugged in – if Ψ is no more than $\frac{2}{|\mathcal{F}|}$-satisfiable, the fraction of satisfiable conjunctions in the system obtained from Ψ_{sc} might be somewhat higher. A simple amplification technique is hence applied to the system of conjunctions to avoid that, and then each conjunction is replaced by equations, obtaining Ψ'.

3.1. Generating the LDF-Readers. To generate LDF-readers we use a *constructor* algorithm, as defined below.

Definition 3.2 (constructor). *A constructor is an algorithm that takes as input a domain F and a k-tuple $(x_1, \ldots, x_k)$ of points in $\mathcal{F}^{d(F)}$, where k is a constant*[†]*. It generates an LDF-reader evaluating $(x_1, \ldots, x_k)$ in F, i.e. it generates representation variables and all local readers. It must run in time polynomial in $|\mathcal{F}|^{d(F)}$. Also, the number of variables appearing in each local-reader must be bounded by a constant, and so should be the number of equations in the local-test of each local-reader. In addition, the number of local-readers must only depend on the parameters of F.*

Our reduction uses the *Composition-Recursion LDF-reader constructor*, whose properties are stated in the next lemma, to generate LDF-readers. The proof of the lemma appears in Section Section 4.

Lemma 3.3 (Composition-Recursion LDF-reader constructor). *There exists a global constant c_g, $0 < c_g \leq 1/2$, such that for every $c_1 < c_2 < 1$ and $\beta < 1$ the following holds. There exist a constant $c > 0$, and an LDF-Reader constructor for domains of dimension $d = \Theta(\log^{1-\beta} n)$, lower-degree $s \leq |\mathcal{F}|^{c_1}$, and upper-degree $r \geq |\mathcal{F}|^{c_2}$ (the algorithm runs independently of s and r). The LDF-readers generated by the algorithm are $(\rho, O(\rho^c))$-LDF-readers, for all ρ's which satisfy $\rho > (r/|\mathcal{F}|)^{c_g} d$.*

Before the LDF-readers are actually generated, we make some small technical alterations to Ψ_{sc} as follows.

[†]By saying that k is constant, we mean that other parameters and properties of the constructor may depend arbitrarily on k. In addition, any parameter of the constructor that depends only on k is considered constant as well

Uniformization. The number of variables in each equation of Ψ_{sc} is bounded by some constant k. This implies that an equation in Ψ_{sc} may have variables from up to k distinct domains, and that the number of variables it has from each domain is bounded by k. Before generating LDF-readers, let us assume for simplicity that each equation of Ψ_{sc} has variables from *exactly* $k > 1$ distinct domains, and that it has exactly k variables from each domain. This requires the reduction to add arbitrary variables to the equations, multiplied by zero coefficients.

LDF-Reader generation. After the uniformization, the reduction generates the LDF-readers as described above – For each equation ψ of Ψ_{sc}, that has the variables $F(x_1), \ldots, F(x_k)$ in a domain F, it generates an LDF-reader evaluating $(x_1, \ldots, x_k)$ in F (this takes polynomial time in the size of Ψ_{sc}). Note that since all domains in Ψ_{sc} have the same parameters (dimension d, lower-degree s and upper-degree r, as stated in Lemma Lemma 3.1), the number of local-readers in each LDF-reader is the same as well.

The representation variables of the LDF-readers are added to the variables of the system, and the local-readers are plugged into the equations of Ψ_{sc} as described below.

3.2. Plugging LDF-Readers In. For each equation $\psi \in \Psi_{sc}$ there are now k associated LDF-readers – one for each domain it has variables from. The first step in plugging the LDF-readers into Ψ_{sc} is to replace each such equation ψ by a set $\mathcal{E}_\psi$, containing *conjunctions* of quadratic equations that are obtained by plugging local-readers into ψ. $\mathcal{E}_\psi$ represents ψ in the sense that an assignment satisfying a large enough fraction of the conjunctions in $\mathcal{E}_\psi$ implies a satisfying assignment for ψ, as shown in the proof of Claim Claim 3.4 below.

Generating $\mathcal{E}_\psi$. Let $\psi \in \Psi_{sc}$ be an equation that has variables from the domains $F_1, \ldots, F_k$. For each j, let us denote the variables of ψ in F_j by $F_j[x_1^j], \ldots, F_j[x_k^j]$. ψ is therefore associated with k LDF-readers $\mathcal{R}_1, \ldots, \mathcal{R}_k$, where $\mathcal{R}_j$ evaluates the tuple $(x_1^j, \ldots, x_k^j)$ in F_j. The reduction generates one conjunction in $\mathcal{E}_\psi$ for each choice of k local-readers $L_1, \ldots, L_k$, where $L_j \in \mathcal{R}_j$. The first equation in each such conjunction, denoted ψ', is the quadratic equation obtained from ψ by replacing each variable of the form $F_j[x_i^j]$ with the i'th evaluator of L_j (it evaluates x_i^j in F_j). ψ' is then put in conjunction with the local-tests of the local-readers $L_1, \ldots, L_k$.

The system ${\Psi_{sc}}'$. Note that the number of conjunctions in $\mathcal{E}_\psi$ is the same for every $\psi \in \Psi_{sc}$ – it is $|\mathcal{R}|^k$, where $|\mathcal{R}|$ denotes the number of local-readers in each of the LDF-readers we have generated. We set ${\Psi_{sc}}'$ to be the union of all the sets $\mathcal{E}_\psi$. Since the number of variables in each local-reader is constant, the number of variables in each conjunction of ${\Psi_{sc}}'$ is bounded by a constant as well. The system of conjunctions ${\Psi_{sc}}'$ obviously retains the completeness property of Ψ_{sc}. As the next claim shows, it also retains *some* of its soundness property, even with respect to assignments which are not necessarily feasible.

Claim 3.4. *There exists a constant α, $0 < \alpha < 1$, such that ${\Psi_{sc}}'$ has the following properties:*

- *Completeness: If Ψ is completely satisfiable, then ${\Psi_{sc}}'$ is completely satisfiable as well.*
- *Weakened Soundness: If Ψ is at most $2/|\mathcal{F}|$-satisfiable, then ${\Psi_{sc}}'$ is at most $|\mathcal{F}|^{-\alpha}$-satisfiable (by any assignment).*

To prove the claim we need the following proposition, showing that there cannot be many permissible LDFs for a domain – this implies that most local-readers in an LDF-reader will either reject or return the evaluation of one of a *short list* of permissible LDFs. This proposition appears in Section Section 4 as Claim Claim 4.14, and is proven there.

PROPOSITION 3.5. *Let F be a domain, and let $\rho > \left(\frac{r(F)}{|\mathcal{F}|}\right)^{c_g} d(F)$ where c_g is the same constant as in Lemma Lemma 3.3. Then for any assignment to F there can be at most $2\rho^{-1}$ ρ-permissible LDFs in all.*

Proof of Claim Claim 3.4:

Completeness. If Ψ is satisfiable, then there is a good assignment satisfying Ψ_{sc}. For each of the constructed LDF-readers, extend the assignment to its representation using the encoding-assignment of the associated domain. The extended assignment satisfies ${\Psi_{sc}}'$: A conjunction in ${\Psi_{sc}}'$ contains local-tests, which are all satisfied by encoding-assignments, and an equation ψ'. ψ' was generated from an equation $\psi \in \Psi_{sc}$ by replacing variables with evaluators. But for encoding-assignments, the evaluators and the replaced variables have the same values. Hence since ψ is satisfied, ψ' is satisfied as well.

Weakened soundness. Fix an assignment $\mathcal{A}$ for ${\Psi_{sc}}'$, and let γ be the fraction of conjunctions it satisfies. For an appropriate α, we will show that if $\gamma > |\mathcal{F}|^{-\alpha}$ then there exists a feasible assignment for Ψ_{sc} satisfying more than a $\frac{2}{|\mathcal{F}|}$ fraction of its equation. This implies that Ψ is more than $\frac{2}{|\mathcal{F}|}$-satisfiable – a contradiction.

We denote $a \doteq (1-c_2)c_g/k$, where c_2 is the global constant mentioned in the Sum-Check Lemma (Lemma Lemma 3.1). Letting $\rho \doteq |\mathcal{F}|^{-a}$, it follows by the choice of a (and since $k > 1$) that $\rho > (r/|\mathcal{F}|)^{c_g} d$. Therefore by Lemma Lemma 3.3 we have that the LDF-readers have parameters (ρ, ϵ), where $\epsilon = O(\rho^c)$ and c is as mentioned in the lemma.

An equation $\psi \in \Psi_{sc}$ such that the fraction of satisfied conjunctions in $\mathcal{E}_\psi$ is higher than $k\epsilon$, is said to be *potentially satisfiable.* Since the sets $\mathcal{E}_\psi$ are all of the same size, it follows that the fraction of potentially satisfiable equations is at least $\gamma - k\epsilon$.

Consider a potentially satisfiable equation ψ. $\mathcal{E}_\psi$ was generated from ψ by plugging in k LDF-readers $\mathcal{R}_1, \ldots, \mathcal{R}_k$, evaluating tuples in k domains $F_1, \ldots, F_k$ respectively. A conjunction in $\mathcal{E}_\psi$ is defined by choosing a local-reader L_j out of each LDF-reader $\mathcal{R}_j$. For every j, the fraction of conjunctions in $\mathcal{E}_\psi$ where L_j is ρ-erroneous is bounded by ϵ, as implied by the parameters of the LDF-readers, and hence the fraction of conjunctions where *any* of the readers are erroneous is bounded by $k\epsilon$.

It follows that there exists a satisfied conjunction in $\mathcal{E}_\psi$ in which no local-reader is erroneous, namely the evaluator of each local-reader L_j gives the evaluations of a ρ-permissible

LDF f_j with respect to the assignment of F_j. Hence if each domain F_j were re-assigned the function f_j, ψ would be satisfied.

So far we have shown that the potentially satisfiable equations, which make at least a $\gamma - k\epsilon$ fraction of the equations $\psi \in \Psi_{sc}$, can be satisfied by re-assigning the domains with ρ-permissible LDFs. For each domain F in Ψ_{sc}, choose a *random* ρ-permissible LDF, or the zero LDF if no such LDF exists, and re-assign it to F. We have obtained a *feasible assignment* for Ψ_{sc}. We compute the chance of a potentially satisfiable equation to be satisfied by the new assignment.

There are at most $O(\rho^{-1})$ ρ-permissible LDFs for each domain by Proposition Proposition 3.5, and each equation has variables from k domains. Hence the probability of a potentially satisfiable equation in Ψ_{sc} to be actually satisfied by the re-assignment is at least $\Omega(\rho^k)$. It follows that the expected fraction of satisfied equations in Ψ_{sc} is $\Omega(\rho^k(\gamma - k\epsilon))$, and hence at least one of the re-assignments achieves this fraction of satisfaction. We have thus shown that there exists a feasible assignment for Ψ_{sc} satisfying an $\Omega(\rho^k(\gamma - k\epsilon))$ fraction of its equations.

We now choose a constant α so that $0 < \alpha < \min\{1 - ka\,, ac\}$ (note that such an α exists). If $\gamma > |\mathcal{F}|^{-\alpha}$, then

$$\rho^k(\gamma - k\epsilon) = |\mathcal{F}|^{-ak}(\gamma - O(\mathcal{F}^{-ac})) \gg \frac{2}{|\mathcal{F}|}$$

hence there exists a feasible assignment for Ψ_{sc} satisfying more than a $\frac{2}{|\mathcal{F}|}$ fraction of its equations.

■

3.3. Gap Amplification. The reduction now amplifies the soundness of ${\Psi_{sc}}'$ by joining conjunctions together into larger conjunctions, generating ${\Psi_{sc}}''$. The soundness of ${\Psi_{sc}}''$ is even stronger than needed, but it still has conjunctions rather than equations. The next subsection describes how conjunctions may be replaced by equations with only a small cost in the soundness, thus completing the reduction.

The system ${\Psi_{sc}}''$. Denote $N \doteq \lceil 1/\alpha \rceil$, where α is the constant mentioned in Claim Claim 3.4. The reduction generates ${\Psi_{sc}}''$ by taking the conjunction of every ordered N-tuple of (not necessarily distinct) conjunctions from ${\Psi_{sc}}'$, that is

$${\Psi_{sc}}'' = \{ \ \bigwedge_{i=1}^{N} \chi_i \ : \quad \forall\, i \quad \chi_i \in {\Psi_{sc}}' \ \}$$

Note that it takes polynomial time in $|{\Psi_{sc}}'|^N$, and hence in n, to generate ${\Psi_{sc}}''$. Since each conjunction in ${\Psi_{sc}}''$ is composed of a constant number of conjunctions from ${\Psi_{sc}}'$, the number of variables as well as the number of equations in each such conjunction is bounded by a constant. The next claim states the completeness and soundness properties of ${\Psi_{sc}}''$.

Claim 3.6. *Ψ_{sc}'' has the following properties:*

- *Completeness: If Ψ is completely satisfiable, then Ψ_{sc}'' is completely satisfiable as well.*
- *Soundness: If Ψ is at most $2/|\mathcal{F}|$-satisfiable, then Ψ_{sc}'' is at most $1/|\mathcal{F}|$-satisfiable.*

The claim follows easily from Claim Claim 3.4 and from the construction of Ψ_{sc}''.

3.4. From Conjunctions to Equations. Ψ_{sc}'' is a system of conjunctions where, as mentioned above, the number of equations in each conjunction is bounded by a constant. We would like the reduction to transform it from a system of conjunctions into the final system Ψ' of quadratic-equations, but first we make sure that the number of equations in all the conjunctions of Ψ_{sc}'' is the same. To do so the reduction adds equations of the form $0 = 0$ where necessary.

The system Ψ'. To transform Ψ_{sc}'' into Ψ', the reduction replaces each conjunction in Ψ_{sc}'' with the set of all linear-combinations over its equations (equations can be added or multiplied by a scalar, so the notion of a linear-combination of equations is well defined). Since the number of equations in each conjunction is constant the blow-up is polynomial in $|\mathcal{F}|$, and hence in n.

Since the number of variables in each conjunction of Ψ_{sc}'' is bounded by a constant, the number of variables in each equation of Ψ' is constant as well. In order to complete the proof of Theorem Theorem 2.2, it is left to show that Ψ' has the soundness and completeness properties. This follows immediately from Claim Claim 3.6 together with the next proposition.

Proposition 3.7 (conjunction replacement). *Let Ψ_a be a system of conjunctions of equations over $\mathcal{F}$, where the number of equations in each conjunction is the same. Let Ψ_b be the system obtained from Ψ_a by replacing every conjunction $\chi \in \Psi_a$ by all linear combinations over $\mathcal{F}$ of its equations (with multiplicities, if the same equation occurs more than once). Then*

- *If Ψ_a is completely satisfied by a certain assignment, then the same assignment will satisfy Ψ_b as well.*
- *If Ψ_a is at most γ-satisfiable then Ψ_b is at most $(\gamma + 1/|\mathcal{F}|)$-satisfiable.*

Proof. The first property is obvious from the definition of Ψ_b. To prove the second property, fix an assignment for the variables of Ψ_a and Ψ_b. Then it satisfies at most a γ fraction of the conjunctions in Ψ_a. For each conjunction χ in Ψ_a denote by $\omega(\chi)$ the fraction of equations replacing χ that are satisfied in Ψ_b. Since each conjunction of Ψ_a is replaced by the same number of equations, the fraction of satisfied equations in Ψ_b is the average of $\omega(\chi)$ over all the conjunctions $\chi \in \Psi_a$.

For a satisfied conjunction χ, $\omega(\chi) = 1$, and it is easy to observe that $\omega(\chi) = 1/|\mathcal{F}|$ for any unsatisfied conjunction χ. Since satisfied conjunctions make at most a γ fraction of the conjunctions in Ψ_a, we conclude that the fraction of satisfied equations in Ψ_b is no more than $\gamma + 1/|\mathcal{F}|$, as required. ■

4. The Composition-Recursion LDF-Reader Constructor

In this section we describe the construction of the LDF-readers needed for Lemma Lemma 3.3. These LDF-readers are the main step used in Section Section 3 to transform Ψ_{sc} to Ψ'. The construction is carried out by a recursive process, composing smaller LDF-readers on top of one another, hence the name "Composition-Recursion LDF-Readers".

As a first step, we show a constructor for *restricted* LDF-readers, where some of the domains in the representation are considered *active*. These LDF-readers have good parameters only in the case where active domains are given feasible assignments. By a composition of several such LDF-readers we then get a CR.

Definition 4.1 (restricted LDF-readers.). *A restricted LDF-reader $\mathcal{R}$ is an LDF-reader where some of the domains in the representation are considered* active. *The dimension, and the upper and lower degree parameters of all active domains must be the same. These parameters are called the active dimension, active upper-degree and active lower-degree of $\mathcal{R}$, and are denoted by $d_\star(\mathcal{R})$, $r_\star(\mathcal{R})$, and $s_\star(\mathcal{R})$ respectively.*

A local-reader L in $\mathcal{R}$ may have variables from at most one active domain, which is called the active domain of L and is denoted by $\mathrm{Dom}_\star(L)$.

Parameters of restricted LDF-readers. We measure the parameters of restricted LDF-readers only with respect to feasible assignments: An assignment for the representation of a restricted LDF-reader $\mathcal{R}$ is said to be *active-feasible* if the assignment of every active domain is feasible (unlike in the case of equation-systems, we do not require the assignment of all domains to be feasible). $\mathcal{R}$ is hence said to be a restricted (ρ, ϵ)-LDF-Reader if for any active-feasible assignment, the fraction of ρ-erroneous local-readers is at most ϵ. Note that in an encoding-assignment, all domains must still be given a good assignment.

Outline of this section. Subsection Section 4.1 shows a constructor for restricted LDF-readers (the definition of a constructor generalizes naturally for restricted LDF-readers), called Subspace-vs.-Point LDF-readers, SP's for short. These restricted LDF-readers are based upon the Plane-vs-Plane LDF-readers of Raz & Safra (1997), as proven (by a standard, albeit lengthy, probabilistic argument) in Subsection Section 4.2. The representation of an SP evaluating a tuple in a domain F contains, apart from F itself, only active domains, which have the same degree-parameters as F but a *constant* dimension parameter. Therefore, informally speaking, an SP LDF-reader uses constant-dimensional LDFs to represent an LDF over a space of higher dimension, and using evaluations of these constant-dimensional LDFs it produces consistent evaluations of the original LDF.

In Subsection Section 4.4 it is shown how the constant-dimensional active domains of an SP, $\mathcal{R}$, can be replaced by active domains that have non-constant dimension, but greatly decreased degree parameters. This allows the composition of other SP's over $\mathcal{R}$, as described in Subsection Section 4.5, to evaluate tuples in the replaced active domains. Subsection Section 4.6 shows how an iterative application of this procedure yields the Composition-Recursion

LDF-reader (which is not restricted) and Subsection Section 4.7 proves its properties, thus completing the proof of Lemma Lemma 3.3.

4.1. Subspace-vs.-Point LDF-Readers In this subsection we show the SP constructor – a constructor that generates Subspace-vs.-Point restricted LDF-readers. The representation of an SP that evaluates a k-tuple in a domain F contains, in addition to F, active domains with the same degree parameters as F but of dimension $k+2$. Each domain is associated with a $(k+2)$-dimensional subspace U in $\mathcal{F}^{d(F)}$; in an encoding-assignment each of them is assigned the restriction to U of the LDF assigned to F. Before we go into the description of the constructor, let us state its properties in the following lemma.

Lemma 4.2 (Subspace-vs.-Point LDF-reader). *There exists a constructor that given a domain F and a k-tuple of points in $\mathcal{F}^{d(F)}$, generates a restricted LDF-reader $\mathcal{R}$ as follows. The active domains of $\mathcal{R}$ have the same upper and lower-degree parameters as F, and their dimension parameter equals $k+2$. Moreover, $\mathcal{R}$ will have parameters $(\rho, O(\rho^{1/3}))$ for all ρ's which satisfy $\rho > (r(F)/|\mathcal{F}|)^{c_g} d(F)$, where $0 < c_g \leq 1/2$ is a global constant*[‡].

The Subspace-vs.-Point constructor We now describe how the SP constructor generates an LDF-reader $\mathcal{R}$, given a domain F and a k-tuple $(x_1, \ldots, x_k)$ of points in $\mathcal{F}^{d(F)}$. The SP constructor is used later as a procedure of the CR constructor, however in proving the parameters of the CR constructor we only rely on the properties that are stated in Lemma Lemma 4.2. Without loss of generality, throughout the construction it is assumed that $d(F) \geq k+2$ – it is easy to adapt the construction for the case $d(F) < k+2$.

The representation. Other than F itself, the representation only includes active domains, with upper-degree $r(F)$, lower-degree $s(F)$, and dimension $k+2$. The constructor first picks any $(k-1)$-dimensional affine subspace $U_0 \subseteq \mathcal{F}^{d(F)}$ which contains all the points x_i of the tuple (if the x_i's are in general position, there exists exactly one such subspace). Denote by $\mathcal{S}ub\mathcal{S}p(\mathcal{R})$ the set of $(k+2)$-dimensional affine subspaces $U \subseteq \mathcal{F}^{d(F)}$ which contain U_0. One active domain $\mathcal{D}_U$ is then constructed for every affine subspace $U \in \mathcal{S}ub\mathcal{S}p(\mathcal{R})$.

Identification functions. A good assignment $\mathcal{A}$ to F assigns to it an $[s(F), d(F)]$-LDF f. In the encoding-assignment, the assignment to each domain $\mathcal{D}_U$ represents the restriction of f to U. In order to represent $f|_U$ as an LDF over $\mathcal{F}^{k+2}$ the constructor chooses for each domain $\mathcal{D}_U$ an arbitrary linear isomorphism $\phi_U : U \to \mathcal{F}^{k+2}$, called the *identification function* of U, that identifies each point $y \in U$ with the point $\phi_U(y)$ in $\mathcal{F}^{k+2}$.

Encoding-assignments. Let $\mathcal{A}$ be a good assignment for F, assigning to it a $[s(F), d(F)]$-degree LDF f. The encoding-assignment for $\mathcal{A}$ extends it by assigning to each domain $\mathcal{D}_U$ the LDF $f \circ ({\phi_U}^{-1})$. Composing the assignment of $\mathcal{D}_U$ with ϕ_U therefore gives $f|_U$. Since ϕ_U is linear, $\mathcal{D}_U$ is assigned an $s(F)$-degree LDF, and hence the encoding-assignment of $\mathcal{A}$ is a good assignment.

[‡]This is the same constant as in Lemma Lemma 3.3, and in all other places where c_g appears

Local-readers. The SP constructor generates one local-reader for each domain $\mathcal{D}_U$ and point $y \in U$. Its active domain is $\mathcal{D}_U$, its local-test is the single linear equation $\mathcal{D}_U[\phi_U(y)] = F[y]$, and for every $1 \leq i \leq k$ its i'th evaluator is the term $\mathcal{D}_U[\phi_U(x_i)]$.

To get a better understanding of the structure of local-readers, fix an active-feasible assignment $\mathcal{A}$ for the representation variables, and consider a local-reader associated with a domain $\mathcal{D}_U$ and a point y. The LDF assigned to $\mathcal{D}_U$ represents an $r(F)$-degree LDF g_U over U, defined by $g_U(x) \doteq \mathcal{A}(\mathcal{D}_U[\phi_U(x)])$ (this is the composition of the LDF assigned to $\mathcal{D}_U$ with ϕ_U). The local-test therefore compares $g_U(y)$ with the assignment of $F[y]$, and the i'th evaluator returns $g_U(x_i)$.

If $\mathcal{A}$ is the encoding-assignment of an LDF f, then $F[y]$ is assigned $f(y)$, and g_U is the restriction of f to U. The local-test is hence satisfied in that case, and the values $g_U(x_i)$ returned by the evaluators are in fact the values of f at the points x_i.

The SP constructor works. It is easy to verify that the SP constructor indeed falls under the definition of a constructor. To verify the parameters of SP LDF-readers (which would conclude the proof of Lemma Lemma 4.2) consider an SP LDF-reader $\mathcal{R}$, that evaluates a k-tuple $(x_1, \ldots, x_k)$ in a domain F, and fix an active-feasible assignment $\mathcal{A}$ for its representation. As explained above, $\mathcal{A}$ determines an $r(F)$-degree LDF g_U over every affine subspace $U \in \mathcal{S}ubSp(\mathcal{R})$.

The local-test of the local-reader determined by an affine subspace $U \in \mathcal{S}ubSp(\mathcal{R})$ and a point $y \in U$ verifies that $g_U(y) = \mathcal{A}(F[y])$, and its evaluators return the values of g_U at the points $x_1, \ldots, x_k$. The local-reader is ρ-erroneous if the local-test is satisfied, yet the values $g_U(x_1), \ldots, g_U(x_k)$ are not the evaluation of any ρ-permissible LDF (with respect to the assignment of F) at $x_1, \ldots, x_k$. The next lemma bounds the fraction of erroneous local-readers, thus proving that $\mathcal{R}$ has the parameters required in Lemma Lemma 4.2.

Lemma 4.3 (SP parameters). *For some global constant* $0 < c_g \leq 1/2$, *and for all* ρ*'s which satisfy* $\rho > (r(F)/|\mathcal{F}|)^{c_g} d(F)$, *the following holds.*

Let U *be a random affine subspace in* $\mathcal{S}ubSp(\mathcal{R})$, *and* y *be a random point in* U *(this determines a random local-reader). Let* $\mathbf{E}rr$ *be the event that* $g_U(y) = \mathcal{A}(F[y])$, *yet* $g_U(x_1), \ldots, g_U(x_k)$ *are not the evaluation of any* ρ*-permissible LDF (with respect to the assignment of* F*) at* $x_1, \ldots, x_k$. *Then* $\Pr[\mathbf{E}rr] = O(\rho^{1/3})$.

In fact we show a stronger statement than the above lemma. We bound by $O(\rho^{1/3})$ the probability that g_U agrees with the assignment of F at y, yet g_U is not the restriction to U of any ρ-permissible LDF (it is easy to observe that this implies Lemma Lemma 4.3).

4.2. Subspace-vs.-Point Parameters In this subsection we prove Lemma Lemma 4.3 based on Raz & Safra (1997), by a somewhat lengthy series of technical arguments. While we are interested in the case where an LDF is associated with every $(k+2)$-dimensional subspace in a certain set $\mathcal{S}ubSp(\mathcal{R})$, the Raz & Safra (1997)-Lemma deals with the case where each plane (2-dimensional affine subspace) is associated with an LDF defined over it.

Definition 4.4 (plane-assignment). *Suppose that every plane P in $\mathcal{F}^{d(F)}$ is associated with an $r(F)$-degree LDF g_P over P. The correspondence $P \to g_P$ is called a plane-assignment. An LDF g_P is called the plane-LDF assigned to P.*

Another difference is that the Raz & Safra (1997)-Lemma discusses a different kind of permissibility, measured with respect to the plane-assignment instead of with respect to the assignment of F.

Definition 4.5 (planewise-permissibility). *Let $(P \to g_P)$ be a plane-assignment. An $r(F)$-degree LDF f over $\mathcal{F}^{d(F)}$ is said to be ρ-planewise-permissible if for at least a ρ-fraction of the planes P, $g_P = f|_P$.*

A plane-LDF g_P is said to be ρ-planewise-permissible if it is the restriction to P of a ρ-planewise-permissible LDF f over $\mathcal{F}^{d(F)}$.

We now state the discussed lemma from Raz & Safra (1997). It shows that planewise-permissibility can be tested by comparing the plane-LDFs assigned to two line-intersecting planes. If the plane-LDFs agree on the line then with high probability they are both planewise-permissible.

Lemma 4.6 (Raz & Safra 1997). *There exists a global constant $0 < c_g \leq 1/2$, such that for any $\rho > (r(F)/|\mathcal{F}|)^{c_g} d(F)$ the following holds.*

Fix a plane-assignment $(P \to g_P)$. Let ℓ be a random line in $\mathcal{F}^{d(F)}$ and let P_1 and P_2 be two random, independently chosen planes that contain ℓ. Denote by $\mathbf{E}rr$ the event that the plane-LDF assigned to P_1 agrees on ℓ with the plane-LDF assigned to P_2, yet they are not both ρ-planewise-permissible. Then $\Pr[\mathbf{E}rr] = O(\rho)$.

Starting from Lemma Lemma 4.6, we gradually approach Lemma Lemma 4.3 by a sequence of technical claims: Claim Claim 4.7 shows Lemma Lemma 4.6 to hold even if two plane-LDFs are compared on a point rather than a line. Claim Claim 4.8 deals with the case where a plane-LDF is compared against the assignment of F at a certain point, rather than against another plane-LDF. Claim Claim 4.12 goes from planewise-permissibility to permissibility, showing that a random plane-LDF g_P that agrees with the assignment of F at a random point on P is with high probability the restriction to P of a ρ-permissible LDF. Finally Claim Claim 4.16 completes the proof of Lemma Lemma 4.3 by showing that the same holds for LDFs g_U associated with a random subspace $U \in \mathcal{S}ub\mathcal{S}p(\mathcal{R})$, instead of plane-LDFs.

Claim 4.7. *Fix a plane assignment $(P \to g_P)$, and suppose ρ satisfies the requirements of Lemma Lemma 4.6. Let y be a random point in $\mathcal{F}^{d(F)}$, let ℓ be a random line containing it, and let P_1 and P_2 be random independently chosen planes that contain ℓ. Denote by $\mathbf{E}rr$ the event that the plane-LDF assigned to P_1 agrees on y with the plane-LDF assigned to P_2, yet they are not both ρ-planewise-permissible. Then $\Pr[\mathbf{E}rr] = O(\rho)$.*

Proof. First note that y can be considered to be a random point on a randomly chosen line ℓ in $\mathcal{F}^{d(F)}$, instead of the other way around.

The only case where $\mathbf{E}rr$ occurs yet the event from Lemma Lemma 4.6 does not, is when the restrictions to ℓ of the plane-LDFs of P_1 and P_2 differ, yet they agree on y. Since the restrictions to ℓ are $r(F)$-degree LDFs, if they differ then the probability of agreement on the random point y is at most $r(F)/|\mathcal{F}| \leq \rho$. Hence by Lemma Lemma 4.6, $\Pr[\mathbf{E}rr] \leq \rho + O(\rho) = O(\rho)$. ■

The next step is to compare a plane-LDF to the assignment of $F[y]$ for some point y on it, instead of to the value at y of another plane-LDF .

CLAIM 4.8. *Fix a plane assignment $(P \to g_P)$, and suppose ρ satisfies the requirements of Lemma Lemma 4.6. Let P be a random plane and y be a random point on P. Denote by $\mathbf{E}rr$ the event that $g_P(y) = \mathcal{A}(F[y])$, yet g_P is not ρ-planewise-permissible. Then* $\Pr[\mathbf{E}rr] = O(\rho^{1/2})$.

PROOF. To be able to apply Claim Claim 4.7, we redefine y and P, and introduce new random variables as follows. Let y be a random point in $\mathcal{F}^{d(F)}$, ℓ be a random line containing y, and P and P' be random independently chosen planes that contain ℓ (note that to obtain the claim it is enough to prove a bound on $\Pr[\mathbf{E}rr]$ in these settings). Let $\mathbf{E}rr_2$ be the event that g_P and $g_{P'}$ agree on y yet they are not both ρ-planewise-permissible. Claim Claim 4.7 implies that the probability of $\mathbf{E}rr_2$ is bounded by $O(\rho)$.

To use the bound we have for $\mathbf{E}rr_2$ we first show that for every fixed line ℓ_0 and point $y_0 \in \ell_0$,

$$\Pr[\mathbf{E}rr|\ell = \ell_0, y = y_0] \leq \left(\Pr[\mathbf{E}rr_2|\ell = \ell_0, y = y_0]\right)^{1/2} \tag{4.9}$$

Let $\mathbf{E}rr'$ be the event that $g_{P'}(y) = \mathcal{A}(F[y])$ yet $g_{P'}$ is not ρ-planewise-permissible (it is similar to $\mathbf{E}rr$, only for P' instead of P). Obviously

$$\Pr[\mathbf{E}rr_2|\ell = \ell_0, y = y_0] \geq \Pr[\mathbf{E}rr \wedge \mathbf{E}rr'|\ell = \ell_0, y = y_0] \tag{4.10}$$

because the event on the left-hand side contains the one on the right-hand side. Since P and P' are independently chosen given ℓ_0, we have

$$\begin{aligned}\Pr[\mathbf{E}rr \wedge \mathbf{E}rr'|\ell = \ell_0, y = y_0] &= \Pr[\mathbf{E}rr|\ell = \ell_0, y = y_0] \cdot \Pr[\mathbf{E}rr'|\ell = \ell_0, y = y_0] \\ &= \left(\Pr[\mathbf{E}rr|\ell = \ell_0, y = y_0]\right)^2\end{aligned}$$

which together with ((4.10)) implies Equation (4.9).

One may discard the conditioning in Equation (4.9), obtaining

$$\Pr[\mathbf{E}rr] \leq \Pr[\mathbf{E}rr_2]^{1/2}$$

using the law of complete probability and the concavity of the square-root function. Since the probability of $\mathbf{E}rr_2$ is bounded by $O(\rho)$, this obtains the claim. ■

Our next step is to convert the statement of Claim Claim 4.8 from planewise-permissibility to permissibility in the usual sense. This requires the following bound on the number of planewise-permissible LDFs.

Claim 4.11. *Fix a plane assignment $(P \to g_P)$, and suppose ρ satisfies the requirements of Lemma Lemma 4.6. Then the number of ρ-planewise-permissible LDFs is less than $2\rho^{-1}$.*

Proof. The proof is similar to that of Claim Claim 4.14 below. ∎

We can now prove the analogue of Claim Claim 4.8 for permissibility in the usual sense.

Claim 4.12. *Fix a plane assignment $(P \to g_P)$, and suppose ρ satisfies the requirements of Lemma Lemma 4.6. Let P be a random plane and y be a random point on P. Denote by $\mathbf{E}rr$ the event that $g_P(y) = \mathcal{A}(F[y])$, yet there is no ρ-permissible LDF f (with respect to the assignment of F), such that $g_P = f|_P$. Then* $\Pr[\mathbf{E}rr] = O(\rho^{1/3})$.

Proof. We separate $\mathbf{E}rr$ into two events, and bound the probability of each by $O(\rho^{1/3})$: Let $\mathbf{E}rr_1$ be the event where $\mathbf{E}rr$ occurs and in addition g_P **is not** $\rho^{2/3}$-planewise-permissible; and let $\mathbf{E}rr_2$ be the event where $\mathbf{E}rr$ occurs and in addition g_P is $\rho^{2/3}$-planewise-permissible.

By applying Claim Claim 4.8 using $\rho^{2/3}$ instead of ρ, we obtain that the probability of $\mathbf{E}rr_1$ is bounded by $O(\rho^{1/3})$ as required (since $\rho > (r(F)/|\mathcal{F}|)^{c_g} d(F)$ as required in Lemma Lemma 4.6, $\rho^{2/3}$ satisfies this requirement as well).

It is left to bound the probability of $\mathbf{E}rr_2$. By definition it occurs only when $g_P(y) = \mathcal{A}(F[y])$ and there exists a $\rho^{2/3}$-planewise-permissible LDF f which is not ρ-permissible, such that $g_P = f|_P$. For an LDF f over $\mathcal{F}^{d(F)}$, denote by $\mathbf{E}rr_3(f)$ the event where $g_P(y) = \mathcal{A}(F[y])$ and $g_P = f|_P$ (note that this implies $f(y) = \mathcal{A}(F[y])$). Then the probability of $\mathbf{E}rr_2$ is bounded by the sum of $\Pr[\mathbf{E}rr_3(f)]$ over all $\rho^{2/3}$-planewise-permissible LDFs f which are not ρ-permissible.

Let us bound the probability of $\mathbf{E}rr_3(f)$ for such an LDF f. Since f is not ρ-permissible the probability that it satisfies $f(y) = \mathcal{A}[F(y)]$ is bounded by ρ, because y is a uniformly random point in $\mathcal{F}^{d(F)}$. The probability of $\mathbf{E}rr_3(f)$ is therefore bounded by ρ as well. Since f should be $\rho^{2/3}$-planewise-permissible, there can be at most $2\rho^{-2/3}$ such f's by Claim Claim 4.11, and therefore a bound of $2\rho^{-2/3}\rho = O(\rho^{1/3})$ is obtained for the probability of $\mathbf{E}rr_2$. ∎

Note that the statement of Claim Claim 4.12 is similar to what we wish to establish (see the remark following Lemma Lemma 4.3), only for planes rather than $(k+2)$-dimensional subspaces in $\mathcal{S}ub\mathcal{S}p(\mathcal{R})$. Claim Claim 4.16 proves that by considering a random plane P, $y \in P \subseteq U$, in addition to the random subspace U and the random point $y \in U$. Claim Claim 4.12 can be applied to P to obtain Claim Claim 4.16, but for it to be applicable it should be shown that when a random subspace $U \in \mathcal{S}ub\mathcal{S}p(\mathcal{R})$ is chosen , and then a random plane P contained in U, and then a point $y \in P$, P and y are almost uniformly distributed. This is shown in the following claim.

Claim 4.13. *Let U be a random subspace in $\mathcal{S}ub\mathcal{S}p(\mathcal{R})$, and let P be a random plane contained in U and y a random point in P. The distribution of P and y is almost uniform, that is if $\mathcal{P}lns$ denotes the set of planes in $\mathcal{F}^{d(F)}$ then*

$$\sum_{P_0 \in \mathcal{P}lns} |\Pr(P = P_0) - |\mathcal{P}lns|^{-1}| \leq O(|\mathcal{F}|^{-1})$$

and

$$\sum_{y_0 \in \mathcal{F}^{d(F)}} |\Pr(y = y_0) - |\mathcal{F}|^{-d(F)}| \le O(|\mathcal{F}|^{-1})$$

Proof. We begin by proving the second inequality. Observe that since all the subspaces in $\mathcal{S}ub\mathcal{S}p(\mathcal{R})$ contain U_0 the probability of the random point y in U to yield a specific point in U_0 is higher than $|\mathcal{F}|^{-d(F)}$, the probability of a uniformly random point. Also, the probability of y to yield a point outside of U_0 is smaller than $|\mathcal{F}|^{-d(F)}$. Hence

$$\begin{aligned}
&\sum_{y_0 \in \mathcal{F}^{d(F)}} |\Pr[y = y_0] - |\mathcal{F}|^{-d(F)}| \\
&\quad = \sum_{y_0 \in U_0} (\Pr[y = y_0] - |\mathcal{F}|^{-d(F)}) + \sum_{y_0 \in \mathcal{F}^{d(F)} \setminus U_0} (|\mathcal{F}|^{-d(F)} - \Pr[y = y_0]) \\
&\quad = 2 \sum_{y_0 \in U_0} (\Pr[y = y_0] - |\mathcal{F}|^{-d(F)}) \qquad \text{(since probabilities sum up to 1)} \\
&\quad < 2 \Pr[y \in U_0] = 2|U_0|/|U| = 2|\mathcal{F}|^{-3} = O(|\mathcal{F}|^{-1}),
\end{aligned}$$

thus obtaining the second inequality. The proof of the first inequality uses similar arguments, however it is more tedious. We therefore only sketch it here.

Two main observations are needed to prove the first inequality. First, one needs to observe that if three random points y_1, y_2, y_3 are independently chosen within U, then their joint distribution is within statistical distance $O(|\mathcal{F}|^{-1})$ from the distribution of three points chosen independently from $\mathcal{F}^{d(F)}$. This follows by applying the argument used to prove the second inequality three consecutive times. At each application, U is conditioned on containing the affine space generated by U_0 and the previously selected points.

The second observation, which is an easy exercise, is that the distribution of the affine span of three independently chosen points in $\mathcal{F}^{d(F)}$, is within statistical distance $O(|\mathcal{F}|^{-1})$ from the distribution of a random plane in $\mathcal{P}lns$. Together with the first observation we obtain the desired inequality. ■

Before we move to the final claim, we need the following two bounds. Claim Claim 4.14, which appears in Section Section 3 as Proposition Proposition 3.5, bounds number of ρ-permissible LDFs. Claim Claim 4.15 bounds the fraction of planes on which two distinct LDFs may agree.

Claim 4.14. *Suppose ρ satisfies the requirements of Lemma Lemma 4.6. Then there are less than $2/\rho$ ρ-permissible LDFs.*

Proof. Assume for the sake of contradiction that there exists a set $\mathcal{P}er$ containing $2/\rho$ distinct ρ-permissible LDFs. For each LDF $f \in \mathcal{P}er$ denote by $U(f)$ the set of points $y \in \mathcal{F}^{d(F)}$ such that f is the only LDF in $\mathcal{P}er$ satisfying $f(y) = \mathcal{A}(F[y])$.

Each LDF $f \in \mathcal{P}er$ is ρ-permissible, hence it agrees with $\mathcal{A}(F)$ on at least a ρ-fraction of the points. We bound from above the fraction of points for which it also agrees with other

LDFs in $\mathcal{P}er$. Any other LDF in $\mathcal{P}er$ agrees with f on at most an $\frac{r(F)}{|\mathcal{F}|}$ fraction of the points, so overall f may agree with other LDFs in $\mathcal{P}er$ on at most a $\frac{2r(F)}{\rho|\mathcal{F}|}$ fraction of the points. From the assumption $\rho > \left(\frac{r(F)}{|\mathcal{F}|}\right)^{c_g} d(F)$ it follows in particular that $\rho^2 > 4r(F)/|\mathcal{F}|$ (recall that $c_g < 1/2$ and that we assume $d(F) \geq k+2 > 2$), and therefore $\frac{2r(F)}{\rho|\mathcal{F}|} < \frac{1}{2}\rho$.

f thus agrees with $\mathcal{A}(F)$ on at least ρ of the points, and on less than a $\frac{1}{2}\rho$ fraction of the points it agrees with other LDFs in $\mathcal{P}er$. It follows that for every $f \in \mathcal{P}er$, $U(f)$ contains more than a $\frac{1}{2}\rho$ fraction of the points. Since the sets $U(f)$ are disjoint, it follows that the fraction of all points contained in any of the $U(f)$'s is greater than $\frac{1}{2}\rho \cdot |\mathcal{P}er| \geq 1$. This is a contradiction. ■

Claim 4.15. *For any $t > 0$, two distinct $[r,t]$-LDFs must disagree on all but at most an $r/|\mathcal{F}|$ fraction of their possible restrictions to planes.*

Proof. Let f and g be two distinct r-degree LDFs over $\mathcal{F}^t$, and let P be a random plane in $\mathcal{F}^t$. We are to evaluate the probability that $f|_P$ equals $g|_P$. Let y be a random point on P. y is uniformly distributed in $\mathcal{F}^t$, and therefore it produces a disagreement with probability at least $1 - \frac{r}{|\mathcal{F}|}$ (this is a well known property of LDFs). Since y can produce a disagreement only in the case that there is a disagreement over P, it implies that there is a disagreement over P with probability at least $1 - \frac{r}{|\mathcal{F}|}$. ■

The following claim directly implies Lemma Lemma 4.3.

Claim 4.16. *Suppose ρ satisfies the requirements of Lemma Lemma 4.6. Let U be a random affine subspace in $\mathcal{S}ubSp(\mathcal{R})$, and y be a random point in U. Let $\mathbf{E}rr$ be the event that $g_U(y) = \mathcal{A}(F[y])$, yet there is no ρ-permissible LDF (with respect to the assignment of F) whose restriction to U gives g_U. Then $\Pr[\mathbf{E}rr] = O(\rho^{1/3})$.*

Proof. Let P be a random plane contained in the random subspace U. Without loss of generality, we may assume that y is a random point in P.

We define two events $\mathbf{E}rr_1$ and $\mathbf{E}rr_2$ such that $\mathbf{E}rr_1 \cup \mathbf{E}rr_2$ contains $\mathbf{E}rr$, and bound the probability of each by $O(\rho^{1/3})$. Let $\mathbf{E}rr_1$ be the event that $g_U(y) = \mathcal{A}(F[y])$, yet there is no ρ-permissible LDF f that agrees with g_U on P, namely $f|_P = g_U|_P$. Let $\mathbf{E}rr_2$ be the event that there is no ρ-permissible LDF whose restriction to U gives g_U, yet there exists such an LDF that agrees with g_U on P. Obviously $\mathbf{E}rr \subseteq \mathbf{E}rr_1 \cup \mathbf{E}rr_2$.

Bounding $\Pr[\mathbf{E}rr_2]$. For a ρ-permissible LDF f, let $\mathbf{E}rr_3(f)$ be the event that $f|_U \neq g_U$, yet $f|_P = g_U|_P$. $\mathbf{E}rr_2$ is contained in the union of the events $\mathbf{E}rr_3(f)$ over all ρ-permissible LDFs f. For a ρ-permissible LDF f,

$$\Pr[\mathbf{E}rr_3(f)|U = V] \leq r(F)/|\mathcal{F}|$$

for every fixed subspace $V \in \mathcal{S}ubSp(\mathcal{R})$, by applying Claim Claim 4.15 to V. It follows that $\Pr[\mathbf{E}rr_3(f)]$ is bounded by $r(F)/|\mathcal{F}|$ as well. Since there are less than $2/\rho$ ρ-permissible LDFs in all, we obtain that

$$\Pr[\mathbf{E}rr_2] < \frac{2r(F)}{\rho|\mathcal{F}|} < \rho$$

(the last inequality follows easily from the restriction on ρ).

Bounding $\Pr[\mathbf{E}rr_1]$**.** We change the distribution of U, P and y, by letting P be a random plane in $\mathcal{F}^{d(F)}$, U be a random space in $\mathcal{S}ubSp(\mathcal{R})$ that contains P, and y be a random point in P. By Claim Claim 4.13, the statistical distance between the new distribution of P, which is uniform, and the original distribution is $O(|\mathcal{F}|^{-1})$. Under both the original and the new distributions, the distribution of U and y conditioned on P being a fixed plane P_0 are the same – U is a random space in $\mathcal{S}ubSp(\mathcal{R})$ that contains P_0 and y is a random point in P_0. It follows that the statistical distance between the new joint distribution of U, P and y, and the original distribution is bounded by $O(|\mathcal{F}|^{-1})$. It is hence enough to bound the probability of $\mathbf{E}rr_1$ according to the new distribution.

Let $g_P \doteq g_U|_P$ be considered as a random plane-assignment for the (random) plane P. The definition of $\mathbf{E}rr_1$ can hence be articulated as the event that $g_P(y) = \mathcal{A}(F[y])$, yet there is no ρ-permissible LDF f such that $g_P = f|_P$. Claim Claim 4.12 naturally extends to the case where the plane-assignments is random as long as the assignment to F is not random, hence it implies that $\Pr[\mathbf{E}rr_2] = O(\rho^{1/3})$ (note that P is a uniformly random plane and y is a random point in P). ■

4.3. Overview of the CR-Constructor. Let us give an overview of the CR (Composition-Recursion) constructor. Given a domain and a tuple, the CR constructor generates a constant-length sequence of restricted LDF-readers that ends with the final, unrestricted, CR LDF-reader. Each transformation of an LDF-reader $\mathcal{R}$ in the sequence into the next (except for the final one) has the same two steps as follows.

Extension. In the first step, an *extension-procedure* is applied to each active domain of $\mathcal{R}$, replacing it by a domain with greatly reduced degree parameters in the price of an increased dimension parameter. The active degree and dimension parameters of $\mathcal{R}$ are thus changed, but its other properties are maintained.

Composition. The second step is the application of the *composition* procedure, which incorporates new LDF-readers into $\mathcal{R}$. First, it generates several new LDF-readers using the SP constructor (actually, any constructor with properties as in Lemma Lemma 4.2 will do), applying it to different active domains and tuples. The domains generated in the process then become active instead of the old active domains. These new active domains are of constant dimension, and because of the extension step their degree parameters are greatly reduced with respect to the active domains of $\mathcal{R}$. Finally the newly generated local-readers are plugged into the local-readers of $\mathcal{R}$, generating the next LDF-reader in the sequence.

We proceed as follows. First, in the next subsection, we give a formal definition of an extension and show the two extension-procedures used by the CR constructor. In Subsection Section 4.5 we describe the composition procedure and prove its properties. In Subsection Section 4.6 we describe the CR constructor and then we prove its correctness in Subsection Section 4.7.

4.4. Extensions. An extension of a domain F is a domain G which *contains* the variables of F: Each variable $F[x]$ is endowed with another name $G[\phi(x)]$. The function $\phi : \mathcal{F}^{d(F)} \to \mathcal{F}^{d(G)}$ is called the *gluing* of F to G. The extension must preserve good and feasible assignments as follows.

Definition 4.17 (extension). *Let F be a domain, and let G be a domain that contains the variables of F. G is called an* extension *of F if the following properties hold:*

- *Extension Property: Any good assignment $\mathcal{A}$ for F can be extended to a good assignment for G, called the* encoding-assignment *or the* encoding LDF *of $\mathcal{A}$.*
- *Restriction Property: The restriction to F of any feasible assignment for G is a feasible assignment for F.*

The point about extensions is that they allow the representation of an LDF assigned to a domain F by an encoding LDF with different properties. We can hence replace the active domains of a restricted (ρ, ϵ)-LDF-reader $\mathcal{R}$ by their extensions and obtain a restricted (ρ, ϵ)-LDF-reader where the active degree parameters are different, usually considerably smaller.

Proposition 4.18 (extension). *Let $\mathcal{R}$ be a restricted (ρ, ϵ)-LDF-reader, evaluating a tuple in a domain F. Suppose that for each active domain G of $\mathcal{R}$, $e(G)$ is an extension of G, and that all extensions have the same parameters. Then the LDF-reader $\mathcal{R}'$ obtained from $\mathcal{R}$ by just declaring these extensions as the active domains of $\mathcal{R}'$ is a restricted (ρ, ϵ)-LDF-reader.*

Proof. It is given that all the active domains of $\mathcal{R}'$ have the same parameters. To show that $\mathcal{R}'$ is a valid restricted LDF-reader we define an encoding-assignment of $\mathcal{R}'$, for every good assignment to F.

Given a good assignment for F, let $\mathcal{A}$ be its encoding-assignment with respect to $\mathcal{R}$. For each active domain G of $\mathcal{R}$, assign the encoding-assignment of $\mathcal{A}(G)$ to its extension $e(G)$. This obtains a good assignment for the representation of $\mathcal{R}'$, and since the assignments to the variables of $\mathcal{R}$ are not changed all local-tests are satisfied and all local-readers return evaluations consistent with the assignment of F.

The fact that $\mathcal{R}'$ has parameters (ρ, ϵ) follows easily from the restriction property of each extension $e(G)$, which implies that the restriction of an active-feasible assignment for $\mathcal{R}'$ yields an active-feasible assignment for $\mathcal{R}$. ∎

Extension-procedures. An extension-procedure is an algorithm which given a domain F, generates an extension G of F. The running time of the algorithm must be polynomial in $|\mathcal{F}|^{d(G)}$. We next show the two extension-procedures used by the CR constructor – the power-substitution and the linearization extension-procedures. In the sequence of restricted LDF-readers that is generated by the CR constructor (see the overview above), the power-substitution extension-procedure is used in the generation of all restricted LDF-readers but the last. The last restricted LDF-reader is generated using the linearization extension-procedure, and thus has active domains with lower-degree and upper-degree 1. The final, unrestricted, CR LDF-reader is obtained by replacing each such domain with variables that represent the coefficients of a linear function.

Given a domain F, the power-substitution extension-procedure constructs an extension G with greatly reduced degree parameters in the price of increasing the dimension parameter. The linearization extension-procedure, when applied to a domain F, yields a domain G with lower-degree and upper-degree 1. The dimension of G is, however, exponential in the degree parameters of F, hence the linearization is applied by the CR constructor only after very small active degree parameters are achieved.

Gap consumption. Recall that if G is an extension of a domain F, then for every good assignment to F there must be a good assignment for G which extends it. This may (and in fact does, in all cases discussed herein) impose a lower-bound on the lower-degree of G. Similarly, the restriction property of extensions yields an upper-bound on the upper-degree of G. In the case of the power-substitution extension-procedure, this forces the gap between the upper and lower-degrees of G to be smaller than for F. That is, if G is the extension of a domain F obtained by the power-substitution extension-procedure, then $r(G)/s(G) < r(F)/s(F)$.

Since the CR constructor applies the power-substitution extension-procedure several times as described in the overview, if it is applied to a domain F where the gap between the upper-degree and lower-degree is not large enough (see Lemma Lemma 3.3), domains are eventually created where the upper-degree is smaller than the lower-degree. Since the linearization extension-procedure is not applicable to such domains, the CR constructor would not be able to construct an LDF-reader for F.

The power-substitution extension-procedure. We begin by stating the properties of the power-substitution extension-procedure. For simplicity, we omit floor and ceiling signs where they are not essential.

Proposition 4.19 (power-substitution). *There exists an extension-procedure, called* power-substitution, *which given a domain F and a parameter $b > 1$, generates an extension G of F with the following parameters: For $t \doteq \lceil \log_b(s(F)+1) \rceil$,*

- $d(G) = d(F)t$
- $s(G) = d(F)t(b-1)$
- $r(G) = r(F)/b^{t-1} \ \big(\geq r(F)/s(F)\big)$

The procedure is based on the idea that by replacing powers of variables in an LDF f with new auxiliary variables, the degree of f may be decreased dramatically. For example, we fix an LDF over one variable $f(u_1) = u_1^{12} + u_1^{25}$ (the handling of multi-variate LDFs is very similar), and show an encoding LDF g over three variables.

g is obtained from f by substituting powers of u_1 with new variables. Informally speaking, if v_0 is considered as representing u_1, v_1 is considered as representing u_1^3, and v_2 – as representing u_1^9, then $u_1^{12} = v_1 v_2$, and $u_1^{25} = v_0 v_1^2 v_2^2$ (note that we used the base 3 representation of 12 and 25). Replacing these terms in f obtains an LDF $g(v_0, v_1, v_2) = v_1 v_2 + v_0 v_1^2 v_2^2$ of degree 5 rather than 25. g encodes and extends f in the sense that $g(u_1, u_1^3, u_1^9) = f(u_1)$ for every $u_1 \in \mathcal{F}$.

For a domain G, obtained from a domain F using the power-substitution extension-procedure with parameter b, an LDF f of degree $s(F)$ assigned to F is encoded by an LDF g over $\mathcal{F}^{d(G)}$ as follows. g is obtained from f by taking an auxiliary variable for each power of the form $u_i^{b^e}$ of a variable u_i of f. Any other power u_i^j of u_i can then be replaced by a monomial over the new variables of degree at most $b-1$ in each variable, using the base-b representation of j.

Proof of Proposition Proposition 4.19: We begin by describing the power-substitution extension-procedure and then prove that it has the required properties.

The procedure. Given a domain F and a parameter b, the procedure first generates a domain G with parameters as stated in the proposition. It then generates a gluing function $\phi : \mathcal{F}^{d(F)} \to \mathcal{F}^{d(G)}$ as follows:
For every $x = (u_1, \ldots, u_{d(F)}) \in \mathcal{F}^{d(F)}$, $\phi(x)$ is defined to be

$$(u_1, u_1^b, u_1^{b^2}, \ldots, u_1^{b^{t-1}}, \quad u_2, u_2^b, u_2^{b^2}, \ldots, u_2^{b^{t-1}}, \quad \ldots \quad , u_{d(F)}, \ldots, u_{d(F)}^{b^{t-1}}) \in \mathcal{F}^{d(G)}$$

Finally, each variable of the form $G[\phi(x)]$ is discarded, and the name $G[\phi(x)]$ is endowed to the variable $F[x]$ (which now has more than one name).

It is clear that the above procedure generates a domain G with the required parameters, in time polynomial in $|\mathcal{F}|^{d(G)}$. It remains to show that G is indeed an extension of F, namely that it has the extension and restriction properties.

Extension property. Suppose F is assigned an $[s(F), d(F)]$-LDF f (namely a good assignment). We now show its encoding LDF g – it should be an $[s(G), d(G)]$-LDF satisfying $g \circ \phi = f$, so that when assigned to G it does not conflict with F. First, let $f(u_1, \ldots, u_{d(F)})$ be written as a polynomial formula P over the variables $u_1, \ldots, u_{d(F)}$. P is transformed into a polynomial formula P' over the variables

$$v_{(1,0)}, v_{(1,1)}, .., v_{(1,t-1)}, \quad v_{(2,0)}, v_{(2,1)}, .., v_{(2,t-1)}, \quad \ldots \quad , v_{(d(F),0)}, .., v_{(d(F),t-1)}$$

by replacing each term u_i^j in P with a monomial $m_{(i,j)}$ over $v_{(i,0)}, .., v_{(i,t-1)}$: Since the term u_i^j appears in P we gather that $j \leq s(F)$, and hence its representation as a number in base b has at most t digits. Let $e_{t-1}, \ldots, e_1, e_0$ be the base b representation of j, and let

$$m_{(i,j)} \doteq (v_{(i,0)})^{e_0} (v_{(i,1)})^{e_1} \ldots (v_{(i,t-1)})^{e_{t-1}}$$

Replacing each term u_i^j in P with the monomial $m_{(i,j)}$ we obtain P', and then we define g by

$$g(v_{(i,0)}, \ldots, v_{(d(F),t-1)}) \doteq P'(v_{(i,0)}, \ldots, v_{(d(F),t-1)})$$

Since each monomial $m_{(i,j)}$ is of degree at most $t(b-1)$, it easily follows that g is an $[s(G), d(G)]$-LDF. Considering $m_{(i,j)}$ as a function over $\mathcal{F}^{d(G)}$, it is also easy to see that for all $(u_1, \ldots, u_{d(F)})$, $(m_{(i,j)} \circ \phi)(u_1, \ldots, u_{d(F)}) = u_i^j$. It follows that $g \circ \phi = f$, and hence g is indeed an encoding-LDF.

Restriction property. Suppose G is given a feasible assignment, namely it is assigned an $[r(G), d(G)]$-LDF g. The restriction of the assignment to F is hence an LDF f over $\mathcal{F}^{d(F)}$, given by $f = g \circ \phi$. The degree of f is at most $\deg(f) = \deg(g)\deg(\phi) = r(G)b^{t-1} = r(F)$. The restriction is hence a feasible assignment for F, as required.

The linearization extension-procedure. The linearization extension-procedure is very similar to the power-substitution procedure. The idea is to encode an LDF f by a *linear* LDF, replacing every monomial by a new auxiliary variable (recall that in the power-substitution, auxiliary variables where only introduced for some powers of variables in f). Since many auxiliary variables are used, the dimension increases dramatically.

Proposition 4.20 (linearization). *There exists an extension-procedure called* linearization, *which given a domain F with $s(F) \leq r(F)$, generates an extension G with the following parameters: For $t \doteq \binom{s(F)+d(F)}{d(F)}$,*

- $d(G) = t$
- $s(G) = 1$
- $r(G) = 1$

Proof. We begin by describing the linearization extension-procedure, and then prove that it has the required properties.

The procedure. Given a domain F, the procedure first generates a domain G with parameters as stated above. To generate the gluing function, the procedure picks an arbitrary enumeration $m_1, \ldots, m_t$ of the monomial functions of degree at most $s(F)$ over $\mathcal{F}^{d(F)}$ (note that there are exactly t such monomials). The gluing function $\phi : \mathcal{F}^{d(F)} \rightarrow \mathcal{F}^{d(G)}$ is then defined by

$$\forall\, x \in \mathcal{F}^{d(F)} \qquad \phi(x) \doteq (m_1(x), \ldots, m_t(x))$$

Having defined the gluing function, F and G are then “glued” in the usual way – each variable of the form $G[\phi(x)]$ is discarded, and the name $G[\phi(x)]$ is endowed to the variable $F[x]$ (which now has more than one name).

The procedure clearly generates a domain G with the required parameters, in time polynomial in $|\mathcal{F}|^{d(G)}$. It remains to verify that G has the extension and restriction properties.

Extension property. Suppose F is assigned an $[s(F), d(F)]$-LDF f, and let us construct its encoding LDF – a linear LDF over $\mathcal{F}^{d(G)}$ satisfying $g \circ \phi = f$. First, one can write f as a linear-combination of the monomial functions of degree at most $s(F)$:

$$f = \sum_{i=1}^{t} \gamma_i m_i$$

g is then defined by

$$\forall\ (v_1, \ldots, v_t) \in \mathcal{F}^{d(G)} \qquad g(v_1, \ldots, v_t) \doteq \sum_{i=1}^{t} \gamma_i v_i$$

It is clear that $g \circ \phi = f$, as desired.

Restriction property. Suppose G is given a feasible assignment, namely it is assigned a linear LDF g. The restriction of the assignment to F is the LDF $f = g \circ \phi$. Since ϕ is of degree $s(F)$ and g is linear, the degree of f is at most $s(F) \leq r(F)$, as required. ■

4.5. The Composition Procedure. We now turn to describe the composition procedure of the CR constructor algorithm. It takes as input a restricted LDF-reader $\mathcal{R}$, and generates a restricted LDF-reader $\mathcal{R}'$ with the same active degree parameters, but where the dimension of the active domains is constant.

Suppose an LDF-reader $\mathcal{R}$ is given, which evaluates a tuple $(u_1, \ldots, u_k)$ in a domain F. The composition procedure has two main steps: First it generates new LDF-readers using the SP constructor as a sub-procedure, and then it incorporates them into $\mathcal{R}$.

Uniformization. Recall that each local-reader L of $\mathcal{R}$ has variables from only one active domain, $\mathrm{Dom}_\star(L)$. Before applying the main two steps, it is convenient to make sure that all local-readers L in $\mathcal{R}$ have the same number of *active variables*, namely variables from $\mathrm{Dom}_\star(L)$. Denoting the maximal number of active variables in a local-reader of $\mathcal{R}$ by t, the composition procedure adds arbitrary variables to local-readers so that all have t active variables (the variables may be added anywhere in the local-reader, with zero coefficients).

Generating new LDF-readers. For each local-reader L in $\mathcal{R}$, the procedure now generates an LDF-reader denoted $\mathcal{R}_L$ as follows. If G is the active domain of L and $G[x_1], \ldots, G[x_t]$ are its active variables, then $\mathcal{R}_L$ is generated by calling the SP constructor (see Lemma Lemma 4.2) with parameters G and $(x_1, \ldots, x_t)$.

Domain incorporation. The composition procedure now incorporates the domains of the new LDF-readers into $\mathcal{R}$: The newly generated domains are added to the representation. The active domains of $\mathcal{R}$ cease to be active – the active domains of $\mathcal{R}'$ are the active domains of the newly generated LDF-readers.

Local-reader incorporation. In an active-feasible assignment for $\mathcal{R}'$, the active variables of $\mathcal{R}$-local-readers L are no longer promised to be assigned the evaluation of a single feasible LDF over $\mathrm{Dom}_\star(L)$. These variables are therefore replaced by the evaluators of local-readers of $\mathcal{R}_L$, which supposedly return evaluations of one of the (not many) permissible LDFs over $\mathrm{Dom}_\star(L)$.

For each pair of local-readers, L of $\mathcal{R}$ and M from $\mathcal{R}_L$, the composition procedure generates a local-reader of $\mathcal{R}'$, denoted by $L \circ M$, as follows. Let G denote the active domain of L, and let $G[x_1], \ldots, G[x_t]$ denote its active variables. To obtain $L \circ M$ each variable $G[x_i]$ in the evaluator or the local-test of L is replaced by the i'th evaluator of M, and then the local-test of M is added in conjunction to the local-test of L (where the $G[x_i]$'s have been replaced).

Properties of the composition procedure. We now analyze the properties of the composition procedure that are important for its application by the CR constructor – the time it takes, and the properties and parameters of the LDF-readers it generates. In the analysis we assume that the composition procedure is applied to LDF-readers where the the number of variables in each local-reader and the number of conjunctions in each local-test is bounded by a constant, since the CR constructor indeed applies it to such LDF-readers. Notice that under this assumption it is clear that the composition procedure generates LDF-readers where the number of variables in each local-reader and the number of conjunctions in each local-test is also bounded by a (different) constant.

Time. When applied to an LDF-reader $\mathcal{R}$, the composition procedure applies the SP constructor several times. Each call to the SP constructor takes time polynomial in $|\mathcal{F}|^{d_\star(\mathcal{R})}$, according to the definition of a constructor (note that this is polynomial the number of variables in each of the active domains of $\mathcal{R}$). Since the number of calls to the SP constructor equals the number of local-readers in $\mathcal{R}$, it follows that overall the composition procedure takes time polynomial in the size of $\mathcal{R}$.

Encoding-assignments. When the composition procedure is applied to an LDF-reader $\mathcal{R}$ that evaluates a tuple $(x_1, \ldots, x_k)$ in a domain F, the resulting structure $\mathcal{R}'$ has representation-variables and local-readers. To be a valid LDF-reader, we show that for every good assignment $\mathcal{A}$ for F there is an encoding-assignment with respect to $\mathcal{R}'$: First extend $\mathcal{A}$ to an encoding-assignment $\mathcal{A}'$ for $\mathcal{R}$. In particular $\mathcal{A}'$ assigns a good assignment to the active domain $\mathrm{Dom}_\star(L)$ of each local-reader L in $\mathcal{R}$. Then extend the assignment of each active domain $\mathrm{Dom}_\star(L)$ to an encoding-assignment with respect to $\mathcal{R}_L$. This obtains an assignment for all the variables of $\mathcal{R}'$. It is easy to verify that it is an encoding-assignment of $\mathcal{A}$ with respect to $\mathcal{R}'$.

Parameters of $\mathcal{R}'$. Given an LDF-reader $\mathcal{R}$, the composition procedure generates an LDF-reader $\mathcal{R}'$. The parameters of $\mathcal{R}'$ can be computed from the parameters of $\mathcal{R}$ according to the following composition lemma.

LEMMA 4.21 (composition). *Let $\mathcal{R}$ be a restricted (ρ, ϵ)-LDF-reader where $\epsilon^{3/4} > (r_\star(\mathcal{R})/|\mathcal{F}|)^{c_g} d_\star(\mathcal{R})$. Then the restricted LDF-reader $\mathcal{R}'$, generated from $\mathcal{R}$ by the composition procedure, has parameters $(\rho, O(\epsilon^{1/4}))$.*

Before the formal proof is given, we describe its main ideas. There are two types of ρ-erroneous local-readers $L \circ M$. One is where M is $\epsilon^{3/4}$-erroneous – this happens for at most an $O(\epsilon^{1/4})$ fraction of the local-readers since the $\mathcal{R}_L$'s are $(\epsilon^{3/4}, O(\epsilon^{1/4}))$-LDF-readers.

In case M is not erroneous, its evaluators yield evaluations of an $\epsilon^{3/4}$-permissible LDF f with respect to the assignment of $\text{Dom}_\star(L)$. $L \circ M$ is thus ρ-erroneous if and only if L remains ρ-erroneous when $\text{Dom}_\star(L)$ is assigned the feasible LDF f. Since for any active-feasible assignment for $\mathcal{R}$ at most an ϵ-fraction of its local-readers may be ρ-erroneous, and since the number of $\epsilon^{3/4}$-permissible LDFs for every domain is less than $2\epsilon^{-3/4}$, a counting argument implies that the fraction of local-readers $L \circ M$ where M is not erroneous is bounded by $2\epsilon^{-3/4} \cdot \epsilon = O(\epsilon^{1/4})$.

Proof of Lemma Lemma 4.21:

Fix an active-feasible assignment $\mathcal{A}'$ for the representation of $\mathcal{R}'$ and a parameter $\rho' \doteq \epsilon^{3/4}$, and let us divide the ρ-erroneous local-readers of $\mathcal{R}'$ into two sets according to ρ' – the *peripheral-erroneous* local-readers are the local-readers $L \circ M$ where M is ρ'-erroneous as a local-reader of $\mathcal{R}_L$, and the *core-erroneous* are those where M is not ρ'-erroneous. We bound the fraction of both types of local-readers by $O(\epsilon^{1/4})$.

Peripheral-erroneous local-readers. Since $\rho' > (r_\star(\mathcal{R})/|\mathcal{F}|)^{c_g} d_\star(\mathcal{R})$, Lemma Lemma 4.2 implies that every LDF-reader $\mathcal{R}_L$ generated by the composition procedure has parameters $(\rho', O((\rho')^{1/3}))$, so the fraction of ρ'-erroneous local-readers in it is $O((\rho')^{1/3}) = O(\epsilon^{1/4})$. Hence for every local-reader L of $\mathcal{R}$, the fraction of peripheral-erroneous local-readers among local-readers of the form $L \circ M$ is $O(\epsilon^{1/4})$, and therefore the overall fraction of peripheral-erroneous local-readers in $\mathcal{R}'$ is bounded by $O(\epsilon^{1/4})$ as desired.

We move to bound the fraction of core-erroneous local-readers. We first show that in such a local-reader $L \circ M$, L has to be erroneous as a local-reader of $\mathcal{R}$ with respect to a certain class of assignments, as explained below.

The assignments $\mathcal{A}(G, g)$ for $\mathcal{R}$. For an active domain G of $\mathcal{R}$ and an $[r(G), d(G)]$-LDF g, we define a class $\mathcal{A}(G, g)$ of assignments for $\mathcal{R}$, based on $\mathcal{A}'$. The elements of $\mathcal{A}(G, g)$ are the assignments for $\mathcal{R}$ that assign g to G, and that are equal to $\mathcal{A}'$ on all domains of $\mathcal{R}$ which are not active. Active domains of $\mathcal{R}$ other than G may be assigned arbitrarily. A local-reader L of $\mathcal{R}$ with $\text{Dom}_\star(L) = G$, may be either ρ-erroneous with respect to *all* assignments in $\mathcal{A}(G, g)$, or with respect to *none*, because the assignments in $\mathcal{A}(G, g)$ are all equal on the variables of L (recall that L cannot have variables from active domains other than G).

Consider a local-reader $L \circ M$ that is core-erroneous. The evaluators of M yield values consistent with an LDF g, which is ρ'-permissible with respect to the assignment of $G \doteq \mathrm{Dom}_\star(L)$. It follows that as a local-reader of $\mathcal{R}$, L is ρ-erroneous with respect to the assignments in $\mathcal{A}(G, g)$ – these assignments yield the same values for the variables of G as the evaluators of M, and give the same values as $\mathcal{A}'$ to all the other variables of L.

Core-erroneous local-readers. Let G be an active domain of $\mathcal{R}$. Denote by $\alpha(G, g)$ the fraction among $\mathcal{R}$-local-readers, of local-readers whose active domain is G and which are ρ-erroneous with respect to the assignments in $\mathcal{A}(G, g)$. Denote by $\alpha(G)$ the maximum over all $\alpha(G, g)$.

Let $\mathcal{A}$ be the assignment obtained from $\mathcal{A}'$ by assigning to each active domain G of $\mathcal{R}$ an LDF g such that $\alpha(G, g)$ is maximized. Then for every G, the fraction of $\mathcal{R}$-local-readers whose active domain is G, and which are ρ-erroneous with respect to $\mathcal{A}$ is $\alpha(G)$. The parameters of $\mathcal{R}$ imply that the total fraction of ρ-erroneous local-readers is bounded by ϵ, hence $\sum_G \alpha(G) \le \epsilon$.

For an active domain G of $\mathcal{R}$ we denote by $\gamma(G)$ the fraction of local-readers L in $\mathcal{R}$ whose active domain is G, and for which there exists a local-reader M in $\mathcal{R}_L$ where $L \circ M$ is core-erroneous. We have seen that for an $\mathcal{R}$-local-reader L to be accounted in $\gamma(G)$, it must be ρ-erroneous with respect to the assignments in $\mathcal{A}(G, g)$, for some ρ'-permissible g. $\gamma(G)$ is therefore bounded by the sum $\sum \alpha(G, g)$ taken over all permissible LDFs g, and so by $\alpha(G)$ times the number of ρ'-permissible LDFs. By Proposition Proposition 3.5 the number of ρ'-permissible LDFs is less than $2/\rho'$, hence $\gamma(G) < (2/\rho')\alpha(G)$, and we obtain that

$$\sum_G \gamma(G) < (2/\rho') \sum_G \alpha(G) \le 2\epsilon/\rho' = 2\epsilon^{1/4}$$

Namely the fraction of $\mathcal{R}$-local-readers L for which there exists a core-erroneous local-reader $L \circ M$ is bounded by $O(\epsilon^{1/4})$.

We show that this also bounds the total fraction of core-erroneous local-readers: Note that there is the same number of local-readers in every LDF-reader of the form $\mathcal{R}_L$ – this follows from the definition of a constructor, together with the fact that all active domains of $\mathcal{R}$ have the same degree parameters. Hence for each local-reader L of $\mathcal{R}$ there is the same number of local-readers of the form $L \circ M$ in $\mathcal{R}'$. ■

4.6. The CR Constructor. It is now the time to describe the actual CR constructor, proving the Composition-Recursion Constructor Lemma (Lemma Lemma 3.3). Let F be a domain, and let $(x_1, \ldots, x_k)$ be a k-tuple of points in $\mathcal{F}^{d(F)}$ (where, as in Lemma Lemma 3.3, k is a constant). We assume, under the notation as specified in Lemma Lemma 3.3, that $d(F) = O(\log^{1-\beta} n)$, $s(F) \le |\mathcal{F}|^{c_1}$, and $r(F) \ge |\mathcal{F}|^{c_2}$. For simplicity we reset $s(F)$ and $r(F)$ so that the latter inequalities hold as equalities – note that a (ρ, ϵ)-LDF-reader with respect to the new degree parameters remains a (ρ, ϵ)-LDF-reader if $s(F)$ is reduced and $r(F)$ is increased to their original values.

The CR constructor generates an (unrestricted) LDF-reader $\mathcal{R}$ evaluating $(x_1, \ldots, x_k)$ in F. First, it generates a sequence $\mathcal{R}_0, \ldots, \mathcal{R}_K$, where $K = O(\frac{1}{1-\beta})$ is a constant that will be chosen later, of restricted LDF-readers. The transformation of each $\mathcal{R}_i$ into $\mathcal{R}_{i+1}$ is accomplished in two steps. At first a restricted LDF-reader $\mathcal{R}'_i$ is generated by applications of an extension-procedure to the active domains of $\mathcal{R}_i$, as described in the extension proposition (Proposition Proposition 4.18). The degree parameters of $\mathcal{R}'_i$ are decreased with respect to $\mathcal{R}_i$ but the dimension is increased. $\mathcal{R}_{i+1}$ is then generated by applying the composition procedure to $\mathcal{R}'_i$, thus the degree parameters remains the same while the active dimension parameter becomes constant. Finally $\mathcal{R}_K$ has a constant active dimension and both of its active degree parameters are 1, hence in a good or active-feasible assignment each active domain of $\mathcal{R}_K$ is assigned a constant-dimensional linear function. The final unrestricted LDF-reader is obtained by replacing each active domain of $\mathcal{R}_K$ by a constant number of variables that represent the coefficients of a linear function over it.

We now fully describe the generation of the sequence $\mathcal{R}_0, \ldots, \mathcal{R}_K$, and the transformation of $\mathcal{R}_K$ into $\mathcal{R}$. We then show that the CR constructor has the properties required by Lemma Lemma 3.3.

Generating $\mathcal{R}_0$. To generate the first restricted LDF-reader, $\mathcal{R}_0$, the CR constructor applies the SP constructor to the domain F and the tuple $(x_1, \ldots, x_k)$.

Generating $\mathcal{R}_1, \ldots, \mathcal{R}_{K-1}$. From $\mathcal{R}_0$ the CR constructor continues to iteratively generate restricted LDF-readers as follows. Having generated $\mathcal{R}_i$, the constructor transforms it into a restricted LDF-reader $\mathcal{R}'_i$ by applying the power-substitution extension-procedure to each active domain of $\mathcal{R}_i$ with parameter

$$b = \max\left\{ \, (s_\star(\mathcal{R}_i) + 1)^{1/\log^{1-\beta} n} \; , \; 2 \, \right\}$$

and taking these extensions to be the active domains of $\mathcal{R}'_i$. Note that b is chosen to have the smallest value such that the dimension of the domains generated by the extension-procedure is at most $\Theta(\log^{1-\beta} n)$ – a larger dimension would yield domains of super-polynomial size. The constructor then generates $\mathcal{R}_{i+1}$ by applying the composition procedure to $\mathcal{R}'_i$. The CR constructor iteratively generates LDF-readers as described above until finally an LDF-reader $\mathcal{R}_{K-1}$ is generated such that

$$\binom{s_\star(\mathcal{R}_{K-1}) + d_\star(\mathcal{R}_{K-1})}{d_\star(\mathcal{R}_{K-1})} \leq \log^{1-\beta} n$$

As proven below, this occurs for a constant K.

Generating $\mathcal{R}_K$. The transformation of $\mathcal{R}_{K-1}$ into $\mathcal{R}_K$ is carried similarly to the previous transformations described above, only that $\mathcal{R}'_{K-1}$ is generated using the linearization extension-procedure instead of the power-substitution extension-procedure. Note that for the linearization extension-procedure to be applicable the active lower-degree parameter of $\mathcal{R}_{K-1}$ must not be greater than its active upper-degree. We show below that this indeed holds.

Generating $\mathcal{R}$. The constructor now transforms $\mathcal{R}_K$ into the final CR LDF-reader. Having used the linearization extension-procedure to produce $\mathcal{R}'_{K-1}$, we gather that the active lower-degree of $\mathcal{R}_K$ (which equals that of $\mathcal{R}'_{K-1}$) is 1. Its active dimension, $d \doteq d_\star(\mathcal{R}_K)$, is constant since $\mathcal{R}_K$ is generated by the composition procedure. A good assignment to an active domain G of $\mathcal{R}_K$ is thus a *linear* LDF f, that can be represented using a constant number of coefficient γ_i by

$$\forall\ (u_1, \ldots, u_d) \in \mathcal{F}^d \qquad g(u_1, \ldots, u_d) \doteq \gamma_0 + \sum_{i=1}^{d} \gamma_i u_i$$

The CR constructor adds $d+1$ variables to the representation, $G_0, \ldots, G_d$, for each active domain G of $\mathcal{R}_K$, such that an encoding assignment would assign $G_i = \gamma_i$ for each i. It then goes over all the local-readers and replaces every term $G[(u_1, \ldots, u_d)]$, where G is an active-LDF, by $G_0 + \sum_{i=1}^{d} G_i u_i$. It is now possible to deactivate or even remove the active domains altogether (their variables no longer appear anywhere), thus completing the generation of $\mathcal{R}$.

4.7. The CR Constructor Works. Below it is proven that the CR constructor above has the properties stated in Lemma Lemma 3.3. We show that it stops after a constant number of iterations as stated above, and that it takes polynomial time. It is then shown that although each transformation of $\mathcal{R}_i$ into $\mathcal{R}_{i+1}$ consumes some of the lower-degree to upper-degree gap, the active upper-degree of $\mathcal{R}_{K-1}$ is not smaller than the active lower-degree (hence linearization extension-procedure is correctly used by the CR constructor). We conclude by showing that for an appropriate constant $c > 0$, the constructor generates $(\rho, O(\rho^c))$-LDF-readers for all ρ's such that $\rho > (r/|\mathcal{F}|)^{c_g} d$.

First of all observe that as noted in the description of the properties of the composition procedure, for every constant i both the number of variables and the number of conjunctions in each local-reader are bounded by a constant.

The number of iterations is constant. It should be shown that for some constant $K = O(\frac{1}{1-\beta})$, the parameters of the $(K-1)$'th element in the sequence $\mathcal{R}_0, \mathcal{R}_1, \ldots$ of LDF-readers satisfy

$$\binom{s_\star(\mathcal{R}_{K-1}) + d_\star(\mathcal{R}_{K-1})}{d_\star(\mathcal{R}_{K-1})} \leq \log^{1-\beta} n \tag{4.22}$$

To see this we examine the parameters of the LDF-readers in the sequence.

Parameters of the $\mathcal{R}_i$'s. Consider an LDF-reader $\mathcal{R}_i$ in the sequence, and assume $s_\star(\mathcal{R}_i) + 1 > 2^{\log^{1-\beta} n}$. The power-substitution extension-procedure is applied to its active domains using the parameter $b = (s_\star(\mathcal{R}_i)+1)^{1/\log^{1-\beta} n}$, hence the parameter t used within the extension-procedure is $t = \log^{1-\beta} n$ (see Proposition Proposition 4.19). Since the active dimension of $\mathcal{R}_i$ is constant, Proposition Proposition 4.19 implies that

$$s_\star(\mathcal{R}_{i+1}) = s_\star(\mathcal{R}'_i) = d_\star(\mathcal{R}_i) t(b-1) = polylog(n) s_\star(\mathcal{R}_i)^{1/\log^{1-\beta} n} \tag{4.23}$$

As $\mathcal{R}_0$ is generated by the SP constructor, its active lower-degree parameter equals $s(F)$, so $s_\star(\mathcal{R}_0) = |\mathcal{F}|^{c_1} = 2^{\Theta(\log^\beta n)}$. By inductively using Equation (4.23) one easily sees that as long as $\beta - i(1-\beta) > 0$,

$$s_\star(\mathcal{R}_i) = 2^{\Theta(\log^{\beta - i(1-\beta)} n)} \tag{4.24}$$

(the poly-logarithmic factor is absorbed in the exponent).

Parameters of $\mathcal{R}_{i_o}$. Fix $i_o \doteq \lceil \beta/(1-\beta) \rceil$, and note that it is constant (it depends only on β, which remains constant throughout the proof). We have

$$1 - \beta \geq \beta - (i_o - 1)(1-\beta) > 0$$

hence by Equation (4.24),

$$s_\star(\mathcal{R}_{i_o-1}) = 2^{\Theta(\log^{\beta-(i_o-1)(1-\beta)} n)}$$

$\mathcal{R}'_{i_o-1}$ is generated by applying the extension-procedure with parameter

$$b = \max\left\{ \left(2^{\Theta(\log^{\beta - i_o(1-\beta)} n)} \right) \,,\ 2 \right\} = O(1)$$

since $\beta - i_o(1-\beta) \leq 0$. The parameter t used is poly-logarithmic in n, specifically $t = O(\log^{\beta-(i_o-1)(1-\beta)} n) \leq O(\log^{1-\beta} n)$. It hence follows from Proposition Proposition 4.19 that $s_\star(\mathcal{R}_{i_o}) = s_\star(\mathcal{R}'_{i_o-1})$ is poly-logarithmic in n.

Parameters of $\mathcal{R}_{i_o+1}$. The power-substitution extension-procedure is applied with parameter $b = 2$ to generate $\mathcal{R}'_{i_o}$ from $\mathcal{R}_{i_o}$. Since $s_\star(\mathcal{R}_{i_o})$ is poly-logarithmic, t is poly-log-logarithmic in n, and therefore $s_\star(\mathcal{R}'_{i_o}) = s_\star(\mathcal{R}_{i_o+1})$ is also poly-log-logarithmic in n. Since $d_\star(\mathcal{R}_{i_o+1})$ is constant, it follows that

$$\binom{s_\star(\mathcal{R}_{i_o+1}) + d_\star(\mathcal{R}_{i_o+1})}{d_\star(\mathcal{R}_{i_o+1})} \leq \log^{1-\beta} n \tag{4.25}$$

Setting $K \doteq i_o + 2$, we have that $K = O(1/(1-\beta))$ is constant and that Inequality (4.22) clearly holds for $\mathcal{R}_{K-1} = \mathcal{R}_{i_o+1}$.

The lower – upper-degree gap remains. Going from $\mathcal{R}_{K-1}$ to $\mathcal{R}'_{K-1}$, the CR constructor applies the linearization extension-procedure to each active domain of $\mathcal{R}_{K-1}$. This procedure is only applicable to domains where the lower-degree is not greater than the upper-degree, hence we must show that $s_\star(\mathcal{R}_{K-1}) \leq r_\star(\mathcal{R}_{K-1})$.

Let us compute how $s_\star(\mathcal{R}_{i+1})/r_\star(\mathcal{R}_{i+1})$ behaves with respect to $s_\star(\mathcal{R}_i)/r_\star(\mathcal{R}_i)$ for $0 \leq i < K-1$. Let b_i denote the b-parameter with which the power-substitution extension-procedure is applied to the active domains of $\mathcal{R}_i$ to obtain $\mathcal{R}'_i$, and let t_i denote the associated t-parameter. According to Proposition Proposition 4.19,

$$\frac{s_\star(\mathcal{R}_{i+1})}{r_\star(\mathcal{R}_{i+1})} = \frac{s_\star(\mathcal{R}'_i)}{r_\star(\mathcal{R}'_i)} \leq \frac{s_\star(\mathcal{R}_i) \cdot d_\star(\mathcal{R}_i) t_i (b_i - 1)}{r_\star(\mathcal{R}_i)} = O\left(\frac{s_\star(\mathcal{R}_i)}{r_\star(\mathcal{R}_i)} \cdot t_i b_i \right)$$

hence the ratio between the active lower-degree and the active upper-degree is consumed by a factor of up to $O(t_i b_i)$ in the transition from $\mathcal{R}_i$ to $\mathcal{R}_{i+1}$.

Let us bound $t_i b_i$. By the choice of the parameters b_i it follows that $t_i \leq \log^{1-\beta} n$ for all i. According to the above computations of $s_\star(\mathcal{R}_i)$, for $0 \leq i < K-3$

$$b_i = (s_\star(\mathcal{R}_i)+1)^{1/\log^{1-\beta} n} = \left(2^{\Theta(\log^{\beta - i(1-\beta)} n)}\right)^{1/\log^{1-\beta} n} = 2^{\Theta(\log^{\beta-(i+1)(1-\beta)} n)} = 2^{O(\log^{\beta-(1-\beta)} n)}$$

and therefore $t_i b_i = 2^{O(\log^{\beta-(1-\beta)} n)}$. For $i = K-3$ or $i = K-2$, $b_i = O(1)$ so in these cases $t_i b_i$ is poly-logarithmic in n.

The initial lower-degree upper-degree fraction is $s_\star(\mathcal{R}_0)/r_\star(\mathcal{R}_0) = |\mathcal{F}|^{c_1-c_2} = 2^{-\Theta(\log^\beta n)}$. This fraction is consumed in each of the constant number of iterations by either $2^{O(\log^{\beta-(1-\beta)} n)}$ or a poly-logarithm, hence $s_\star(\mathcal{R}_{K-1})/r_\star(\mathcal{R}_{K-1}) = 2^{-\Theta(\log^\beta n)}$ and in particular $s_\star(\mathcal{R}_{K-1}) < r_\star(\mathcal{R}_{K-1})$, as desired.

The CR constructor takes polynomial time. We need to show that the CR constructor takes polynomial time in $|\mathcal{F}|^{d(F)}$. Since

$$|\mathcal{F}|^{d(F)} = \left(2^{\log^\beta n}\right)^{\Theta(\log^{1-\beta} n)} = n^{\Theta(1)}$$

this is equivalent to showing that it takes polynomial time in n. The proof is by showing that the generation of each LDF-reader $\mathcal{R}_i$ in the sequence $\mathcal{R}_0, \mathcal{R}'_0, \mathcal{R}_1, \mathcal{R}'_1, \ldots, \mathcal{R}_K$ takes polynomial time in n and in the size of the predecessor of $\mathcal{R}_i$ (clearly the time it takes to generate the final LDF-reader from $\mathcal{R}_K$ is polynomial in the size of $\mathcal{R}_K$). This implies that the CR constructor indeed takes polynomial time.

Generating $\mathcal{R}_0$. The CR constructor generates $\mathcal{R}_0$ using the SP constructor, which does take time polynomial in $|\mathcal{F}|^{d(F)}$.

Generating $\mathcal{R}_i$ for $0 < i \leq K$. The CR constructor generates $\mathcal{R}_i$ by applying the composition procedure to $\mathcal{R}'_{i-1}$. As mentioned in Subsection Section 4.5, this takes polynomial time in the size of $\mathcal{R}'_{i-1}$.

Generating $\mathcal{R}'_i$ for $i < K-1$. The CR constructor generates $\mathcal{R}'_i$ by applying the power-substitution extension-procedure to all active domains of $\mathcal{R}_i$. The time it takes is bounded by the size of $\mathcal{R}_i$ times the time needed for each application of the extension-procedure. By the definition of extension-procedures, each such application takes polynomial time in $|\mathcal{F}|^{d_\star(\mathcal{R}'_i)}$. According to Proposition Proposition 4.19, $d_\star(\mathcal{R}'_i) = d_\star(\mathcal{R}_i)t_i = O(t_i) \leq O(\log^{1-\beta} n)$ where t_i is as denoted in the degree-gap computation, hence $|\mathcal{F}|^{d_\star(\mathcal{R}'_i)} = n^{O(1)}$. Therefore each application of the extension procedure takes polynomial time in n, as needed.

Generating $\mathcal{R}'_{K-1}$. The difference between the generation of $\mathcal{R}'_{K-1}$ and that of the other $\mathcal{R}'_i$'s, is that the linearization extension-procedures is applied to each active domain instead of the power-substitution extension-procedure. Each such application still takes polynomial time in $|\mathcal{F}|^{d_\star(\mathcal{R}_{K-1})}$ but here $d_\star(\mathcal{R}'_{K-1})$ is calculated according to Proposition Proposition 4.20,

$$d_\star(\mathcal{R}'_{K-1}) = \binom{s_\star(\mathcal{R}_{K-1}) + d_\star(\mathcal{R}_{K-1})}{d_\star(\mathcal{R}_{K-1})} \leq \log^{1-\beta} n$$

$|\mathcal{F}|^{d_\star(\mathcal{R}_{K-1})}$ is therefore still polynomial.

(ρ, ϵ)-parameters of the CR constructor. We now show that $\mathcal{R}$ has parameters $(\rho, O(\rho^{4^{-K}/3}))$ for all ρ's that satisfy $\rho > (r/|\mathcal{F}|)^{c_g} d$. We first prove by induction that for all i, $\mathcal{R}_i$ is a restricted $(\rho, O(\rho^{4^{-i}/3}))$-LDF-reader: For $\mathcal{R}_0$ it follows directly from Lemma Lemma 4.2. Assume now that $\mathcal{R}_{i-1}$ is a restricted $(\rho, O(\rho^{4^{-i+1}/3}))$-LDF-reader. The extension proposition (Proposition Proposition 4.18) implies that $\mathcal{R}'_{i-1}$ has the same parameters. Since $\mathcal{R}_i$ is generated from $\mathcal{R}'_{i-1}$ by the composition procedure, Lemma Lemma 4.21 yields that $\mathcal{R}_i$ is a restricted $(\rho, O(\rho^{4^{-i}/3}))$-LDF-reader, as desired (the O notation here is justified since we only make a constant number of steps in the induction). Note that the requirement over ϵ in Lemma Lemma 4.21 holds here, since we apply it with $\epsilon^{3/4} = O(\rho^{4^{-i}}) \geq \rho > (r/|\mathcal{F}|)^{c_g} d$.

By the above induction, $\mathcal{R}_K$ is a restricted $(\rho, O(\rho^c))$-LDF-reader for $c \doteq 4^{-K}/3$. To show that $\mathcal{R}$ has the same parameters, we define for each assignment $\mathcal{A}$ of $\mathcal{R}$ an *active-feasible* assignment $\mathcal{A}'$ for $\mathcal{R}_K$, such that an $\mathcal{R}_K$-local-reader is ρ-erroneous with respect to $\mathcal{A}'$ if and only if the $\mathcal{R}$-local-reader generated from it is ρ-erroneous with respect to $\mathcal{A}$. This would imply that $\mathcal{R}$ has the same (ρ, ϵ) parameters as $\mathcal{R}_K$.

$\mathcal{A}'$ differs from $\mathcal{A}$ only on active domains of $\mathcal{R}_K$ – for an active domain G and a variable $G[(u_1, \ldots, u_d)]$ in it we define

$$\mathcal{A}'(\, G[(u_1, \ldots, u_d)] \,) \doteq \sum_{i=1}^{d} \mathcal{A}(G_i) u_i$$

where the G_i's are the new variables added in the generation of the final CR constructor. $\mathcal{A}'$ assigns to each active domain G a linear LDF represented by the assignment of the G_i's, and is hence active-feasible. It is clear from the construction of $\mathcal{R}$ that a local-reader of $\mathcal{R}_K$ is ρ-erroneous with respect to $\mathcal{A}'$ if and only if the $\mathcal{R}$-local-reader obtained from it is ρ-erroneous with respect to $\mathcal{A}$.

5. The Sum-Check

In this section we prove the Sum-Check Lemma, Lemma Lemma 3.1. A reduction algorithm is shown that given a system Ψ of n quadratic-equations, with up to n variables in each equation, generates a system Ψ_{sc} whose variables belong to domains, and where every equation accesses only a constant number of variables. The reduction of Ψ into Ψ_{sc} is gap-preserving with respect to certain restricted sets of assignments. That is, if Ψ is completely satisfiable then Ψ_{sc} can be completely satisfiable by a **good** assignment to its domains (see Definition Definition 2.6); and if there is no assignment for Ψ that satisfies more than a $\frac{2}{|\mathcal{F}|}$ fraction of its equations, then no **feasible** assignment for Ψ_{sc} can satisfy more than a $\frac{2}{|\mathcal{F}|}$ fraction of its equations as well.

The reduction begins with the given system $\Psi_0 \doteq \Psi$, and puts it through a constant number ($O(\frac{1}{1-\beta})$) of transformations, obtaining a sequence $\Psi_0, \Psi_1, \ldots, \Psi_l$ of equation-systems, where the final system, Ψ_{sc}, is generated by a small alteration of Ψ_l. The number of variables in each equation decreases gradually throughout the sequence from up to n in Ψ_0, to a constant in Ψ_l.

We continue as follows. The next subsection defines the structure of a restricted equation-system. The transformation of each restricted equation-system Ψ_i into Ψ_{i+1} (the transformation of Ψ_0 into Ψ_1 is an exception) is performed by an algorithm that is described in Subsection Section 5.2. This algorithm is used for the transformation of each intermediate system into the next, however a crucial part of the algorithm, called the *representation-procedure*, varies in different transformations. A representation-procedure is defined in Subsection Section 5.2, and Subsection Section 5.3 describes the properties of the different representation-procedures used, and of the algorithm which transforms Ψ_0 into Ψ_1. The complete reduction of Ψ into Ψ_{sc} is finally described in Subsection Section 5.4.

The following subsections are dedicated to proving the correctness of the reduction (Subsection Section 5.5), and to the description and correctness proofs of the product-check and the representation-procedures used (Subsections Section 5.6, Section 5.7, Section 5.8, and Section 5.9).

5.1. Restricted Equation-Systems.

All the systems Ψ_i, $i = 1, 2, \ldots, l$, in the sequence generated by the reduction algorithm have a similar structure. The following is an exact definition thereof.

Definition 5.1 (restricted equation-systems). *A* restricted equation-system Ψ, *is a quadratic equation-system where some domains are considered* active. *The dimension, and the upper and lower-degree parameters of the active domains must all be the same. These parameters are called the active dimension, active upper-degree and active lower-degree of Ψ, and are denoted by $d_\star(\Psi)$, $r_\star(\Psi)$ and $s_\star(\Psi)$ respectively.*

Each equation $\psi \in \Psi$ is written in the form "$\psi_\star = \psi_{\mathbf{c}}$". $\psi_\star$ is called the active part of ψ and $\psi_{\mathbf{c}}$ is called the core of ψ. While $\psi_{\mathbf{c}}$ may contain any variable of Ψ, including variables from any active domains, and can have quadratic as well as linear terms, $\psi_\star$ contains only linear terms and the variables in it must all be from one active domain, called the active domain of ψ and denoted by $\mathrm{Dom}_\star(\psi)$. *The variables that appear in $\psi_\star$ are called the active variables of ψ.*

Note that here the meaning of active domain is different, yet similar, to the one used in Section Section 4.

The equations in all intermediate equation-systems will have only a constant number of variables in their core – all other variables appear in the active part of the equations. It is hence useful to denote the number of variables in the core of an equation and the number of active variables by different names.

DEFINITION 5.2 (active and core-dependency). *Let Ψ be a restricted equation-system. The* active-dependency *of an equation $\psi \in \Psi$, denoted by $\mathrm{D}_\star(\psi)$, is defined as the number of variables in $\psi_\star$. The* core-dependency *of ψ, $\mathrm{D}_{\mathbf{c}}(\psi)$, is defined as the number of variables in $\psi_{\mathbf{c}}$. The active-dependency of Ψ, denoted by $\mathrm{D}_\star(\Psi)$, is the maximum of $\mathrm{D}_\star(\psi)$ over all equations $\psi \in \Psi$. The core-dependency of Ψ is denoted $\mathrm{D}_{\mathbf{c}}(\Psi)$, and is defined similarly.*

As mentioned above, the core-dependency parameters of all the restricted equation-systems in the sequence $\Psi_1, \ldots, \Psi_l$ are constants. The active-dependency parameter is decreased gradually until it becomes constant in Ψ_l. The total number of variables in an equation of Ψ_l is therefore constant, as required by Lemma Lemma 3.1 (this property is preserved in the final transition from Ψ_l to Ψ_{sc}).

5.2. The Main Transformation-Scheme. The transformation of each restricted equation-system Ψ_i into the next is done by substituting each equation ψ of Ψ_i by a "representation" containing several new equations. The transformations of Ψ_i into Ψ_{i+1} where $i = 1, \ldots, l-1$ are in fact of a more specific structure, and are carried out by the *system-representation algorithm.* An important part of this algorithm is the application of a *representation-procedure* to each equation in the system (a different representation-procedure is used for different transformations). This procedure is applied to each of the equations in the system, replacing it with a set of conjunctions over both old and newly generated variables. The conjunction will maintain consistency between the values of the new variables and that of the old ones, and thus consistency between the equations of the system. We now define a representation-procedure, and then describe the system-representation algorithm.

Representation-procedures. A representation-procedure is an algorithm that is applied to an equation ψ and produces a set $\mathcal{E}_\psi$ of *conjunctions* of equations, and a new domain denoted by $\mathrm{Dom}_\star(\mathcal{E}_\psi)$ (the new conjunctions have variables from $\mathrm{Dom}_\star(\mathcal{E}_\psi)$).

The conjunctions of $\mathcal{E}_\psi$ represent ψ in the sense that they are only satisfied by extensions to $\mathrm{Dom}_\star(\mathcal{E}_\psi)$, of assignments that also satisfy ψ – feasible assignments which do not satisfy ψ will satisfy almost none of the conjunctions in $\mathcal{E}_\psi$.

For $i = 1, \ldots, l$, Ψ_{i+1} is obtained from Ψ_i by applying a representation-procedure to each equation $\psi \in \Psi_i$, generating a system Ψ'_i of conjunctions which is the union of the sets $\{\mathcal{E}_\psi\}_{\psi \in \Psi_i}$. Ψ_{i+1} is then generated by replacing the conjunctions with equations as in Proposition Proposition 3.7. If the representation-procedure generates conjunctions with a small number of variables, then the dependency parameter of Ψ_{i+1} will be smaller than that of Ψ_i (eventually the dependency is constant). Also, the active-domains of Ψ_{i+1} are set to be the new domains generated by the representation-procedure, and hence the active parameters of Ψ_{i+1} are changed. Actually the reduction generates domains with different parameters, contrary to a requirement in Lemma Lemma 3.1. This is rectified by applying a simple technical method at the end of the reduction, that makes all the domains uniform.

An $[s,d]$-representation-procedure. An $[s,d]$-representation-procedure is an algorithm $\mathbf{A}$ that receives as input an equation ψ in a restricted equation-system Ψ, and generates a set $\mathcal{E}_\psi$ of conjunctions of quadratic-equations that "represent" ψ. It also generates a new domain denoted $\mathrm{Dom}_\star(\mathcal{E}_\psi)$ – the conjunctions in $\mathcal{E}_\psi$ may have variables from $\mathrm{Dom}_\star(\mathcal{E}_\psi)$ in addition to any variables of Ψ. For a conjunction $\chi \in \mathcal{E}_\psi$ we define the active domain of χ to be $\mathrm{Dom}_\star(\chi) \doteq \mathrm{Dom}_\star(\mathcal{E}_\psi)$. The variables of χ that are from the domain $\mathrm{Dom}_\star(\chi)$ are called the active variables of χ.

The parameters $r(\mathrm{Dom}_\star(\psi))$, s and d determine the parameters of the new domain, namely $\mathrm{Dom}_\star(\mathcal{E}_\psi)$ must satisfy $r(\mathrm{Dom}_\star(\mathcal{E}_\psi)) = r(\mathrm{Dom}_\star(\psi))$, $s(\mathrm{Dom}_\star(\mathcal{E}_\psi)) = s$, and $d(\mathrm{Dom}_\star(\mathcal{E}_\psi)) = d$. The running time of $\mathbf{A}$ should be polynomial in $|\mathcal{F}|^d = |\mathrm{Dom}_\star(\mathcal{E}_\psi)|$ and the size of ψ.

Extension and restriction properties. For the conjunctions in $\mathcal{E}_\psi$ to properly represent ψ, it is required that $\mathbf{A}$ has the following extension and restriction properties:

- Extension Property: For every good assignment $\mathcal{A}$ for Ψ that satisfies ψ there should be an s-degree LDF such that if it is assigned to $\mathrm{Dom}_\star(\mathcal{E}_\psi)$, all the conjunctions in $\mathcal{E}_\psi$ are satisfied.

- Restriction Property: If a feasible assignment for Ψ and for $\mathrm{Dom}_\star(\mathcal{E}_\psi)$ satisfies at least an $|\mathcal{F}|^{-1/2}$ fraction of the conjunctions in $\mathcal{E}_\psi$, then ψ is satisfied as well.

Uniformity. It is required that the parameters s and d be functions of Ψ alone, so that the parameters of $\mathrm{Dom}_\star(\mathcal{E}_\psi)$ are the same for all equations $\psi \in \Psi$ to which $\mathbf{A}$ is applied. The number of conjunctions in $\mathcal{E}_\psi$ should also be independent of ψ (and be a function of Ψ alone). The number of equations in each conjunction of $\mathcal{E}_\psi$ should all be the same, and they must be independent of ψ as well. In addition we require that the number of equations in each conjunction is bounded by $O(d)$.

Conjunction-structure. The conjunctions of $\mathcal{E}_\psi$ should have the following structure. $\psi_{\mathbf{c}}$ may appear at most once in at most one equation of each conjunction $\chi \in \mathcal{E}_\psi$. Except for the terms in this copy of $\psi_{\mathbf{c}}$, all terms must be linear and the number of terms not from the domain $\mathrm{Dom}_\star(\mathcal{E}_\psi)$ must be bounded by a constant (that is, a number which is independent of ψ and Ψ).

The system-representation algorithm. Let us now describe how a restricted equation-system Ψ_i is transformed into Ψ_{i+1} using a representation-procedure $\mathbf{A}$.

1. First, $\mathbf{A}$ is applied to every equation $\psi \in \Psi_i$.

2. A system Ψ_i' of conjunctions is constructed by taking the union of the sets $\{\mathcal{E}_\psi\}_{\psi \in \Psi_i}$. Note that the number of equations in each conjunction of Ψ_i' is the same, and that each equation of Ψ_i results in the same number of conjunctions in Ψ_i'.

3. Ψ_{i+1} is generated by replacing each conjunction $\chi \in \Psi_i'$ by all linear-combinations of its equations (with multiplicities, if the same equation occurs more than once). The active domain of each equation is set to be the same as that of the conjunction from which it originated. The active variables of such an equation are thus the same as the active variables of the originating conjunction.

4. For each equation $\xi \in \Psi_{i+1}$, $\xi_\star$ and $\xi_{\mathbf{c}}$ are defined as follows. The variables from $\mathrm{Dom}_\star(\xi)$ are moved to the left-hand side of the equation, and the other variables to the right-hand side (it follows from the properties of **A** that variables from $\mathrm{Dom}_\star(\xi)$ only appear in linear terms). The left-hand side of ξ is then defined to be the active part $\xi_\star$ of ξ, and the right-hand side is set to be its core, $\xi_{\mathbf{c}}$.

Let us examine some of the properties of the system-representation algorithm.

The parameters of Ψ_{i+1}. Since the upper-degree parameter of the domains produced by the representation-procedure **A** are the same as the active upper-degree of Ψ_i, we have $r_\star(\Psi_{i+1}) = r_\star(\Psi_i)$. If **A** is an $[s,d]$-representation-procedure, then the active domains of Ψ_{i+1} will all have lower-degree s and dimension d, hence $s_\star(\Psi_{i+1}) = s$ and $d_\star(\Psi_{i+1}) = d$.

Time. The system-representation algorithm takes polynomial time in the size of Ψ_i and $|\mathcal{F}|^{d_\star(\Psi_{i+1})}$. Especially note that step 3 is applicable in polynomial time in $|\mathcal{F}|^{d_\star(\Psi_{i+1})}$ and in the size of Ψ_i – the uniformity property requires that the number of equations in each conjunction be bounded by $O(d_\star(\Psi_{i+1}))$, hence the number of equations produced for each conjunction is $|\mathcal{F}|^{O(d_\star(\Psi_{i+1}))}$.

Core-dependency. Note that the core-dependency of Ψ_{i+1} is larger by at most a constant than that of Ψ_i. Consider an equation $\psi' \in \Psi_{i+1}$ whose origin is an equation $\psi \in \Psi_i$. It has the variables of $\psi_{\mathbf{c}}$, and at most a constant number of variables not from $\mathrm{Dom}_\star(\mathcal{E}_\psi)$. The other variables are from with $\mathrm{Dom}_\star(\mathcal{E}_\psi)$, and are hence active, so the core-dependency of ψ' is larger by only a constant than that of ψ.

The gap. The extension and restriction properties of the representation-procedure **A** that is used by the system-representation algorithm, ensure that the fraction of satisfiable equations in Ψ_i with respect to good or feasible assignments is close to the satisfiable fraction in Ψ_{i+1}. Here is a precise definition of this property.

Definition 5.3 (gap-preserving algorithm). *An algorithm that transforms a given equation-system Ψ_i into an equation-system Ψ_{i+1} is said to be gap-preserving if it has the following properties:*

- *Completeness: If Ψ_i can be completely satisfied by a good assignment then so can Ψ_{i+1}.*

- *Soundness: If a feasible assignment for Ψ_{i+1} can satisfy a γ-fraction of its equations, then there exists a feasible assignment for Ψ_i satisfying at least a $\gamma - O(|\mathcal{F}|^{-1/2})$ fraction of its equations.*

Note that even in a gap-preserving algorithm, the gap is actually consumed by an $O(|\mathcal{F}|^{-1/2})$ fraction. The reduction deals with this by applying a simple gap amplification technique in the transformation from the system Ψ_l into the final system Ψ_{sc}.

Proposition 5.4. *The system-representation algorithm is gap-preserving.*

Proof. Assume that the system-representation algorithm is applied to a restricted equation-system Ψ_i using a representation-procedure $\mathbf{A}$, and outputs Ψ_{i+1}.

The completeness property is implied from the extension property of $\mathbf{A}$ as follows. Suppose Ψ_i is satisfied by an assignment $\mathcal{A}$. According to the extension property, $\mathcal{A}$ can be extended to assign for each ψ an $s(\mathrm{Dom}_\star(\mathcal{E}_\psi))$ degree LDF to $\mathrm{Dom}_\star(\mathcal{E}_\psi)$ such that the conjunctions in $\mathcal{E}_\psi$ are satisfied. The system of conjunctions Ψ'_i, generated by the system-representation algorithm, is hence satisfied by this extended assignment, and therefore Ψ_{i+1} is satisfied as well by Proposition Proposition 3.7.

Let us now prove the soundness property. Assume that Ψ_{i+1} is γ-satisfiable by a feasible assignment $\mathcal{A}$. Then by Proposition Proposition 3.7, Ψ'_i is at least $\gamma - |\mathcal{F}|^{-1}$ satisfied by $\mathcal{A}$. Recall that the number of conjunctions in the set $\mathcal{E}_\psi$ is the same for every $\psi \in \Psi_i$. Hence for at least a $\gamma - 2|\mathcal{F}|^{-1/2}$ fraction of the sets $\mathcal{E}_\psi$, a $|\mathcal{F}|^{-1/2}$ fraction of the conjunctions are satisfied: Otherwise the fraction of satisfied conjunctions in Ψ_{i+1} would be less than $\gamma - 2|\mathcal{F}|^{-1/2} + (1 - \gamma + 2|\mathcal{F}|^{-1/2})|\mathcal{F}|^{-1/2} < \gamma - |\mathcal{F}|^{-1}$.

By the restriction property of $\mathbf{A}$, it follows that at least a $\gamma - 2|\mathcal{F}|^{-1/2}$ fraction of the equations in Ψ_i are satisfied by $\mathcal{A}$. The proof is thus completed, noting that the restriction of $\mathcal{A}$ to the variables of Ψ_i is feasible. ■

5.3. The Representation-Procedures. We now state the properties of the representation-procedures that are utilized in reducing Ψ to Ψ_{sc}. Only the properties that are needed for the reduction are discussed – the actual representation-procedures and the proofs of their stated properties appear later.

Product-check. The product-check algorithm is actually not a representation-procedure. Its properties are stated here since it is applied to Ψ_0 to produce Ψ_1. The generated system Ψ_1 is a restricted equation-system with just one domain, which is an active domain. Another important property of Ψ_1 is that its equations are *condensed*, as defined below. This property is necessary since the arithmetization representation-procedure is applied to Ψ_1 to obtain Ψ_2, and it can only be applied to systems with condensed equations. A more detailed explanation of the use of condensed equations appears in Subsection Section 5.7, which describes the arithmetization representation-procedure.

The principal cube of a domain. To define the principal cube we assume that for each non-negative number $s < |\mathcal{F}|$ an arbitrary subset $\mathcal{H}_s$ of $\mathcal{F}$ is fixed, of size $s + 1$. The principal cube of a domain F is defined to be the subset $\left(\mathcal{H}_{s(F)}\right)^{d(F)} \subseteq \mathcal{F}^{d(F)}$.

Definition 5.5 (condensed equations). *Let ψ be an equation or a conjunction with an active domain F, in a restricted equation-system. ψ is said to be condensed if all of its active variables are associated with points in the principal cube of F. A restricted equation-system where all of its equations are condensed is called condensed.*

Lemma 5.6 (product-check). *Let Ψ be a system of n quadratic equations over $\mathcal{F}$, where there are at most n variables in each equation. There exists a gap-preserving polynomial time algorithm that given such a system, constructs a restricted equation-system Ψ_* that has the following properties:*

- $\mathrm{D}_{\mathbf{c}}(\Psi_*)$ *is bounded by a constant.*
- Ψ_* *has exactly one domain F which is the active domain of all of its equations. The parameters of F, that also determine the active degree and dimension parameters of Ψ_*, are $r(F) = |\mathcal{F}|^{1/4}$, $s(F) = |\mathcal{F}|^{1/8}$, and $d(F) = \Theta(\log^{1-\beta} n)$.*
- *The equations in Ψ_* are all condensed.*

Arithmetization. The arithmetization representation-procedure uses a technique from Babai *et al.* (1991) to generate systems with a reduced active-dependency parameter. Given a condensed equation ψ, it produces a representation $\mathcal{E}_\psi$ where the number of active variables in each conjunction is a function of the degree and dimension parameters of $\mathrm{Dom}_\star(\psi)$. If these parameters are small enough, then the active-dependency is decreased. Note that the degree and dimension parameters themselves are not decreased, hence an iterative application of the arithmetization representation-procedure would not further reduce the dependency.

Lemma 5.7 (arithmetization). *Let Ψ be a restricted equation-system satisfying $s_\star(\Psi)d_\star(\Psi)+r_\star(\Psi) < |\mathcal{F}|^{1/2}$ and $s_\star(\Psi) > 2$, and where all the equations are condensed.*

There exists a $[2d_\star(\Psi)s_\star(\Psi)\,,\, d_\star(\Psi)+1]$-representation-procedure called **arithmetization**, *applicable to the equations of such systems, that generates conjunctions with at most $2d_\star(\Psi)\cdot s_\star(\Psi)$ active variables.*

Curve-extension. When applied to equations with a small active-dependency parameter, the curve-extension representation-procedure generates domains with small degree and dimension parameters. The active dependency is not reduced (in fact it increases somewhat), but then the system-representation algorithm is applied to the resulting system using the arithmetization representation-procedure, and the decrease in the degree and dimension parameters is utilized to reduce the active dependency as well (note that the arithmetization representation procedure doesn't care at all about the active dependency of the equations it is applied to). By applying the system-representation algorithm using the curve-extension and the arithmetization representation-procedures alternately, the reduction gradually reduces the active-dependency, the active degree and the dimension parameters of the intermediate systems.

Lemma 5.8 (curve-extension). *Let Ψ be a restricted equation-system such that $s_\star(\Psi)\mathrm{D}_\star(\Psi)$ and $r_\star(\Psi)\mathrm{D}_\star(\Psi)$ are smaller than $|\mathcal{F}|^{1/2}$. There exists an $[s,d]$-representation-procedure called* **curve-extension**, *applicable to the equations of such systems, for*

$$d \doteq \min\left\{d_\star(\Psi), \log_2\left(s_\star(\Psi)\cdot \mathrm{D}_\star(\Psi)\right)\right\}$$

$$\text{and} \quad s \doteq d\cdot \max\left\{\left(s_\star(\Psi)\cdot \mathrm{D}_\star(\Psi)\right)^{\frac{1}{d_\star(\Psi)}}, 2\right\}$$

that generates only condensed conjunctions.

Linearization. Applying the system-representation algorithm using the linearization representation-procedure obtains a system with constant active-dependency, as desired. However it is applicable in polynomial time only to systems where the active degree and dimension parameters are very small (the running time of a representation-procedure is polynomial in the size of the newly generated domains, which may become very large in the case of linearization). Hence the reduction generates a sequence of intermediate equation-systems where the active parameters are gradually reduced, until they finally become suitable for the linearization representation-procedure to be applied.

Lemma 5.9 (linearization). *Let Ψ be a system such that $r_\star(\Psi)\mathrm{D}_\star(\Psi)$ and $s_\star(\Psi)\mathrm{D}_\star(\Psi)$ are smaller than $(|\mathcal{F}|^{1/2})/2$.*

There exists a $[1, s_\star(\Psi)\mathrm{D}_\star(\Psi)]$-representation-procedure called **linearization** *and is applicable to such systems, that generates conjunctions with at most 4 active variables.*

Note that as mentioned above, for the linearization representation-procedure to be applicable within the reduction, $s_\star(\Psi)\mathrm{D}_\star(\Psi)$ should be in fact considerably smaller than the above bound of $|\mathcal{F}|^{1/2}$.

5.4. The Reduction Algorithm of Ψ Into Ψ_{sc}. We now state the reduction algorithm that transforms Ψ into Ψ_{sc}, as claimed by Lemma Lemma 3.1. This algorithm is mostly a concatenation of the algorithms that were discussed above. Starting with $\Psi = \Psi_0$, the reduction algorithm applies the product-check algorithm to obtain Ψ_1, and from there it continues to use the system-representation algorithm, applying it a constant ($O(\frac{1}{1-\beta})$) number of times with different representation-procedures. This yields a sequence of equation-systems $\Psi_2, \ldots, \Psi_l$. Ψ_{sc} is then obtained from Ψ_l by a simple transformation.

We next give the sequence of transformations and representation-procedures used to obtain Ψ_l from Ψ_0, and then describe how Ψ_{sc} is obtained from Ψ_l. In Subsection Section 5.5 it is shown that this reduction takes polynomial time in n, and that the generated system Ψ_{sc} has the desired properties. Subsection Section 5.5 also shows that although each representation-procedure is applicable only to systems with certain parameters, the reduction algorithm does use them correctly.

The sequence of systems. First, the reduction applies the product-check algorithm to Ψ_0 and obtains Ψ_1. The next $\left(\frac{2\beta}{1-\beta}+4\right)$ systems[§], $\Psi_2, \ldots, \Psi_{\frac{2\beta}{1-\beta}+5}$ are generated, by ap-

[§]For simplicity of exposition, we assume here that $\beta/(1-\beta)$ is an integer.

plying the system-representation algorithm with the arithmetization and the curve-extension representation-procedures alternately (the arithmetization is used first). The system-representation algorithm is then applied once more to $\Psi_{\frac{2\beta}{1-\beta}+5}$ using the arithmetization representation-procedure, and then finally it is applied once again using the linearization representation-procedure. The outcome is Ψ_l, where $l\doteq\frac{2\beta}{1-\beta}+7$. Apart from a simple transformation that is described shortly below, Ψ_l is the outcome of the reduction.

Properties of Ψ_l. Before we describe how Ψ_l is transformed into Ψ_{sc}, let us overview its main properties.

Constant dependency. Ψ_l has the desired dependency parameter, namely a constant. Since it is generated using the linearization representation-procedure, it follows from Lemma Lemma 5.9 that its active-dependency parameter is constant. As for the core-dependency, Ψ_1 is generated using the product-check algorithm and therefore by Lemma Lemma 5.6 its core-dependency is constant. Since the other systems in the sequence $\Psi_2, \ldots, \Psi_l$, are generated using the system-representation algorithm, the core-dependency increases only by a constant throughout the sequence (recall that the sequence is of constant length).

Completeness, and soundness. Since each of the intermediate transformations that were applied so far are gap-preserving, it follows immediately that the transformation from $\Psi = \Psi_0$ into Ψ_l is gap preserving as well. Hence Ψ_l has the following properties:

- Completeness: If Ψ can be completely satisfied by a good assignment then so can Ψ_l.
- Weakened Soundness: If Ψ is no more than $\frac{2}{|\mathcal{F}|}$-satisfiable then Ψ_l cannot be more than $O(|\mathcal{F}|^{-1/2})$-satisfied by a feasible assignment.

From Ψ_l to Ψ_{sc}. Ψ_l fails to comply with two requirements of Lemma Lemma 3.1: The parameters of its domains are not all the same, and it has only a weakened soundness property, which is less than what is required in Lemma Lemma 3.1. The reduction hence transforms Ψ_l into Ψ_{sc} in two steps. First it resets the degree and dimension parameters of its domains without changing any of the other properties, and then it applies a simple technique to amplify the soundness property.

Parameter uniformization. First note that the upper-degree parameter is the same for all the domains of Ψ_l since Ψ_1 has only one domain, and the representation-procedures generate domains with the same upper-degree as the active domain of the equation to which they are applied. Denote this upper-degree by $r(\Psi_{sc})$, and fix $s(\Psi_{sc})$ to be the maximum over all lower-degrees of domains in Ψ_l, and $d(\Psi_{sc})$ to be the maximum over all the dimension parameters. As shown in Subsection Section 5.5, $s(\Psi_{sc})$ is smaller than $r(\Psi_{sc})$.

The reduction replaces each domain F of Ψ_l by a new domain F' with $r(F') = r(\Psi_{sc})$, $s(F') = s(\Psi_{sc})$ and $d(F') = d(\Psi_{sc})$. Each variable $F[x]$ which appears in an equation of Ψ_l is then replaced by the variable $F'[x']$, where x' is obtained from x by padding it with the appropriate number of zeros (in case the dimension parameter of F' is larger than that of F).

Note that the completeness and weakened soundness properties of Ψ_l are not affected by the uniformization step. Resetting the lower-degree parameter maintains the completeness property since the lower-degree parameters may only be increased, and it has no effect on the soundness. The dimension enlargement also preserves the completeness property, as an LDF that was assigned to a domain before the change of dimension extends naturally to the larger domain maintaining the same degree, and thus a satisfying assignment can be translated through the uniformization step. Similarly, a feasible assignment to a domain with an enlarged dimension translates to a feasible assignment to the original domain by restriction, thus preserving the values of the variables appearing in the equations, and therefore the weakened soundness property is also maintained.

Soundness amplification. To amplify the soundness of Ψ_l the reduction first generates all conjunctions of three (not necessarily distinct) equations from Ψ_l. It then replaces each such conjunction with the set of all linear-combinations over its equations. The set of equations of Ψ_{sc} is thus

$$\{ \sum_{i=1}^{3} \lambda_i \psi_i \; : \quad \forall\, i \quad \lambda_i \in \mathcal{F}\,,\ \psi_i \in \Psi_l \}$$

Completeness and soundness for Ψ_{sc}. Since it is simple to observe that the completeness property is maintained by the soundness amplification step, let us verify that Ψ_{sc} has the soundness property. Assume then that Ψ is no more than $2/|\mathcal{F}|$-satisfiable. As mentioned above, a feasible assignment for Ψ_l cannot satisfy more than an $O(|\mathcal{F}|^{-1/2}) < |\mathcal{F}|^{-1/3}$ fraction of its equations, and this remains true when the domain-parameters of Ψ_l are reset. The fraction of conjunctions of three equations that can be satisfied by a feasible assignment is hence less than $(|\mathcal{F}|^{-1/3})^3 = 1/|\mathcal{F}|$. It then follows from Proposition Proposition 3.7 that Ψ_{sc} cannot be more than $2/|\mathcal{F}|$-satisfiable by a feasible assignment (Proposition Proposition 3.7 discusses general assignments but it is easily extendible to feasible assignments).

5.5. The Reduction Works. Based on the stated properties of the representation-procedures, we now verify that the reduction algorithm described above is applicable, and that the generated system Ψ_{sc} has the required parameters. The completeness and soundness properties of Ψ_{sc} have already been verified. From the properties of Ψ_l and the construction of Ψ_{sc} it is obvious that the number of variables in the equations of Ψ_{sc} is bounded by a constant and that the parameters of its domains are all the same.

We now compute the active parameters of all the intermediate systems $\Psi_1, \ldots, \Psi_l$, and at the same time verify that all representation-procedures are correctly used by the reduction. The computation will also imply that the parameters of the domains of Ψ_{sc} are as required by Lemma Lemma 3.1, and that the reduction takes polynomial time. For simplicity, we use O and Θ notations in the computation, where any function that depends solely on β is regarded as constant.

The active parameters of the intermediate systems. As mentioned above the domains of Ψ_{sc}, as well as the domains in all the intermediate systems, all have the same upper-degree parameter, namely $r(\Psi_{sc})$. It also equals the active upper-degree of Ψ_1, hence $r(\Psi_{sc}) = |\mathcal{F}|^{1/4}$. Let us consider the other parameters of the intermediate systems.

The parameters of Ψ_1. Ψ_1 is generated from Ψ_0 using the product-check algorithm (see Lemma Lemma 5.6), hence it has the parameters

- $s_\star(\Psi_1) = |\mathcal{F}|^{1/8}$
- $d_\star(\Psi_1) = \Theta(\log^{1-\beta} n)$

The parameters of Ψ_2. Ψ_2 is obtained from Ψ_1 using the arithmetization representation-procedure. Note that the parameters of Ψ_1 are such that the arithmetization representation-procedure is applicable. The parameters of Ψ_2, as follows from the arithmetization lemma, are

- $s_\star(\Psi_2) = 2d_\star(\Psi_1)s_\star(\Psi_1) = \Theta(|\mathcal{F}|^{1/8}\log^{1-\beta} n) = 2^{\Theta(\log^\beta n)}$
- $\mathrm{D}_\star(\Psi_2) = 2d_\star(\Psi_1)s_\star(\Psi_1) = \Theta(|\mathcal{F}|^{1/8}\log^{1-\beta} n) = 2^{\Theta(\log^\beta n)}$
- $d_\star(\Psi_2) = d_\star(\Psi_1) + 1 = \Theta(\log^{1-\beta} n)$

The active parameters of $\Psi_3, \Psi_4, \ldots, \Psi_{l-7}$ (recall that $l - 7 = \frac{2\beta}{1-\beta}$) are given by the following proposition.

PROPOSITION 5.10. *For i such that $3 \leq 2i-1 \leq l-8$, the active parameters of Ψ_{2i-1} (generated using the curve-extension representation-procedure) are*

- $s_\star(\Psi_{2i-1}) = 2^{\Theta(\log^{\beta-(i-1)(1-\beta)} n)}$
- $d_\star(\Psi_{2i-1}) = \Theta(\log^{1-\beta} n)$

and for i such that $4 \leq 2i \leq l-7$, the parameters of Ψ_{2i} (that is generated using the arithmetization representation-procedure) are

- $s_\star(\Psi_{2i}) = 2^{\Theta(\log^{\beta-(i-1)(1-\beta)} n)}$
- $\mathrm{D}_\star(\Psi_{2i}) = 2^{\Theta(\log^{\beta-(i-1)(1-\beta)} n)}$
- $d_\star(\Psi_{2i}) = \Theta(\log^{1-\beta} n)$

PROOF. The proposition is obtained by induction over i, calculating the parameters of an equation-system according to the parameters of the previous system and the properties of the appropriate representation-procedure. We omit the calculation. ■

Note that the systems Ψ_{2i} have parameters such that the curve-extension representation-procedure is applicable, and that the arithmetization representation-procedure is applicable for the Ψ_{2i-1} systems, hence the sequence of transformation is valid up to and including Ψ_{l-7}. From the computations below it is also implied that the representation-procedures used for generating $\Psi_{l-6}, \ldots, \Psi_l$ are also applicable.

Parameters of Ψ_{l-6}. Setting $2i = l - 7 = 2(\frac{\beta}{1-\beta})$ in the above proposition we obtain that $s_\star(\Psi_{l-7}) = \mathrm{D}_\star(\Psi_{l-7}) = 2^{\Theta(\log^{1-\beta} n)}$, and that $d_\star = \Theta(\log^{1-\beta} n)$. Hence according to the Curve-Extension Lemma (Lemma Lemma 5.8),

- $s_\star(\Psi_{l-6}) = \Theta(\log^{1-\beta} n) \cdot \Theta(1) = \Theta(\log^{1-\beta} n)$
- $d_\star(\Psi_{l-6}) = \Theta(\log^{1-\beta} n)$

Parameters of Ψ_{l-5}. The active parameters of this system, that is obtained using the arithmetization representation-procedure, are

- $s_\star(\Psi_{l-5}) = \Theta(\log^{2(1-\beta)} n)$
- $\mathrm{D}_\star(\Psi_{l-5}) = \Theta(\log^{2(1-\beta)} n)$
- $d_\star(\Psi_{l-5}) = \Theta(\log^{1-\beta} n)$

Parameters of Ψ_{l-4}. The system Ψ_{l-4}, generated using the curve-extension representation-procedure, has parameters

- $s_\star(\Psi_{l-4}) = \Theta(\log^{1-\beta} n)$
- $d_\star(\Psi_{l-4}) = \Theta(\log \log n)$

Parameters of Ψ_{l-3}. This system, obtained via the arithmetization representation-procedure, is the last before the linearization representation-procedure is applied. Its parameters are

- $s_\star(\Psi_{l-3}) = \Theta(\log \log n \cdot \log^{1-\beta} n)$
- $\mathrm{D}_\star(\Psi_{l-3}) = \Theta(\log \log^2 \cdot \log^{1-\beta} n)$
- $d_\star(\Psi_{l-3}) = \Theta(\log \log n)$

Parameters of Ψ_{l-2}. The system Ψ_{l-2}, generated using the curve-extension representation-procedure, has parameters

- $s_\star(\Psi_{l-2}) = \Theta(\log \log n)$
- $d_\star(\Psi_{l-2}) = \Theta(\log \log n)$

Parameters of Ψ_{l-1}. This system, obtained via the arithmetization representation-procedure, is the last before the linearization representation-procedure is applied. Its parameters are

- $s_\star(\Psi_{l-1}) = \Theta(\log\log^2 n)$
- $\mathrm{D}_\star(\Psi_{l-1}) = \Theta(\log\log^2 n)$
- $d_\star(\Psi_{l-1}) = \Theta(\log\log n)$

Parameters of Ψ_l. Ψ_{l-1} obviously satisfies the conditions of the Linearization Lemma (Lemma Lemma 5.9). According to the lemma, the parameters of Ψ_l are

- $s_\star(\Psi_l) = 1$
- $\mathrm{D}_\star(\Psi_l) \leq 4$
- $d_\star(\Psi_l) = \Theta(\log\log^4 n)$

The parameters of Ψ_{sc}. By the above computations it is possible to deduce the parameters of the domains of Ψ_{sc}. Noting that $s_\star(\Psi_2)$ is the highest active low-degree parameter of all intermediate systems it follows that $s(\Psi_{sc}) = s_\star(\Psi_2) = \Theta(|\mathcal{F}|^{1/8} \log^{1-\beta} n)$. Since $r(\Psi_{sc}) = |\mathcal{F}|^{1/4}$, it follows that the requirements over s and r in Lemma Lemma 3.1 hold. The above computations also imply that the active dimension of all intermediate systems is bounded by $O(\log^{1-\beta} n)$, and hence $d(\Psi_{sc}) = \Theta(\log^{1-\beta} n)$ as required.

Polynomial time. Since Ψ_{sc} was shown to satisfy all the requirements of Lemma Lemma 3.1, it is only left to verify that it is obtained from Ψ_0 in polynomial time. Ψ_1 is obtained in polynomial time, as stated in lemma Lemma 5.6. The other intermediate systems $\Psi_2, \ldots, \Psi_l$, are obtained by applying the system-representation algorithm. As stated in Subsection Section 5.2, an application of the system-representation algorithm to a system Ψ_{i-1} takes polynomial time in the size of Ψ_{i-1} and in $|\mathcal{F}|^{d_\star(\Psi_i)}$.

According to the computations above $d_\star(\Psi_i) = O(\log^{1-\beta} n)$ for all i, so $|\mathcal{F}|^{d_\star(\Psi_i)}$ is polynomial in n. By induction it is therefore easy to verify that all intermediate systems are produced in polynomial time in n. The transformation of Ψ_l into Ψ_{sc} obviously takes polynomial time in the size of Ψ_l, so the entire reduction takes polynomial time in n.

■

5.6. The Product-Check Lemma. In this subsection we prove the product-check lemma. We show an algorithm that transforms a given quadratic-equation system into a restricted equation-system with one domain, which has a relatively small (with respect to the size of the field) dimension parameter.

The product-check algorithm actually disposes of all the variables of Ψ, substituting them by the variables of the new domain F. Each variable of Ψ and each product of two such variables is replaced by a variable of the form $F[x]$ that represent it. This is done so that for every assignment of Ψ there is a *good* assignment to F, where the value of each variable $F[x]$ is equal to the value of the corresponding term in Ψ.

However, not every feasible assignment to F indeed represents an assignment of Ψ. Consider two variables of Ψ that are represented by $F[x_1]$ and $F[x_2]$ in F. There is no guarantee that the value of the variable $F[x]$ that represents their product is indeed the product of the values of $F[x_1]$ and $F[x_2]$. Each equation of Ψ is hence replicated several times in Ψ_*, where a "product-test" is added in conjunction to each copy to verify the correctness of the assignment.

The product-check algorithm.

Setting parameters and generating F. Let $h \doteq |\mathcal{F}|^{1/9}$, and let $d \doteq \lceil \log_h(n+1) \rceil$ (note that $d = O(\log^{1-\beta} n)$). The procedure constructs a new domain F with lower-degree parameter $s(F) = |\mathcal{F}|^{1/8}$, upper-degree $r(F) = |\mathcal{F}|^{1/4}$, and dimension $d(F) = 2d$.

Representing terms. The procedure chooses $\mathcal{H} \subseteq \mathcal{H}_{s(F)} \subseteq \mathcal{F}$ to be an arbitrary set of size h. It then selects an arbitrary *injection* $v \to x_v$, associating every variable of Ψ with a point in $\mathcal{H}^d \subseteq \mathcal{F}^d$ (such an injection exists). The procedure chooses another distinct point $x_I \in \mathcal{H}^d$ to represent the value 1. Writing points in $\mathcal{F}^{2d}$ as pairs (x_1, x_2) of points in $\mathcal{F}^d$, each variable v of Ψ is represented in Ψ_* by $F[(x_v, x_I)]$, and the product of two variables u, v is represented by $F[(x_u, x_v)]$.

Generating conjunctions. The procedure replaces each equation ψ of Ψ by a set $\mathcal{E}_\psi$ of conjunctions as follows. Given ψ, it produces one conjunction in $\mathcal{E}_\psi$ for every point $(x_1, x_2) \in \mathcal{F}^{2d}$, consisting of the following equations:

1. ψ itself, where every product $u \cdot v$ is replaced by $F[(x_u, x_v)]$ and every variable v in a linear term is replaced by $F[(x_v, x_I)]$.
2. The product-test equation $F[(x_1, x_I)] \cdot F[(x_2, x_I)] = F[(x_1, x_2)]$.

From conjunctions to equations. Let Ψ' denote the system of conjunctions, containing the union of all the sets $\mathcal{E}_\psi$ where $\psi \in \Psi$. The system Ψ_* is generated from Ψ' by replacing each conjunction with all linear combinations of its equations, as described in Proposition Proposition 3.7. For every $\chi \in \Psi_*$ we set $\mathrm{Dom}_\star(\chi)$ to be F.

Observing the construction of the conjunctions and of Ψ_*, one notes that there is at most one quadratic term in each equation $\chi \in \Psi_*$, and at most one more variable in each equation that is associated with a point outside $\mathcal{H}^{2d}$. These terms (the quadratic term and the additional variable), and also the constant term of each equation χ are moved, if they exist, to the right-hand side of ψ and are set to be the core of ψ. The other terms are moved to the left-hand side, which is set to be the active part of ψ. The active variables of ψ are therefore all associated with points in $\mathcal{H}^{2d}$.

Proof of correctness. It is easy to observe that the product-check algorithm indeed takes polynomial time. The generated system Ψ_* has one domain F, with parameters as stated by Lemma Lemma 5.6, and its core dependency is bounded by the constant 3 as required. Since the active variables of equations in Ψ_* are all associated with points in $\mathcal{H}^{2d} \subseteq \left(\mathcal{H}_{s(F)}\right)^{d(F)}$, namely with points in the principal cube of F, we have that they are all condensed. It is left to show that the product-check algorithm is gap-preserving.

Completeness. Suppose Ψ is satisfiable by a good assignment $\mathcal{A}$. We show a good assignment $\mathcal{A}'$ for F which represents it, namely that

- For every variable v of Ψ, $\mathcal{A}'(F[(x_v, x_I)]) = \mathcal{A}(v)$.
- $F[(x_1, x_I)] \cdot F[(x_2, x_I)] = F[(x_1, x_2)]$ for every $x_1, x_2 \in \mathcal{F}^d$.

It is easy to observe that an assignment $\mathcal{A}'$ with the above properties will satisfy Ψ_*.

We define an LDF $f : \mathcal{F}^d \to \mathcal{F}$ and then use it to define $\mathcal{A}'$. For points $x_v \in \mathcal{H}^d$ associated with a variable v of Ψ we set $f(x_v) \doteq \mathcal{A}(v)$, and we also set $f(x_I) \doteq 1$. For points $x \in \mathcal{H}^d$ not associated with variables, we arbitrarily set $f(x) \doteq 0$. We extend f over $\mathcal{F}^d$ by the unique extension to an LDF of degree $h-1$ in each variable. The total degree of f is therefore $(h-1)d = O(|\mathcal{F}|^{1/9} \log n)$. $\mathcal{A}'$ will assign to F the LDF g, defined by $g(x_1, x_2) \doteq f(x_1) f(x_2)$. This is a good assignment since g is of total-degree $O(|\mathcal{F}|^{1/9} \log n) < |\mathcal{F}|^{1/8}$ (the inequality is true for large-enough n). The other stated properties of $\mathcal{A}'$ are easy to verify.

Soundness. The next proposition is the first step in proving the soundness property. It shows that in order for Ψ_* to be $|\mathcal{F}|^{-5/8}$-satisfiable by a feasible assignment $\mathcal{A}'$, $\mathcal{A}'$ must be consistent with an assignment $\mathcal{A}$ for Ψ. After proving the proposition we show that in that case $\mathcal{A}$ must satisfy almost the same (up to $\mathcal{F}^{-1}$) fraction of the equations in Ψ as $\mathcal{A}'$ does for Ψ_*.

PROPOSITION 5.11. *Let $\mathcal{A}'$ be an assignment of an $r(F)$-degree LDF g to F. If it satisfies at least an $|\mathcal{F}|^{-5/8}$ fraction of the equations in Ψ_*, then there is an assignment $\mathcal{A}$ for Ψ such that for every variable v of Ψ, $\mathcal{A}(v) = g(x_v, x_I)$, and for every two variables u, v of Ψ $\mathcal{A}(u)\mathcal{A}(v) = g(x_u, x_v)$.*

PROOF. Consider an assignment $\mathcal{A}'$ as above, that assigns an LDF g to F and satisfies at least a $|\mathcal{F}|^{-5/8}$ fraction of the equations of Ψ_*. We define an $[r(F), d]$-LDF f by $f(x) \doteq g(x, x_I)$, and set an assignment $\mathcal{A}$ for every variable v of Ψ by $\mathcal{A}(v) \doteq f(x_v)$ (hence the first stated property of $\mathcal{A}$ holds).

By Proposition Proposition 3.7, if $\mathcal{A}'$ satisfies more than an $|\mathcal{F}|^{-5/8}$ fraction of the equations of Ψ_*, then it satisfies an $\Omega(|\mathcal{F}|^{-5/8})$ fraction of the conjunctions in Ψ'. Then, for at least one of the equations $\psi \in \Psi$, the fraction of satisfied conjunctions in $\mathcal{E}_\psi$ is at least $\Omega(|\mathcal{F}|^{-5/8})$. By observing the product-test in each conjunction of $\mathcal{E}_\psi$, we obtain that for an $\Omega(|\mathcal{F}|^{-5/8})$ fraction of the points $(x_1, x_2) \in \mathcal{F}^{2d}$,

$$f(x_1)f(x_2) = \mathcal{A}'(F[(x_1, x_I)])\mathcal{A}'(F[(x_2, x_I)]) = \mathcal{A}'(F[(x_1, x_2)]) = g(x_1, x_2) \tag{5.12}$$

In both sides of the equation we have LDFs of degree at most $2r(F) = O(|\mathcal{F}|^{1/4})$. Different LDFs of such parameters may only agree on an $O(|\mathcal{F}|^{1/4}/|\mathcal{F}|) = O(|\mathcal{F}|^{-3/4})$ fraction of the points, however the LDFs in Equation (5.12) agree on an $\Omega(|\mathcal{F}|^{-5/8})$ fraction and are hence equal. We therefore have

$$\forall\, (x_1, x_2) \in \mathcal{F}^{2d} \quad g(x_1, x_2) = f(x_1)f(x_2)$$

and specifically

$$\forall\, v, u \quad \mathcal{A}(u)\mathcal{A}(v) = f(x_u)f(x_v) = g(x_u, x_v)$$

as required. ■

We now return to the soundness proof of the product-check procedure. Assume that Ψ_* is γ-satisfiable by a feasible assignment $\mathcal{A}'$, and let us show an assignment $\mathcal{A}$ satisfying a $\gamma - O(|\mathcal{F}|^{-1/2})$ fraction of the equations in Ψ. We may assume that $\gamma > |\mathcal{F}|^{-1/2}$ (otherwise there is nothing to show), and hence there exists an assignment $\mathcal{A}$ for Ψ that corresponds to $\mathcal{A}'$ as in Proposition Proposition 5.11.

The fraction of conjunctions in Ψ' that are satisfied by $\mathcal{A}'$ is, by Proposition Proposition 3.7, at least $\gamma - |\mathcal{F}|^{-1}$. Hence for the same fraction of equations ψ of Ψ, there is at least one conjunction $\chi \in \mathcal{E}_\psi$ which is satisfied by $\mathcal{A}'$. One of the equations in such a conjunction χ is a copy of ψ where certain terms are replaced. According to Proposition Proposition 5.11 the replaced terms have the same value as the replacing terms, and therefore ψ is satisfied by $\mathcal{A}$. This implies that at least a $\gamma - |\mathcal{F}|^{-1} > \gamma - |\mathcal{F}|^{-1/2}$ fraction of the equations of Ψ' are satisfied by $\mathcal{A}$.

5.7. The Arithmetization Representation-Procedure.

In this subsection we show how to reduce the active dependency parameter by using a sum-check technique in the spirit of Babai *et al.* (1991). When applied to an equation ψ whose active LDF active domain? has small degree and dimension parameters, the arithmetization procedure produces a representation $\mathcal{E}_\psi$ with small active-dependency.

Basic idea. Let ψ be a condensed equation of the form $\psi_\star = \psi_{\mathbf{c}}$ (where $\psi_\star$ is the active part of ψ). Then $\psi_\star$ can be written as a sum

$$\psi_\star \;: \qquad \sum_{y \in {\mathcal{H}_s}^d} \kappa(y)E[y],$$

where E is the active domain of ψ, and the elements $\kappa(y)$ are coefficients . Our goal is to reduce the active dependency of ψ. For this purpose, the arithmetization procedure generates a domain F which corresponds to a *sum-check LDF*. The sum-check LDF encodes a sequence of partial-sum polynomials, similar to the one used in Babai *et al.* (1991), for the evaluation of $\psi_\star$. The conjunctions in $\mathcal{E}_\psi$ will ensure the consistency of these partial-sum polynomials thereby verifying the original equation.

Notation. We describe the running of the arithmetization representation-procedure over a given *condensed* equation ψ in a restricted equation-system Ψ. For shortness we denote $E\doteq\mathrm{Dom}_\star(\psi)$, $r\doteq r(E)$, $s\doteq s(E)$, and $d\doteq d(E)$.

The sum-check LDF. We define the sum-check LDF of ψ with respect to a given good assignment $\mathcal{A}$ for E. First, we extend κ to an LDF of degree ds over $\mathcal{F}^d$ – such an extension exists and is computable in polynomial time in $|\mathcal{F}|^d < |F|$, and hence it is possible to compute the extension within the representation-procedure. We now define d LDFs that encode different partial sums of $\psi_\star$. The sum-check LDF is constructed from these LDFs below.

Definition 5.13 (the sum-check tree). *For $k = 1, 2, \ldots, d$, we define a function $g_k : \mathcal{F}^k \to \mathcal{F}$ by*

$$\forall x \in \mathcal{F}^k \quad g_k(x)\doteq \sum_{y\in{\mathcal{H}_s}^{(d-k)}} \kappa(x,y)\mathcal{A}(E[x,y])$$

where "x, y" means the concatenation of the vector x and the vector y. The sequence $g_1, \ldots, g_d$ is called the sum-check tree with respect to $\mathcal{A}(E)$.

For an $x \in {\mathcal{H}_s}^k$ the value of $g_k(x)$ is a partial sum of $\psi_\star$. The value of g_d at a point $x \in \mathcal{F}^d$ is just $\kappa(x)\mathcal{A}(E[x])$, and hence g_d is an LDF of degree at most $ds + s = (d + 1)s$. It follows from the above definition that the other g_k's have degree at most $(d + 1)s$ as well.

The LDFs $g_1, \ldots, g_d$ form a tree of partial sums in the following sense. Consider a tree of depth d, where every non-leaf node has $|\mathcal{F}|$ offsprings, and every node of depth $k > 0$ is labeled by a point evaluation of g_k. We label the root by $\sum_{y\in{\mathcal{H}_s}^d} \kappa(y)\mathcal{A}(E[y])$, which is the evaluation of $\psi_\star$. The root has an offspring labeled by $g_1(z)$, for each $z \in \mathcal{F}$. Note that for $z \in \mathcal{H}_s$, $g_1(z)$ is a partial sum of $\psi_\star$, and in fact the root-label is the sum of labels of its offsprings that are assigned $g_1(z)$ for $z \in \mathcal{H}_s$.

For a non-leaf node that has been labeled $g_k(x)$, we label one of its offsprings by $g_{k+1}(x, z)$ for every $z \in \mathcal{F}$. From the definition of the g_k's it follows that for every $k < d$ and $x \in \mathcal{F}^k$,

$$g_k(x) = \sum_{z\in\mathcal{H}_s} g_{k+1}(x, z) \tag{5.14}$$

Hence the label of each node labeled $g_k(x)$ in the tree is the sum of labels of its $s+1$ offsprings that are assigned $g_{k+1}(x, z)$ for $z \in \mathcal{H}_s$.

The sum-check LDF. We now incorporate all the LDFs $g_1, \ldots, g_k$ into a single LDF of degree at most $(d + 1)s + d \le 2ds$, called the sum-check LDF. For this purpose, let $\mathcal{H}_{d-1} = \{a_1, \ldots, a_d\}$ be an arbitrary subset of size d in $\mathcal{F}$. The sum-check LDF, denoted by f, will satisfy

$$f(a_k, x_1, \ldots, x_k, 0, \ldots, 0) = g_k(x_1, \ldots, x_k) \tag{5.15}$$

for every $1 \leq k \leq d$, and every $x = (x_1, \dots, x_k) \in \mathcal{F}^k$. There exists such an f – for example it can be defined by

$$f(x_0, x_1, \dots, x_d) \doteq \sum_{k=1}^{d} \left(\prod_{i \neq k} \frac{x_0 - a_i}{a_k - a_i} \right) \cdot g_k(x_1, .., x_k)$$

Properties of the sum-check LDF. From Equation (5.15) and the discussion above it follows that the sum-check LDF has the following properties:

- $\sum_{z \in \mathcal{H}_s} f[(a_1, z, 0, \dots, 0)]$ is the evaluation of $\psi_\star$, as follows from the explanation after Definition Definition 5.13.
- For $k = 1, 2, \dots, (d-1)$ and every $(x_1, \dots, x_k) \in \mathcal{F}^k$
$$f[(a_k, x_1, \dots, x_k, 0, \dots, 0)] = \sum_{z \in \mathcal{H}_s} f[(a_{k+1}, x_1, \dots, x_k, z, 0, \dots, 0)]$$
as follows from Equation (5.14).
- For every $(x_1, \dots, x_d) \in \mathcal{F}^d$,
$$f[(a_d, x_1, \dots, x_d)] = \kappa(x_1, \dots, x_d) \mathcal{A}(E[x_1, \dots, x_d])$$
as follows from the explanation after Definition Definition 5.13.

The arithmetization representation-procedure. We now give the details of the arithmetization representation procedure. At first the representation-procedure produces a new domain $F = \text{Dom}_\star(\mathcal{E}_\psi)$ with parameters as stated in Lemma Lemma 5.7, namely $r(F) = r$, $s(F) = 2ds$ and $d(F) = d+1$. The procedure generates conjunctions that can only be satisfied if F is assigned the sum-check f. For each $x = (x_1, \dots, x_d) \in \mathcal{F}^d$ the procedure generates one conjunction, denoted by $\chi[x]$, consisting of the following $d + 1$ equations:

- The *root equation*:
$$\sum_{z \in \mathcal{H}_s} F[a_1, z, 0, \dots, 0] = \psi_{\mathbf{c}}$$
- The $d - 1$ *path equations* for $k = 1, 2, \dots, (d-1)$:
$$F[a_k, x_1, \dots, x_k, 0, \dots, 0] = \sum_{z \in \mathcal{H}_z} F[a_{k+1}, x_1, \dots, x_k, z, 0, \dots, 0]$$
- The *leaf* equation:
$$F[a_d, x_1, \dots, x_d] = \kappa(x_1, \dots, x_d) E[x_1, \dots, x_d]$$

Proof of correctness. Let us show that the arithmetization representation-procedure has the required properties. It is easy to verify that it runs in polynomial time, and that it generates a domain $F = \mathrm{Dom}_\star(\psi)$ with parameters as required. As for the number of active variables in each conjunction, there are $s+1$ variables from F in the root equation, $s+2$ variables in each of the $d-1$ path equations, and one variable in the leaf equation. The total number is therefore $s+1+(d-1)(s+2)+1 = ds+2d \leq 2ds$ as required. It is left only to verify the extension and restriction properties.

Extension. Let $\mathcal{A}$ be a good assignment for the variables of Ψ. Extend $\mathcal{A}$ to F by assigning the sum-check LDF f to it (f is of degree less than $s(F)$). From the properties of f stated above, it easily follows that if ψ is satisfied by $\mathcal{A}$ then all the conjunctions of $\mathcal{E}_\psi$ are also satisfied by the extension of $\mathcal{A}$.

Restriction. Let $\mathcal{A}$ be a feasible assignment for the variables of Ψ and for F, and assume that at least an $|\mathcal{F}|^{-1/2}$ fraction of the conjunctions in $\mathcal{E}_\psi$ are satisfied. We define the sum-check tree $g_1,\ldots,g_d$ and the sum-check LDF f with respect to the assignment of E, as in Definition Definition 5.13 and Equation (5.15) above. Since now the degree of the LDF assigned to E may be up to r, the degree of the g_k's can be up to $sd+r < |\mathcal{F}|^{1/2}$. We claim that F must be assigned f, at least at the points that matter, as stated in the following claim.

Claim 5.16 (sum-check). *Suppose that at least an $|\mathcal{F}|^{-1/2}$ fraction of the conjunctions in $\mathcal{E}_\psi$ are satisfied by a feasible assignment. Then for every k, $1 \leq k \leq d$, and every $x = (x_1,\ldots,x_k) \in \mathcal{F}^k$,*

$$\mathcal{A}(F[a_k, x_1, \ldots, x_k, 0, \ldots, 0]) = g_k(x_1, \ldots, x_k)$$

Before proving the claim we show how it implies the restriction property. Note that the root equation is common to all the conjunctions in $\mathcal{E}_\psi$, and hence it must be satisfied. So together with the claim we have that the evaluation of $\psi_{\mathbf{c}}$ equals $\sum_{z\in\mathcal{H}_s} f(a_1, z, 0, \ldots, 0)$, which by the properties of the sum-check LDF equals the evaluation of $\psi_\star$. Therefore ψ is satisfied, as required.

Proof of the sum-check claim. For every k, $1 \leq k \leq d$, we define an $[r,k]$-degree LDF g'_k by

$$g'_k(x_1,\ldots,x_k) \doteq \mathcal{A}(F[a_k, x_1, \ldots, x_k, 0, \ldots, 0])$$

For the sake of contradiction, assume that $g'_k \neq g_k$ for some k, and choose k to be the highest for which this inequality holds. We distinguish between two cases for k:

- $k = d$: At least an $|\mathcal{F}|^{-1/2}$ fraction of the conjunctions of $\mathcal{E}_\psi$ are satisfied, and therefore at least the same fraction of the leaf equations are satisfied. So for at least an $|\mathcal{F}|^{-1/2}$ fraction of the points $x \in \mathcal{F}^d$, $g'_d(x) = \mathcal{A}(F[a_d, x]) = \kappa(x)\mathcal{A}(E[x]) = g_d(x)$. But according to our assumption $g'_d \neq g_d$ and therefore their evaluations can not be equal on more than an $\frac{sd+r}{|\mathcal{F}|} < |\mathcal{F}|^{-1/2}$ fraction of the points, a contradiction.

- $1 \le k < d$: At least an $|\mathcal{F}|^{-1/2}$ fraction of the conjunctions of $\mathcal{E}_\psi$ are satisfied, and therefore in at least the same fraction of them the k'th path equation is satisfied. It follows that for at least an $|\mathcal{F}|^{-1/2}$ fraction of the points $x = (x_1, \ldots, x_k) \in \mathcal{F}^k$,

$$\begin{aligned} g'_k(x) &= \mathcal{A}(F[a_k, x_1, \ldots, x_k, 0, \ldots, 0]) = \\ &= \sum_{z \in \mathcal{H}_s} \mathcal{A}(F[a_{k+1}, x_1, \ldots, x_k, z, 0, \ldots, 0]) = \\ &= \sum_{z \in \mathcal{H}_s} g'_{k+1}(x_1, \ldots, x_k, z) \end{aligned}$$

By the maximality of k we have that $g'_{k+1} = g_{k+1}$, hence for at least an $|\mathcal{F}|^{-1/2}$ fraction of the points x,

$$g'_k(x) = \sum_{z \in \mathcal{H}_s} g_{k+1}(x_1, \ldots, x_k, z) = g_k(x) \quad \text{(by Equation (5.14))}$$

This is a contradiction to our assumption that $g'_k \ne g_k$, since they are both of degree at most $sd + r$ and therefore our assumption implies that they can be equal on at most an $\frac{sd+r}{|\mathcal{F}|} < |\mathcal{F}|^{-1/2}$ fraction of the points.

■

5.8. The Curve-Extension Representation-Procedure. In this subsection we show the curve-extension representation-procedure. If it is applied to an equation with a small enough active-dependency, then the new generated domain has a small active lower-degree parameter, and for equations with even smaller active-dependency the active dimension parameter becomes small as well. The conjunctions that are generated by the procedure are all condensed.

Let us describe the running of the curve-extension representation-procedure over a given equation ψ. For shortness we denote $E \doteq \mathrm{Dom}_\star(\psi)$, $r \doteq r(E)$, $s \doteq s(E)$, $d \doteq d(E)$, and $D \doteq \mathrm{D}_\star(\psi)$.

The principle of the algorithm. Denote the active variables of ψ by $E[x_1], \ldots, E[x_D]$. We define below a polynomial vector function of small degree $\Gamma : \mathcal{F} \to \mathcal{F}^d$, that goes through the points $x_1, \ldots, x_D$. The assignment of the domain $F \doteq \mathrm{Dom}_\star(\mathcal{E}_\psi)$, generated by the curve-extension representation-procedure, encodes the restriction of the assignment of E to the points of the curve Γ.

Variables in F associated with certain points in its principal cube have, in a correct encoding, the values of the assignment of E at certain points on Γ. The values at other points on Γ can be computed by interpolation over these variables of F, making use of the fact that Γ has a small degree, and hence restricting the assignment of E to its points yields an LDF of small degree as well. The conjunctions of $\mathcal{E}_\psi$ use the variables of F to evaluate $\psi_\star$ and verify that ψ is satisfied, and they also test whether F is indeed given a correct encoding.

The curve-extension algorithm. At first the representation-procedure produces a new domain $F = \mathrm{Dom}_\star(\mathcal{E}_\psi)$ with parameters as stated in Lemma Lemma 5.8, that is

$$r(F) = r, \quad d(F) = \min\{d, \log_2(s\mathrm{D})\}, \quad \text{and} \quad s(F) = d(F) \cdot \max\left\{\left(s \cdot \mathrm{D}\right)^{1/d}, 2\right\}$$

Each element of $\mathcal{E}_\psi$ will be a conjunction of two condensed equations. One is an equation ψ', derived from ψ by replacing each of its active variables with a variable of F that "encodes" it. The other equation is taken from a set of equations called a *curve-verifier*. These equations are not satisfied unless the assignment of F is a correct encoding. Before the construction of these equations, we define the curve Γ and describe how the assignment of F encodes the restriction of the assignment of E to the points of Γ.

Definition 5.17 (the curve Γ). *Let $\mathcal{H}_{sD-1}$ be an arbitrary subset of $\mathcal{F}$ of size sD, and denote its elements by $a_1, \ldots, a_{sD}$. $\Gamma : \mathcal{F} \to \mathcal{F}^d$ is defined to be the $(D-1)$-degree polynomial vector function satisfying*

$$\forall\, 1 \le i \le D \qquad \Gamma(a_i) = x_i$$

where $E[x_1], \ldots, E[x_D]$ are the active variables of ψ. Γ can clearly be computed in polynomial-time.

Associating points with $a_1, \ldots, a_{sD}$. Let $\left(\mathcal{H}_{s(F)}\right)^{d(F)}$ be the principal cube of F. The procedure chooses an arbitrary subset $\mathcal{H} \subseteq \mathcal{H}_{s(F)}$ of size $s(F)/d(F) = \max\left\{\left(s \cdot \mathrm{D}\right)^{1/d}, 2\right\}$, and associates to each point a_i in $\mathcal{H}_{sD-1}$ a distinct point y_i in $\mathcal{H}^{d(F)}$ (note that $\mathcal{H}^{d(F)}$ is a subset of the principal cube of F and that it contains at least sD points). Each of the variables $F[y_i]$ will encode the value of $E[\Gamma(a_i)]$. The active variables of the conjunctions in $\mathcal{E}_\psi$ will all be of the form $F[y_i]$, so the conjunctions of $\mathcal{E}_\psi$ are condensed. It is important to note that any assignment to the variables $F[y_i]$ can be extended by interpolation to a good assignment for F, as is shown below.

Generating the curve-verifier. Suppose E is assigned an LDF g. Then a correct encoding assigns to $F[y_i]$ the value of g at $\Gamma(a_i)$. Since Γ is of degree at most $D-1$, if g is of degree s then $g \circ \Gamma$ is of degree less than $sD-1$. The value of g at any point on the curve Γ can hence be evaluated by interpolation over its values at $\Gamma(a_1), \ldots, \Gamma(a_{sD})$ or, if F is assigned a correct encoding, by interpolation over the variables $F[y_1], \ldots, F[y_{sD}]$. This is stated precisely in the following claim.

Claim 5.18 (curve-interpolation). *Let s and D be such that $sD < |\mathcal{F}|$. Then there exists a polynomial (in $|\mathcal{F}|$) algorithm that receives as input a point $x \in \mathcal{F}$ and outputs a* coefficient function *$\kappa_x : \{a_1, \ldots, a_{sD}\} \to \mathcal{F}$ with the following property: Every function $f' : \{a_1, \ldots, a_{sD}\} \to \mathcal{F}$ has a unique extension to an $[sD-1, 1]$-LDF f over $\mathcal{F}$, and f satisfies*

$$\forall\, x \in \mathcal{F} \qquad f(x) = \sum_{i=1}^{sD} \kappa_x(a_i) f'(a_i)$$

The curve-verifier will have one equation $\chi[x]$ for each point $x \in \mathcal{F}$. $\chi[x]$ verifies that the interpolation over $F[y_1], \ldots, F[y_{sD}]$ using the κ_x from Claim Claim 5.18 yields the value of $E[\Gamma(x)]$, as it should if F is assigned the encoding of a good assignment to E:

$$\chi[x] : \qquad \sum_{i=1}^{sD} \kappa_x(a_i) F[y_i] = E[\Gamma(x)]$$

The next claim shows that the curve-verifier equations cannot be satisfied unless F is indeed assigned a correct encoding.

CLAIM 5.19. *Let $\mathcal{A}$ be a feasible assignment for E and F. Let f be the $[sD-1,1]$-LDF defined by $f(x) = \sum_{i=1}^{sD} \kappa_x(a_i)\mathcal{A}(F[y_i])$, as in Claim Claim 5.18. Then either $\mathcal{A}(E) \circ \Gamma = f$, in which case all of the curve-verifier equations are satisfied, or less than an $|\mathcal{F}|^{-1/2}$ fraction of the curve-verifier equations are satisfied.*

PROOF. Note that an equation $\chi[x]$ of the curve-verifier is satisfied if and only if $E[\Gamma(x)]$ is assigned $f(x)$. It is thus obvious that these equations will all be satisfied if $\mathcal{A}(E) \circ \Gamma = f$. If this is not the case, then $\mathcal{A}(E) \circ \Gamma$ and f are in particular two different $[rD, 1]$-LDFs. Since $rD < |\mathcal{F}|^{1/2}$ it follows that their evaluations differ on all but less than an $|\mathcal{F}|^{1/2}/|\mathcal{F}| \leq |\mathcal{F}|^{-1/2}$ fraction of the points. Hence if $\mathcal{A}(E) \circ \Gamma \neq f$, then less than an $|\mathcal{F}|^{-1/2}$ fraction of the curve-verifier equations can be satisfied. ■

Generating ψ'. The procedure generates an equation ψ' by replacing each active variable $E[x_i]$ in $\psi_\star$ with the variable $F[y_i]$. If F is assigned a correct encoding then ψ' simulates ψ, as stated in the following claim.

CLAIM 5.20. *Let $\mathcal{A}$ be an assignment for Ψ and for F. Let f be the $[sD-1,1]$-degree LDF defined by $f(x) = \sum_{i=1}^{sD} \kappa_x(a_i)\mathcal{A}(F[y_i])$, as in Claim Claim 5.18, and assume that $\mathcal{A}(E) \circ \Gamma = f$. In that case ψ is satisfied by $\mathcal{A}$ if and only if ψ' is satisfied by it.*

PROOF. According to the definition of Γ, $\Gamma(a_i) = x_i$ for $i = 1, \ldots, D$. Hence it follows from the assumption that $\mathcal{A}(E[x_i]) = \mathcal{A}(E[\Gamma(a_i)]) = f(a_i)$ for every i, $1 \leq i \leq D$. But according to Claim Claim 5.18 $f(a_i) = \mathcal{A}(F[y_i])$ for every i. Therefore the assignment of every active variable $E[x_i]$ equals the assignment of $F[y_i]$. The claim immediately follows. ■

Generating $\mathcal{E}_\psi$. The set $\mathcal{E}_\psi$ is composed of all the conjunctions of ψ' and an equation $\chi[x]$ of the curve-verifier.

Proof of correctness. The domain F that is generated by the curve-extension representation-procedure has the parameters required by Lemma Lemma 5.8, and the conjunctions of $\mathcal{E}_\psi$ are all condensed. It is also easy to verify that the curve-extension representation-procedure takes polynomial time in the size of ψ and in $|F|$. To complete the proof of Lemma Lemma 5.8 it remains to show that $\mathcal{E}_\psi$ has the extension and restriction properties. The other properties required of a representation-procedure are obvious.

- Extension: Let $\mathcal{A}$ be a good assignment for the variables of Ψ that satisfies ψ. We extend $\mathcal{A}$ by assigning an $s(F)$-degree LDF to F such that all the conjunctions of $\mathcal{E}_\psi$ are satisfied. The LDF g, to be assigned to F, is defined as follows. First, let $g(y_i) \doteq \mathcal{A}(E[\Gamma(a_i)])$ for $i = 1, \ldots, sD$. Since all the y_i's are contained in $\mathcal{H}^{d(F)}$, there exists an extension of g to an LDF over $\mathcal{F}^{d(F)}$ of degree at most $s(F)/d(F)$ in each variable. The total degree of this g is hence at most $s(F)$. We assign g to F. Then $\mathcal{A}(F[y_i]) = \mathcal{A}(E[\Gamma(a_i)])$ for every i, and so Claim Claim 5.20 implies that ψ' is satisfied.

  Let f be the $[sD-1,1]$-LDF defined by $f \doteq \mathcal{A}(E) \circ \Gamma$. Then by Claim Claim 5.18,

$$\forall\, x \in \mathcal{F} \qquad f(x) = \sum_{i=1}^{sD} \kappa_x(a_i) f(a_i) = \sum_{i=1}^{sD} \kappa_x(a_i)\mathcal{A}(E[\Gamma(a_i)]) \\ = \sum_{i=1}^{sD} \kappa_x(a_i)\mathcal{A}(F[y_i])$$

  where the coefficients $\kappa_x(a_i)$ are as in Claim Claim 5.18. It hence follows that the curve-verifier equations are all satisfied by the extended $\mathcal{A}$. Since $\mathcal{E}_\psi$ consists of conjunctions of ψ' and equations of the curve-verifier, we have that all of its conjunctions are satisfied.

- Restriction: Let $\mathcal{A}$ be a feasible assignment for the variables of Ψ and for F, and assume that at least an $|\mathcal{F}|^{-1/2}$ fraction of the conjunctions in $\mathcal{E}_\psi$ are satisfied by $\mathcal{A}$. Since ψ' appears in every conjunction of $\mathcal{E}_\psi$, ψ' is satisfied, and at least an $|\mathcal{F}|^{-1/2}$ fraction of the curve-verifier equations are satisfied as well.

  Define an $(sD-1)$-degree LDF f by

$$\forall\, x \in \mathcal{F} \qquad f(x) \doteq \sum_{i=1}^{sD} \kappa_x(a_i)\mathcal{A}(F[y_i])$$

  where the coefficients $\kappa_x(a_i)$ are as in Claim Claim 5.18. It follows from Claim Claim 5.19 that $\mathcal{A}(E) \circ \Gamma = f$. Since ψ' is satisfied, it then follows from Claim Claim 5.20 that ψ is satisfied as well, thereby proving the restriction property.

5.9. The Linearization Representation-Procedure. In this subsection we show the Linearization representation-procedure. It is the final representation-procedure used in the sequence of transformations, resulting in a system of a constant active-dependency parameter.

The linearization representation-procedure is similar to the curve-extension. When applied to an equation ψ, it uses the newly generated domain to encode the restriction of the assignment of $\mathrm{Dom}_\star(\psi)$ to a curve that contains the active variables of ψ. The curve-extension representation-procedure encoded directly the assignment at only some points of the curve; to obtain other evaluations it applied interpolation by computing an appropriate linear-combinations over the encoded values.

The linearization representation-procedure applies a method of Arora *et al.* (1998), using the newly generated domain to encode all linear-combinations of these values. Hence each curve-verifier equation requires just one active variable of the new domain. Also, since the active part of ψ is a linear-combination of variables associated with points on the curve, $\psi_\star$ can also be evaluated using one access to the new domain.

The linearization representation-procedure. We now describe the linearization representation-procedure. We fix the notations $E \doteq \mathrm{Dom}_\star(\psi)$, $r \doteq r(E)$, $s \doteq s(E)$, $d \doteq d(E)$, and $D \doteq \mathrm{D}_\star(\psi)$. The linearization representation-procedure first generates a new domain F with parameters as stated in Lemma Lemma 5.9, that is

$$r(F) = r, \;\; s(F) = 1, \quad \text{and} \quad d(F) = sD$$

In each conjunction in $\mathcal{E}_\psi$ there will be an equation ψ', that is derived by replacing the active part of ψ with a variable of F that encodes it. Another equation in each conjunction is taken from a set of equations called a *linearization-verifier*, that are not satisfied unless F is assigned a homogeneous linear-LDF. As in the curve-extension representation-procedure, the last equation in each conjunction is taken from a set called the *curve-verifier*, whose equations are not satisfied unless the assignment of F is a correct encoding.

Generation of the linearization-verifier. The linearization-verifier has one equation $\chi[y,t]$ for every $y \in \mathcal{F}^{d(F)}$ and $t \in \mathcal{F}$:

$$\chi[y,t] \;: \qquad tF[y] = F[ty]$$

CLAIM 5.21. *unless F is assigned a linear homogeneous LDF, the number of satisfied equations of the linearization-verifier is $\Omega(|\mathcal{F}|^{1/2})$.*

PROOF. We consider the polynomials $g(t,y) = F[ty]$ and $h(t,y) = tF[y]$, both on one more variable than F. Then either $g \equiv h$ or they agree in at most $2r(F)/|\mathcal{F}|$ places (since the degree of g is at most $2r(F)$ and the degree of h is at most $r(F)+1$. In the second case we are done, since the size of the agreement is less than $O(|\mathcal{F}|^{1/2})$. In the first case the polynomial $F(ty)$ coincides with $tF(y)$. This polynomial equation, by comparing the individual terms, implies that F is linear and homogeneous. ■

The curve Γ. Write the active part of ψ as

$$\psi_\star \;: \qquad \sum_{i=1}^{D} \alpha_i E[x_i] \tag{5.22}$$

As in the curve-extension representation-procedure, we define a curve $\Gamma : \mathcal{F} \to \mathcal{F}^d$ which goes through the points associated with the active variables of ψ.

Definition 5.23 (the curve of ψ). *Let $\mathcal{H}_{sD-1} = \{a_1, \ldots, a_{sD}\}$ be an arbitrary subset of $\mathcal{F}$. Define $\Gamma : \mathcal{F} \to \mathcal{F}^d$ to be the vector of $(D-1)$-degree polynomial functions satisfying*

$$\forall\, 1 \le i \le D \qquad \Gamma(a_i) = x_i$$

Given an assignment $\mathcal{A}$ for E, the assignment of F is used as an encoding of $\mathcal{A}(E) \circ \Gamma$. Unlike in the curve-extension representation-procedure, the correct encoding here is a linear-homogeneous LDF.

The encoding. The procedure generates a curve-verifier, whose equations are only satisfied if the assignment of F is the correct encoding of $\mathcal{A}(E) \circ \Gamma$. To define what the correct encoding is, suppose E is assigned an s-degree LDF g. The LDF $g \circ \Gamma : \mathcal{F} \to \mathcal{F}$, which is to be encoded by the assignment of F, has degree at most $sD - 1$. Its encoding is the following linear-homogeneous LDF, L_g:

$$\forall\, (y_1, \ldots, y_{sD}) \in \mathcal{F}^{sD} \qquad L_g(y_1, \ldots, y_{sD}) \doteq \sum_{i=1}^{sD} y_i g(\Gamma(a_i)) \tag{5.24}$$

The next claim shows how $g \circ \Gamma$ can be reconstructed, given L_g.

Claim 5.25 (linearizing-interpolation). *Let g be an $[s,d]$-LDF, and for $i = 1, \ldots, sD$, let γ_i be the $[sD-1, 1]$-LDF satisfying $\gamma_i(a_i) = 1$ and $\gamma_i(a_j) = 0$ for every $j \neq i$.*

Then the polynomial vector function $\widehat{\gamma} = (\gamma_1, \ldots, \gamma_{sD})$ satisfies $L_g \circ \widehat{\gamma} = g \circ \Gamma$, where L_g is as defined in Equation (5.24).

Proof. L_g is linear, hence $L_g \circ \widehat{\gamma}$ is of degree at most $sD - 1$. Since it follows from the definition of L_g that $L_g \circ \widehat{\gamma}(a_i) = g(\Gamma(a_i))$ for $i = 1, \ldots, sD$, we obtain that $L_g \circ \widehat{\gamma} = g \circ \Gamma$ (also recall that $g \circ \Gamma$ is of degree at most $sD - 1$). ■

Generating the curve-verifier. It follows from Claim Claim 5.25 that if an assignment $\mathcal{A}$ assigns a good LDF to E and assigns its encoding to F, then $\mathcal{A}(F[\widehat{\gamma}(x)]) = \mathcal{A}(E[\Gamma(x)])$ for every $x \in \mathcal{F}$. To verify that F is assigned a correct encoding, the representation-procedure generates one equation $\chi[x]$ in the curve-verifier for every $x \in \mathcal{F}$ as follows:

$$\chi[x] : \qquad F[\widehat{\gamma}(x)] = E[\Gamma(x)]$$

where the vector-function $\widehat{\gamma}$ is as defined in Claim Claim 5.25.

The following claim shows that indeed the curve-verifier equations are not satisfied unless the assignment for F is a correct encoding, in the sense that $\mathcal{A}(F) \circ \widehat{\gamma} = \mathcal{A}(E) \circ \Gamma$. It is assumed that F is assigned a linear LDF, since otherwise the linearization-verifier equations cannot be satisfied.

Claim 5.26. *Let $\mathcal{A}$ be a feasible assignment for E and F, assigning a linear homogeneous LDF to F. Then less than an $|\mathcal{F}|^{-1/2}$ fraction of the curve-verifier equations are satisfied unless $\mathcal{A}(F) \circ \widehat{\gamma} = \mathcal{A}(E) \circ \Gamma$. In the latter case all of the curve-verifier equations are satisfied, and moreover,*

$$\forall\, (y_1, \ldots, y_{sD}) \in \mathcal{F}^{sD} \qquad \mathcal{A}(F[y_1, \ldots, y_{sD}]) \doteq \sum_{i=1}^{sD} y_i \mathcal{A}(E[\Gamma(a_i)]) \tag{5.27}$$

Proof. Assume that at least an $|\mathcal{F}|^{-1/2}$ fraction of the equations $\chi[x]$ are satisfied. This means that $\mathcal{A}(F[\widehat{\gamma}(x)]) = \mathcal{A}(E[\Gamma(x)])$ for at least an $|\mathcal{F}|^{-1/2}$ fraction of the x's. Since $\mathcal{A}(F) \circ \widehat{\gamma}$ is an LDF of degree at most $sD - 1 < |\mathcal{F}|^{-1/2}$ and $\mathcal{A}(E) \circ \Gamma$ is an LDF of degree at most $r(D-1) < |\mathcal{F}|^{-1/2}$, this implies that $\mathcal{A}(F) \circ \widehat{\gamma} = \mathcal{A}(E) \circ \Gamma$.

In this case it follows that $\mathcal{A}(F[\widehat{\gamma}(x)]) = \mathcal{A}(E[\Gamma(x)])$ for every $x \in \mathcal{F}$, and hence the equation $\chi[x]$ is satisfied. In addition, since in particular $\mathcal{A}(F[\widehat{\gamma}(a_i)]) = \mathcal{A}(E[\Gamma(a_i)])$ for every a_i, $i = 1, \ldots, sD$, one obtains from the definitions of Γ and γ that Equation (5.27) holds for all unit vectors. By the linearity of the assignment for F, it follows that Equation (5.27) holds for all points. ∎

Generating ψ'. The procedure now generates the equation ψ' from ψ by replacing its active part by just one variable. Specifically, ψ' is obtained from ψ by removing $\psi_\star$ and replacing it by $F[y_*]$, where $y_* \doteq (\alpha_1, \alpha_2, \ldots, \alpha_D, 0, 0, \ldots, 0)$, and the α_i's are the coefficients that appear in the active part of ψ (see Equation (5.22)). The rational behind this replacement is explained by the following immediate claim.

Claim 5.28. *Let $\mathcal{A}$ be an assignment for E and F which satisfies Equation (5.27) in Claim Claim 5.26. Then the value of $\mathcal{A}(F[y_*])$ is the same as the evaluation of $\psi_\star$.*

Generating $\mathcal{E}_\psi$. The linearization representation-procedure constructs the set of conjunctions $\mathcal{E}_\psi$ as follows. For each triplet (x, y, t) where $x, t \in \mathcal{F}$ and $y \in \mathcal{F}^{d(F)}$, $\mathcal{E}_\psi$ will have the conjunction of ψ', the curve-verifier equation $\chi[x]$, and the linearization-verifier equation $\chi[y, t]$.

Correctness of the algorithm. The domain F that is generated by the linearization representation-procedure has the required parameters, and it is easy to verify that the running time is polynomial in $|F|$. To complete the proof of Lemma Lemma 5.9 let us show that $\mathcal{E}_\psi$ has the extension and restriction properties, as the other required properties are obvious.

- Extension: Let $\mathcal{A}$ be a good assignment for the variables of Ψ, that satisfies ψ. Let g be the $[s, d]$-LDF assigned to E, and extend $\mathcal{A}$ to F by assigning L_g to it. We need to show that the extended $\mathcal{A}$ satisfies the conjunctions of $\mathcal{E}_\psi$. According to the construction, it is enough to show that ψ' is satisfied and that the curve-verifier and linearization-verifier equations are satisfied as well.

Since $\mathcal{A}(F) = L_g$ is a linear-homogeneous LDF, the linearization-verifier equations are satisfied by Claim Claim 5.21. Also, g is an s-degree LDF, so by Claim Claim 5.25

$$\mathcal{A}(F) \circ \widehat{\gamma} = L_g \circ \widehat{\gamma} = g \circ \Gamma = \mathcal{A}(E) \circ \Gamma \ .$$

From Claim Claim 5.26 we thus have that all of the curve-verifier equations are satisfied. Moreover, Claim Claim 5.26 also implies that Equation (5.27) is satisfied, from which, by Claim Claim 5.28, it follows that the evaluations of $\psi_\star$ and of $F[y_*]$ are equal. Since ψ is satisfied, and ψ' differs from ψ only in the substitution of $\psi_\star$ by $F[y_*]$, we obtain that ψ' is satisfied as well.

- Restriction: Let $\mathcal{A}$ be a feasible assignment for the variables of Ψ and for F. We assume that these assignments satisfy at least an $|\mathcal{F}|^{-1/2}$ fraction of the conjunctions in $\mathcal{E}_\psi$. This implies that ψ' is satisfied, and that at least an $|\mathcal{F}|^{-1/2}$ fraction of the curve-verifier equations are satisfied, as well as an $|\mathcal{F}|^{-1/2}$ fraction of the linearization-verifier equations. Let us prove that ψ is satisfied.

  Since at least an $|\mathcal{F}|^{-1/2}$ fraction of the linearization-verifier are satisfied, we gather from Claim Claim 5.21 that F is assigned a linear-homogeneous LDF. Now Claim Claim 5.26 implies that Equation (5.27) holds, and therefore by Claim Claim 5.28, $\psi_\star$ has the same evaluation as $F[y_*]$. Since ψ' is satisfied, this implies that ψ is satisfied as well.

Acknowledgements

We wish to thank the anonymous referee for the careful reading of this manuscript and for many helpful comments.

References

M. Alekhnovich, S. Buss, S. Moran & T. Pitassi (1998). Minimum Propositional Proof Length is NP-Hard to Linearly Approximate. Manuscript.

Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan & Mario Szegedy (1998). Proof verification and the hardness of approximation problems. *Journal of the ACM* **45**(3), 501–555. ISSN 0004-5411.

Sanjeev Arora & Shmuel Safra (1998). Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM* **45**(1), 70–122. ISSN 0004-5411.

Sanjeev Arora & Madhu Sudan (1997). Improved Low Degree Testing and its Applications. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, 485–495. El Paso, Texas.

L. Babai, L. Fortnow & C. Lund (1991). Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity* **1**, 3–40.

M. Bellare, S. Goldwasser, C. Lund & A. Russell (1993). Efficient Multi-Prover Interactive Proofs with Applications to Approximation Problems. In *Proc. 25th ACM Symp. on Theory of Computing*, 113–131.

I. Dinur, E. Fischer, G. Kindler, R. Raz & S. Safra (1999). PCP Characterizations of NP: Towards a Polynomially-Small Error-Probability. In *Proc. 31th ACM Symp. on Theory of Computing.*

I. Dinur & S. Safra (1998). Monotone-Minimum-Satisfying Assignment is NP-hard for Almost Polynomial Factors. Manuscript.

J. Hastad, R. Phillips & S. Safra (1993). A well-characterized approximation problem. *Information Processing Letters* **47**, 301–305.

Carsten Lund & Mihalis Yannakakis (1994). On the Hardness of Approximating Minimization Problems. *Journal of the ACM* **41**(5), 960–981.

R. Raz & S. Safra (1997). 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*, 475–484.

Ran Raz (1998). A Parallel Repetition Theorem. *SIAM Journal on Computing* **27**(3), 763–803.

Manuscript received 27 October 2004

Irit Dinur
Department of Math and Computer Science
The Weizmann Institute of Science
Israel
irit.dinur@weizmann.ac.il

Eldar Fischer
The Faculty of Computer Science
Technion
eldar@cs.technion.ac.il
Israel

Guy Kindler
School of Computer Science
The Hebrew University
Israel
gkindler@cs.huji.ac.il

Ran Raz
Department of Math and Computer Science
Weizmann Institute of Science
Israel
ran.raz@weizmann.ac.il

Shmuel Safra
School of Computer Science
Tel Aviv University
Israel
safra@post.tau.ac.il