Innovations in Computer Science 2011

Ideal Forms of Coppersmith's Theorem and Guruswami-Sudan List Decoding

Henry Cohn* Nadia Heninger*
*Microsoft Research New England, One Memorial Drive, Cambridge, MA 02142
*Department of Computer Science, Princeton University, Princeton, NJ 08540
cohn@microsoft.com nadiah@cs.princeton.edu

Abstract: We develop a framework for solving polynomial equations with size constraints on solutions. We obtain our results by showing how to apply a technique of Coppersmith for finding small solutions of polynomial equations modulo integers to analogous problems over polynomial rings, number fields, and function fields. This gives us a unified view of several problems arising naturally in cryptography, coding theory, and the study of lattices. We give (1) a polynomial-time algorithm for finding small solutions of polynomial equations modulo ideals over algebraic number fields, (2) a faster variant of the Guruswami-Sudan algorithm for list decoding of Reed-Solomon codes, and (3) an algorithm for list decoding of algebraic-geometric codes that handles both single-point and multi-point codes. Coppersmith's algorithm uses lattice basis reduction to find a short vector in a carefully constructed lattice; powerful analogies from algebraic number theory allow us to identify the appropriate analogue of a lattice in each case and provide efficient algorithms to find a suitably short vector, thus allowing us to give completely parallel proofs of the above theorems.

Keywords: Coppersmith's theorem, list decoding, lattice basis, reduction, cryptanalysis, coding theory.

1 Introduction

Many important problems in areas ranging from cryptanalysis to coding theory amount to solving polynomial equations with side constraints or partial information about the solutions.

One of the most important cases is solving equations given size bounds on the solutions. Coppersmith's algorithm is a celebrated technique for finding small solutions to polynomial equations modulo integers, and it has many important applications in cryptography, particularly in the cryptanalysis of RSA.

In this paper, we show how the ideas of Coppersmith's theorem can be extended to a more general framework encompassing the original number-theoretic problem, list decoding of Reed-Solomon and algebraic-geometric codes, and the problem of finding solutions to polynomial equations modulo ideals in rings of algebraic integers. These seemingly different problems are all perfectly analogous when viewed from the perspective of algebraic number theory.

Coppersmith's algorithm provides a key example of the power of lattice basis reduction. In order to extend the method beyond the integers, we examine the analogous structures for polynomial rings, number fields, and function fields. Ideals over number fields have a natural embedding into a lattice, and thus we can find a short vector simply by applying the LLL algorithm to this canonical embedding. In contrast to integer lattices, it turns out that lattice basis reduction is much easier over a lattice of polynomials, and in fact a shortest vector can always be found in polynomial time. Recasting the list decoding problem in this framework allows us to take advantage of very efficient reduction algorithms and thus achieve the fastest known list decoding algorithm for Reed-Solomon codes.

To extend this approach to function fields, we must overcome certain technical difficulties. In addition, we prove a much more general result about finding short vectors under arbitrary non-Archimedean norms, which may have further applications beyond list decoding of algebraic-geometric codes. As an illustration of the generality of our approach, we give the first list decoding algorithm that works for all algebraic-geometric codes, not just those defined using a single-point divisor.

In the remainder of the introduction, we set up our

framework with a brief review of Coppersmith's theorem, and then state our theorems on polynomial rings, number fields, and function fields.

1.1 Coppersmith's theorem

The following extension of Coppersmith's theorem [9]was developed by Howgrave-Graham [20] and May [29].

Theorem 1.1([9,20,29]). *Let $f(x)$ be a monic polynomial of degree d with coefficients modulo an integer $N > 1$, and suppose $0 < \beta \leq 1$. In time polynomial in $\log N$ and d, one can find all integers w such that*

$$|w| \leqslant N^{\beta^2/d}$$

and

$$\gcd(f(w), N) \geq N^{\beta}.$$

Note that when $\beta = 1$, this amounts to finding all sufficiently small solutions of $f(w) \equiv 0 \pmod{N}$, and the general theorem amounts to solving $f(w) \equiv 0 \pmod{B}$, where B is a large factor of N.

We give a brief example to illustrate the power of this theorem in cryptography [9,20]. Imagine that an adversary has obtained through a side-channel attack some knowledge about one of the prime factors p of an RSA modulus $N = pq$, for example some of its most significant bits. We denote this known quantity by r. Then we may write $p = r + w$, where the bound on w depends on how many bits of p are known. Suppose more than half of the bits have leaked, i.e., $0 \leq w \leq N^{1/4-o(1)}$ (we assume, as is typical, that p and q are both $N^{1/2+o(1)}$). Now let $f(x) = x + r$ and $\beta = 1/2 + o(1)$. Theorem 1.1 tells us that we can in polynomial time learn w, and hence p, thereby factoring N.

Further applications of this theorem in cryptography include other partial key recovery attacks against RSA [5,7], blomer:rsa, attacks on stereotyped messages and improper padding [9], and the proof of security for the RSA-OAEP+ padding scheme [35]. See [30] for many other applications.

It is remarkable that Theorem 1.1 allows us to solve polynomial equations modulo N without knowing the factorization of N, and this fact is critical for the cryptanalytic applications. However, even if one already has the factorization, Theorem 1.1 remains nontrivial if N has many prime factors.

To solve an equation modulo a composite number, one generally solves the equation modulo each prime power factor of the modulus and uses the Chinese remainder theorem to construct solutions for the original modulus. (Recall that modulo a prime, such equations can be solved in polynomial time, and we can use Hensel's lemma to lift the solutions to prime power moduli.) The number of possible solutions can be exponential in the number of prime factors, in which case it is infeasible to enumerate all of the roots and then select those that are within the desired range. In fact, the problem of determining whether there is a root in an arbitrary given interval is NP-complete [27]. Of course, if N has only two prime factors, then there can be only d^2 solutions modulo N, but our methods are incapable of distinguishing between numbers with two or many prime factors.

It is not even obvious that the number of roots modulo N of size at most $N^{1/d}$ is polynomially bounded. From this perspective, the exponent $1/d$ is optimal without further assumptions, because $f(x) = x^d$ will have exponentially many roots modulo $N = k^d$ of absolute value at most $N^{1/d+\varepsilon}$ (specifically, the $2N^{\varepsilon}$ such multiples of k). Theorem 1.1 can be seen as a constructive bound on the number of solutions. See [10] for further discussion of this argument and [23] for non-constructive bounds.

1.2 A polynomial analogue

To introduce our analogies, we will begin with the simplest and most familiar case: polynomials.

There is an important analogy in number theory between the ring $\mathbb{Z}$ of integers and the ring $F[z]$ of univariate polynomials over a field F. To formulate the analogue of Coppersmith's theorem, one just needs to recognize that the degree of a polynomial is the appropriate measure of its size. Thus, the polynomial version of Coppersmith's theorem should involve finding low-degree solutions of polynomial equations over $F[z]$ modulo a polynomial $p(z)$. That is, given a polynomial $f(x) = \sum_{i=0}^{d} f_i(z)x^i$ with coefficients $f_i(z) \in F[z]$, we seek low-degree polynomials $w(z) \in F[z]$ such that $f(w(z)) \equiv 0 \pmod{p(z)}$.

In the following theorem, we assume that we can efficiently represent and manipulate elements of F, and that we can find roots in $F[z]$ of polynomials over $F[z]$. For example, that holds if we can factor bivariate polynomials over F in polynomial time. This assumption holds for many fields, including $\mathbb{Q}$ and even number fields [24] as well as all finite fields [16] (with a ran-

domized algorithm in the latter case).

Theorem 1.2. *Let $f(x)$ be a monic polynomial in x of degree d over $F[z]$ with coefficients modulo $p(z)$, where $\deg_z p(z) = n > 0$. In polynomial time, for $0 < \beta \leq 1$, one can find all $w(z) \in F[z]$ such that*

$$\deg_z w(z) < \beta^2 n/d$$

and

$$\deg_z \gcd(f(w(z)), p(z)) \geq \beta n.$$

In the case when $p(z)$ factors completely into linear factors, this theorem is equivalent to the influential Guruswami-Sudan theorem on list decoding of Reed-Solomon codes [19]. See Section 4.1 for the details of the equivalence. The above statement of Theorem 1.2, as well as the extension to higher-degree irreducible factors, appear to be new.

It has long been recognized that the Coppersmith and Guruswami-Sudan theorems are in some way analogous, although we are unaware of any previous, comparably explicit statement of the analogy. Boneh used Coppersmith's theorem in work on Chinese remainder theorem codes inspired by the Guruswami-Sudan theorem [6], and in a brief aside in the middle of [3], Bernstein noted that the Guruswami-Sudan theorem is the polynomial analogue of a related theorem of Coppersmith, Howgrave-Graham, and Nagaraj [11]. See also [18] for a general ideal-theoretic setting for coding theory, and [36] for a survey of relationships between list decoding and number-theoretic codes.

1.3 Number fields

A number field is a finite extension of the field $\mathbb{Q}$ of rational numbers. Thus it is natural to investigate how a statement over the rationals, the simplest number field, extends to more general number fields. We extend our analogy by adapting Coppersmith's theorem to the number field case.

Every number field K is of the form

$$\begin{aligned} K &= \mathbb{Q}(\alpha) \\ &= \{a_0 + a_1\alpha + \cdots + a_{n-1}\alpha^{n-1} : a_i \in \mathbb{Q}\}, \end{aligned}$$

where α is an algebraic number of degree n (i.e., a root of an irreducible polynomial of degree n over $\mathbb{Q}$). The degree of K is defined to be n. Within K, there is a ring $\mathcal{O}_K$ called the ring of algebraic integers in K. It plays the same role within the field K as the ring $\mathbb{Z}$ of integers plays within $\mathbb{Q}$.

In $\mathcal{O}_K$, we study the solutions of polynomial equations modulo ideals, the analogue of working modulo integers in $\mathbb{Z}$. (Recall that an ideal is a non-empty subset closed under addition and under multiplication by arbitrary elements of a ring; intuitively, it is a subset modulo which one can reduce ring elements.) Additionally, elements of $\mathcal{O}_K$ have n absolute values coming from n embeddings of K into $\mathbb{C}$, and to formulate our theorem we must bound them all. (See Section 5.)

The number field analogue of Coppersmith's theorem is as follows:

Theorem 1.3. *Let K be a number field of degree n with ring of integers $\mathcal{O}_K$, $f(x) \in \mathcal{O}_K[x]$ a monic polynomial of degree d, and $I \subsetneq \mathcal{O}_K$ an ideal in $\mathcal{O}_K$. Assume that we are given $\mathcal{O}_K$ and I explicitly by integral bases. For $0 < \beta \leq 1$ and $\lambda_1, \ldots, \lambda_n > 0$, in time polynomial in the input length and exponential in n^2 we can find all $w \in \mathcal{O}_K$ with $|w|_i < \lambda_i$ such that*

$$N(\gcd(f(w)\mathcal{O}_K, I)) > N(I)^\beta,$$

provided that

$$\prod_i \lambda_i < N(I)^{\beta^2/d}.$$

Furthermore, in polynomial time we can find all such w provided that

$$\prod_i \lambda_i < (2 + o(1))^{-n^2/2} N(I)^{\beta^2/d}.$$

Equivalently, we can find small solutions of equations $f(x) \equiv 0 \pmod{J}$, where the ideal J is a large divisor of I. Using improved lattice basis reduction algorithms [2] we can achieve slightly subexponential behavior in n^2. Note also that $\gcd(f(w)\mathcal{O}_K, I)$ is the largest ideal that contains both the principal ideal $f(w)\mathcal{O}_K$ and I; in other words, it is their sum $f(w)\mathcal{O}_K + I$.

Section 5 explains the techniques required to prove Theorem 1.3; see [8] for a full proof.

Recently, Peikert and Rosen [32] and Lyubashevsky, Peikert, and Regev [26] developed lattice-based cryptographic schemes using lattices representing the canonical embeddings of ideals in number fields. As a special case, Theorem 1.3 can be used to solve certain cases of the bounded-distance decoding problem for such lattices, and improving our approximation factor from $(2 + o(1))^{-n^2/2}$ to $2^{-n}\sqrt{|\Delta_K|}$, where Δ_K is the discriminant of K, would solve the problem in general; see [8] for more details.

In addition, number fields have many applications to purely classical problems, the most prominent example being the number field sieve factoring algorithm. All sieve algorithms require generating smooth numbers, and in this context Boneh [6] showed how to use Coppersmith's theorem to find smooth integer solutions of polynomials in short intervals. Using Theorem 1.3 analogously, one can do the same over number fields.

1.4 Function fields

Algebraic number theorists have developed a more sophisticated version of the analogy between the ring of integers and polynomial rings. In this analogy, the analogues of number fields are called function fields; they are the fields of rational functions on algebraic curves over finite fields. The parallels between number fields and function fields are truly astonishing, and this analogy has played a crucial role in the development of number theory over the last century.

We now complete the analogy by extending Coppersmith's theorem to the function field case. See [8] for a review of the setting and notation.

Theorem 1.4. *Let $\mathcal{X}$ be a smooth, projective, absolutely irreducible algebraic curve over $\mathbb{F}_q$, and let K be its function field over $\mathbb{F}_q$. Let D be a divisor on $\mathcal{X}$ whose support* $\operatorname{supp}(D)$ *is contained in the $\mathbb{F}_q$-rational points $\mathcal{X}(\mathbb{F}_q)$, let S be a subset of $\mathcal{X}(\mathbb{F}_q)$ that properly contains* $\operatorname{supp}(D)$*, let $\mathcal{O}_S$ be the subring of K consisting of functions with poles only in S, and let $\mathcal{L}(D)$ be the Riemann-Roch space*

$$\mathcal{L}(D) = \{0\} \cup \{f \in K^* : (f) + D \succeq 0\}.$$

Let $f(x) \in \mathcal{O}_S[x]$ be a monic polynomial of degree d, and let I be a proper ideal in $\mathcal{O}_S$.

Then in probabilistic polynomial time, we can find all $w \in \mathcal{L}(D)$ such that

$$N(\gcd(f(w)\mathcal{O}_S, I)) \geq N(I)^{\beta},$$

provided that

$$q^{\deg(D)} < N(I)^{\beta^2/d}.$$

In the case when S contains only a single point, the function field version of Coppersmith's theorem is equivalent to the Guruswami-Sudan theorem on list-decoding of algebraic-geometric codes, as outlined in [8]. The Guruswami-Sudan theorem and the earlier Shokrollahi-Wasserman theorem [34] are specialized to that case, which covers many but not all algebraic-geometric codes. Our theorem extends list decoding to the full range of such codes.

We assume that we can efficiently compute bases of Riemann-Roch spaces for divisors in $\mathcal{X}$. That can be done in many important cases (for example, for a smooth plane curve, or even one with ordinary multiple points [21]), and it is a reasonable assumption because even the encoding problem for algebraic-geometric codes requires a basis of a Riemann-Roch space. Note also that although our algorithm is probabilistic, it is guaranteed to give the correct solution in expected polynomial time; in other words, it is a "Las Vegas" algorithm.

Section 6 explains the techniques required to prove Theorem 1.4; see [8] for a full proof.

1.5 Analogies in number theory

The connections we have described are not isolated phenomena. Many theorems in number theory and algebraic geometry have parallel versions for the integers and for polynomial rings, or more generally for number fields and function fields, and translating statements or techniques between these settings can lead to valuable insights.

One particular advantage of this sort of arbitrage is that proving results for polynomial rings is usually easier. For example, the prime number theorem for $\mathbb{Z}$ is a deep theorem, but the analogue for the polynomial ring $\mathbb{F}_q[z]$ over a finite field is much simpler. It says that asymptotically a $1/n$ fraction of the q^n monic polynomials of degree n are irreducible, and in fact the error term is on the order of $q^{n/2}$ (see Lemma 14.38 in [15]). Proving a similarly strong version of the prime number theorem for $\mathbb{Z}$ would amount to proving the Riemann hypothesis. Similarly, the ABC conjecture for $\mathbb{Z}$ is a profound unsolved problem, while for polynomials rings it has an elementary proof [28].

Thus, polynomial rings are worlds in which many of the fondest dreams of mathematicians have come true. If a result cannot be proved in such a setting, then it is probably not even worth trying to prove it in $\mathbb{Z}$. If it can be proved for polynomial rings, then the techniques may not apply to the integers, but they often provide inspiration for how a proof might work if technical obstacles can be overcome.

Similarly, in computer science many computational problems that appear to be hard for integers are tractable for polynomials. For example, factoring polynomials can be done in polynomial time for many fields, while for the integers the problem seems to be hard. The polynomial analogue of the shortest vector problem for lattices can be solved exactly in polynomial time [14], while for integer lattices the problem is NP-hard [1]. This difference in the difficulty of lattice problems is at the root of the poor running time in Theorem 1.3 for number fields of high degree.

2 Preliminaries

One of the main steps in Coppersmith's theorem uses lattice basis reduction to find a short vector in a lattice. In this section, we will review preliminaries on integral lattices, and introduce the analogues that we will use in our generalizations.

2.1 Integer lattices

Recall that a *lattice* in $\mathbb{R}^m$ is a discrete subgroup of rank m. Equivalently, it is the set of integer linear combinations of a basis of $\mathbb{R}^m$.

The *determinant* $\det(L)$ of a lattice L is the absolute value of the determinant of any basis matrix; it is not difficult to show that it is independent of the choice of basis. One way to see why is that the determinant is the volume of the quotient $\mathbb{R}^m/L$, or equivalently the volume of a fundamental parallelotope.

One of the fundamental problems in lattice theory is finding short vectors in lattices, with respect to the ℓ_p norm

$$|v|_p = \left(\sum_{i=1}^{m} |v_i|^p \right)^{1/p}.$$

Most often we use the ℓ_2 norm, which is of course the usual Euclidean distance. The LLL lattice basis reduction algorithm [25] can be used to find a short vector in a lattice.

Theorem 2.1 ([25]). *Given a basis of a lattice L in $\mathbb{Q}^m$, a nonzero vector $v \in L$ satisfying*

$$|v|_2 \leqslant 2^{(m-1)/4} \det(L)^{1/m}$$

can be found in polynomial time.

Note that the LLL algorithm's input is a rational lattice, and the rationality plays an important role in the running time analysis. In the proof of Theorem 1.3, we must apply it to a lattice whose basis vectors are not in $\mathbb{Q}^m$; however, for our purposes using a close rational approximation suffices.

2.2 Polynomial lattices

If R is the polynomial ring $F[z]$ over a field F, then we define a *polynomial lattice* to be a free module over $F[z]$ of finite rank. A polynomial lattice will usually be generated by a basis of vectors whose coefficients are polynomials in z. Vectors in our polynomial lattice will be linear combinations of the basis vectors (where the coefficients are also polynomials in z).

An appropriate definition of the length (i.e., degree) of such a lattice vector is the maximum degree of its coordinates:

$$\deg_z(v_1(z), v_2(z), \ldots, v_m(z)) = \max_i \deg_z v_i(z). \tag{2.1}$$

This defines a non-Archimedean norm. In fact, for lattices with a norm defined as above, it is possible to find the exact shortest vector in polynomial time (see, for example, [14]).

Lattices of polynomials have been well studied because of their applications to the study of linear systems [22]. There are several notions of basis reduction for such lattices. A basis is *column-reduced* (or, as appropriate, *row-reduced*) if the degree of the determinant of the lattice (i.e., of a basis matrix) is equal to the sum of the degrees of its basis vectors. Such bases always contain a minimal vector for the lattice, and m-dimensional column reduction can be carried out in $m^{\omega+o(1)}D$ field operations [17], where ω is the exponent of matrix multiplication and D is the greatest degree occurring in the original basis of the lattice.

In particular, for an m-dimensional lattice L with the norm (2.1), the above algorithms are guaranteed to find a nonzero vector v for which

$$\deg v \leq \frac{1}{m} \deg \det L, \tag{2.2}$$

where $\det L$ denotes the determinant of a lattice basis.

2.3 Finding short vectors under general non-Archimedean norms

The above algorithms are specialized to norms defined by (2.1). In fact, for all non-Archimedean norms, one can find a vector satisfying the equivalent of (2.2)

in a lattice by solving a system of linear equations. Solving this system may be less efficient than a specialized algorithm, but it gives a general approach that works in polynomial time for any norm. See [8] for the details.

3 Coppersmith's theorem

We now review how Coppersmith's method works over the integers, as this provides a template for the techniques we will apply later. We will follow the exposition of May [30].

Let $f(x)$ be a monic univariate polynomial of degree d, and N an integer of potentially unknown factorization. We wish to find all small integers w such that $\gcd(f(w), N)$ is large.

To do so, we will choose some positive integer k (to be determined later) and look at integer combinations of the polynomials $x^j f(x)^i N^{k-i}$. If B divides both N and $f(w)$, then B^k will divide $w^j f(w)^i N^{k-i}$ and thus also any linear combination of such polynomials.

Let

$$Q(x) = \sum_{i,j} a_{i,j} x^j f(x)^i N^{k-i} = \sum_i q_i x^i,$$

for some coefficients $a_{i,j}$ and q_i to be determined. We will choose Q so that the small solutions to our original congruence become actual solutions of $Q(x) = 0$ in the integers. This will allow us to find w by factoring $Q(x)$ over the rationals. The construction of Q tells us that

$$Q(w) \equiv 0 \pmod{B^k}. \tag{3.1}$$

If in addition we have a lower bound N^β on the size of B, and we can show that

$$|Q(w)| < N^{\beta k} \leq B^k, \tag{3.2}$$

then $Q(w) = 0$ and we may find w by factoring Q. In fact, this observation tells us that we can find *all* such w in this way. A similar observation will appear in all of our proofs.

In the case of the integers, we introduce the bound $|w| < X$ on our roots, and the triangle inequality tells us that

$$|Q(w)| \leqslant \sum_i |q_i| X^i. \tag{3.3}$$

To finish the theorem, we will show that if X is sufficiently small, then we can choose Q so that its coefficients q_i satisfy

$$\sum_i |q_i| X^i < N^{\beta k}. \tag{3.4}$$

We are now ready to prove Coppersmith's theorem for the integers.

Proof of Theorem 1.1. Having outlined the general technique above, it remains to be shown that we can construct a polynomial $Q(x)$ whose coefficients satisfy the bound in (3.4).

The polynomial $Q(x)$ will be a linear combination of the polynomials

$$x^j f(x)^i N^{k-i} \quad \text{for} \quad 0 \leqslant i < k \text{ and } 0 \leqslant j < d$$

and

$$x^j f(x)^k \quad \text{for} \quad 0 \leqslant j < t.$$

The right-hand side of (3.3) is the ℓ_1 norm of the vector of coefficients of the polynomial $Q(xX)$, which in turn will be a linear combination of the polynomials $(xX)^j f(xX)^i N^{k-i}$. Finding our desired $Q(x)$ is thus equivalent to finding a suitably short vector in the lattice L spanned by the coefficient vectors of the polynomials $(xX)^j f(xX)^i N^{k-i}$.

To compute the determinant of this lattice, we can order the basis vectors by the degrees of the polynomials they represent to obtain an upper triangular matrix whose determinant is the product of the terms on the diagonal:

$$\begin{aligned}\det(L) &= \prod_{0 \leqslant i < dk+t} X^i \prod_{0 \leqslant j \leqslant k} N^{dj} \\ &= X^{(dk+t-1)(dk+t)/2} N^{dk(k+1)/2}.\end{aligned}$$

Set $m = dk + t$. We can use the LLL algorithm [25] to find a vector v whose ℓ_2 norm is bounded by

$$|v|_2 \leqslant 2^{(m-1)/4} \det(L)^{1/m}.$$

By Cauchy-Schwarz, $|v|_1 \leqslant \sqrt{m}\,|v|_2$, and hence whenever $|w| < X$,

$$|Q(w)| \leqslant \sqrt{m} 2^{(m-1)/4} \det(L)^{1/m}.$$

We assume $m \geq 7$, and use the weaker bound

$$|Q(w)| \leqslant 2^{(m-1)/2} \det(L)^{1/m}.$$

To prove inequality (3.2), we must show that

$$2^{(m-1)/2} \left(X^{m(m-1)/2} N^{dk(k+1)/2} \right)^{1/m} < N^{\beta k}.$$

This inequality is equivalent to

$$(2X)^{(m-1)/(2k)} N^{d(k+1)/(2m)} < N^\beta. \tag{3.5}$$

Applying Lemma 3.1 below with $\ell = \log 2X$ and $n = \log N$, we obtain parameters k and t such that (3.5) holds for

$$2X < N^{\beta^2/d-\varepsilon}.$$

To eliminate ε from the statement of the theorem, take $\varepsilon < \frac{1}{\log_2 N}$. Then our bound becomes $X \le \frac{1}{4}N^{\beta^2/d}$. We can divide the interval $[-N^{\beta^2/d}, N^{\beta^2/d}]$ into four intervals of width $2X$ and solve the problem for each interval by finding solutions for the polynomials $f(x-3X)$, $f(x-X)$, $f(x+X)$, and $f(x+3X)$. Thus, we achieve a bound of $X \le N^{\beta^2/d}$, as desired. □

We end with a brief lemma that will tell us how to optimize our parameters in equation (3.5).

Lemma 3.1. *The inequality* $\ell\frac{m-1}{2k} + nd\frac{k+1}{2m} < n\beta$ *is satisfied for* $\ell < n\left(\frac{\beta^2}{d} - \varepsilon\right)$, *any* $m \ge \left\lceil \frac{2\beta}{\varepsilon} \right\rceil$, *and* $k = \left\lfloor \frac{\beta m}{d} - 1 \right\rfloor$.

As intuition, note that if we set the two terms $\ell\frac{m-1}{2k}$ and $nd\frac{k+1}{2m}$ roughly equal to $\frac{n\beta}{2}$, then we have $\ell m^2 \approx ndk^2 \approx n\beta mk$ and hence $\ell \approx n\beta^2/d$. The proof amounts to making this precise.

Proof. It suffices to show that these values of m and k satisfy $n\left(\frac{\beta^2}{d} - \varepsilon\right)\frac{m-1}{2k} < \frac{n\beta}{2}$ and $nd\frac{k+1}{2m} \le \frac{n\beta}{2}$.

The first inequality is equivalent to $\frac{k}{m-1} > \frac{\beta}{d} - \frac{\varepsilon}{\beta}$. Similarly, the second is equivalent to $\frac{k+1}{m} \le \frac{\beta}{d}$. If we set $k = \left\lfloor \frac{\beta m}{d} - 1 \right\rfloor$, then $\frac{k+1}{m} \le \frac{\beta}{d}$, so the second inequality is satisfied. If in addition we take $m \ge \frac{2\beta}{\varepsilon}$, then $\frac{\varepsilon m}{\beta} \ge 2$ and hence $k > \frac{\beta m}{d} - 2 \ge \frac{\beta m}{d} - \frac{\varepsilon m}{\beta}$. It follows that $k\frac{m}{m-1} > \frac{\beta m}{d} - \frac{\varepsilon m}{\beta}$, which is equivalent to the first inequality. □

Note that improving the approximation factor for the length of the short lattice vector that we find will only improve the constants and running time of the theorem, but will not provide an asymptotic improvement to the bound $N^{\beta^2/d}$ on $|w|$.

4 Polynomials and Reed-Solomon list decoding

In this section, we show how to prove Theorem 1.2 using an approach analogous to that of the previous section. Guruswami and Sudan's technique for list decoding of Reed-Solomon codes [19] is similar in that it involves constructing a bivariate polynomial that vanishes to high order at particular points. To construct such a polynomial, they write each vanishing condition as a set of linear equations on the coefficients of the polynomial under construction. The linear equations can be solved to obtain the desired polynomial, and the polynomial factored to obtain its roots.

Similarly, the polynomials used in Coppersmith's method are constructed in order to vanish to high order, the condition ensured by equation (3.1). The conceptual difference is that this condition follows from the form of the lattice basis, rather than being imposed as linear constraints. With the right definition of lattice basis reduction in the polynomial setting, we can emulate the proof from the integer case.

We regard $f(x)$ as a polynomial in x with coefficients that are polynomials in the variable z. To prove Theorem 1.2, we would like to construct a polynomial $Q(x)$ over $F[z]$ from the polynomials $x^j f(x)^i p(x)^{k-i}$. If $b(z)$ divides both $p(z)$ and $f(w(z))$, then $b(z)^k$ divides $w(z)^j f(w(z))^i p(z)^{k-i}$ and thus also any linear combination of such polynomials.

Instead of an integer combination of these polynomials, we will allow coefficients that are polynomials in z. Let

$$Q(x) = \sum_{i,j} a_{i,j}(z) x^j f(x)^i p(z)^{k-i} = \sum_i q_i(z) x^i.$$

If we have an upper bound ℓ on the degree of our root $w(z)$, then the degree of $Q(w(z))$ will be

$$\deg_z Q(w(z)) \le \max_i \left(\deg_z q_i(z) + \ell i\right).$$

If similarly we have a lower bound $n\beta$ on the degree of $b(z)$, then if we know that both

$$Q(w(z)) \equiv 0 \pmod{b(z)^k}$$

and

$$\deg_z Q(w(z)) < n\beta k \le k \deg_z b(z), \qquad (4.1)$$

then we may conclude that

$$Q(w(z)) = 0.$$

To find a polynomial satisfying inequality (4.1), we construct a lattice of polynomials. From this point on, the analysis of the proof is almost exactly the same as in the integer case. The major differences are that an exact shortest vector in the lattice can be found, so there is no approximation factor, and that in place of the element X which bounds the size of the root, we use the polynomial z^ℓ.

See [8] for full proof.

We cannot achieve degree equal to $\beta^2 n/d$ (as opposed to strict inequality): for infinite F, the equation $x^d \equiv 0 \pmod{p(z)^d}$ has infinitely many solutions $x = c\,p(z)$.

4.1 Reed-Solomon list decoding and noisy polynomial interpolation

A Reed-Solomon code is determined by evaluating a polynomial $w(z) \in \mathbb{F}_q[z]$ of degree at most ℓ at a collection of distinct points $(x_1, \ldots, x_n)$ to obtain a codeword $(w(x_1), \ldots, w(x_n))$. In the Reed-Solomon decoding problem, we are provided with $(y_1, \ldots, y_n)$, where at most e values have changed, and we want to recover $w(z)$ by finding a polynomial of degree at most ℓ that fits at least $n-e$ points (x_i, y_i). Guruswami and Sudan [19] showed how to correct up to $e = n - \sqrt{n\ell}$ errors by providing a list of all possible decodings.

In the noisy polynomial interpolation problem, at each x_i a set $\{y_{i1}, \ldots, y_{id}\}$ of values is specified, and the goal is a low-degree polynomial passing through a point from each set. This problem has been proposed as a cryptographic primitive, for example by Naor and Pinkas [31], and studied by Bleichenbacher and Nguyen [4].

We can use Theorem 1.2 to solve both problems, and in particular recover the exact decoding rates of Guruswami-Sudan. Our input is a collection of points

$$\{(x_i, y_{ij}) : 1 \le i \le n, 1 \le j \le d\}.$$

We set $p(z) = \prod_i (z - x_i)$, and we define a monic polynomial $f(x)$ of degree d in x by

$$f(x) = \sum_{i=1}^{n} \prod_{j=1}^{d} (x - y_{ij}) \prod_{\substack{k=1 \\ k \neq i}}^{n} \frac{z - x_k}{x_i - x_k}.$$

We have constructed $f(x)$ by interpolation so that $f(x) \equiv \prod_j (x - y_{ij}) \pmod{(z - x_i)}$. Thus, $f(y_{ij}) = 0$ whenever $z = x_i$.

To correct e errors, we seek a polynomial $w(z)$ of degree at most ℓ such that for at least $n - e$ values of i, we have $w(x_i) = y_{ij}$ for some j. In other words, $f(w(z))$ must be divisible by at least $n - e$ factors $z - x_i$; that is, $\deg \gcd(f(w(z)), p(z)) \ge n - e$. By Theorem 1.2, we can solve this problem in polynomial time if $\ell < n(1 - e/n)^2/d$ (here $\beta = 1 - e/n$). That is equivalent to the Guruswami-Sudan bound $e < n - \sqrt{n\ell d}$.

4.2 Running time

The Guruswami-Sudan algorithm consists of two parts: constructing the polynomial $Q(x)$, and finding roots of $Q(x)$ in $\mathbb{F}_q[z]$. In this paper, we do not address the second part, but we improve the running time of the first part, which has been the bottleneck in the algorithm.

Emulating the analysis from [19], when $(\beta n)^2 = (1 + \delta)\ell n$, using the fastest row reduction algorithm (see Section (2.2)), the running time for our algorithm is $O(n/(\delta^{\omega+1+o(1)}))$. In the worst case the running time is $O(n^{2\omega+3+o(1)}d)$ field operations. With cubic-time matrix multiplication we achieve $O(n^9 d)$, and with fast matrix multiplication [12] we achieve $O(n^{7.752} d)$.

The original Guruswami-Sudan approach [19]requires roughly $O(n^3\delta^{-6})$ field operations with $d = 1$, or $O(n^{15})$ in the worst case. (The second part of their algorithm runs in time $O(n^{12})$, although there have been improvements since then [33].) The fastest previous algorithm proposed for this problem [37] apparently runs in worst case time $\widetilde{O}(n^8)$ when $d = 1$, although its running time analysis is only heuristic (see the footnote on page 13 of [37]). See [8] for more details.

5 Number fields

There are two major conceptual differences distinguishing the number field case from the integer case.

The first is that each element $\gamma \in K$ has n absolute values, corresponding to the n embeddings σ_i of K into $\mathbb{C}$: $|\gamma|_i = |\sigma_i(\gamma)|$. We cannot focus on a single absolute value, but must instead treat them all symmetrically. (Note that these absolute value functions are not necessarily all distinct, because pairs of complex conjugate embeddings lead to the same absolute value.)

The norm of γ is the product $N(\gamma) = \sigma_1(\gamma) \ldots \sigma_n(\gamma)$. It is a natural measure of size for elements of $\mathcal{O}_K$, but bounding the norm alone cannot suffice in Theorem 1.3. (It will not even guarantee that there are only finitely many solutions, since $\mathcal{O}_K$ typically has infinitely elements of norm 1.) Instead, we must bound each absolute value individually.

The second difference is that ideals in $\mathcal{O}_K$ are generally not principal (i.e., they do not have a single gen-

erator), and that means $\mathcal{O}_K$-lattices have a more complicated algebraic structure than $\mathbb{Z}$-lattices do. However, we can address this issue simultaneously with the first one, by using a 19th century construction due to Dedekind, called the canonical embedding. It uses all n complex embeddings of K to embed the $\mathcal{O}_K$-lattice as a $\mathbb{Z}$-lattice in a Euclidean space of n times the rank of the $\mathcal{O}_K$-lattice. We can then find a short vector by applying the LLL lattice basis reduction algorithm to the canonical embedding. This approach treats all the absolute values symmetrically and reduces to the more familiar case of $\mathbb{Z}$-lattices. (One can go further, and find not only a short vector but also a reduced pseudo-basis for a lattice over a number field [13], but a short vector suffices for our purposes.)

Once we have dealt with these technical obstacles, the proof of Theorem 1.3 follows the outline in Section 3. See [8] for the details of the construction and the full proof.

6 Function fields

As is often the case in number theory, we can prove stronger results about function fields than number fields: the exponential dependence on n^2 from Theorem 1.3 does not occur in Theorem 1.4, because lattice basis reduction is a more powerful technique in function fields. However, to take advantage of this power we must bring to bear results from algebraic geometry. (See[8] for more details.)

The absolute values on the function field K of a curve $\mathcal{X}$ correspond to points on $\mathcal{X}$, and they measure the order of vanishing of functions in K. If $f \in K$ and p is a point of $\mathcal{X}$, then $|f|_p$ is small if f vanishes to high order at p and large if f has a high order pole at p.

There is no direct analogue of the Archimedean absolute values from the number field case. Instead, we have more flexibility, and we can choose an arbitrary finite, nonempty subset S of the points on $\mathcal{X}$ to play the analogous role. This will be the set S from the statement of Theorem 1.4. We will restrict our attention to the ring $\mathcal{O}_S$ of functions whose poles are all contained in S, and we will measure size in $\mathcal{O}_S$ using the absolute values coming from the points in S.

The first obstacle to proving Theorem 1.4 is identifying the right sort of lattice to consider. For comparison, in the number field case, we use the canonical embedding to reduce from $\mathcal{O}_K$-lattices to $\mathbb{Z}$-lattices, because $\mathbb{Z}$ is a principal ideal domain and hence $\mathbb{Z}$-lattices are structurally simpler. In the function field case, $\mathbb{F}_q[z]$-lattices are the analogous structures, but $\mathbb{F}_q[z]$ has infinitely many embeddings as a subring of $\mathcal{O}_S$, while $\mathbb{Z}$ has only one embedding into $\mathcal{O}_K$. We must identify an embedding of a special sort, namely one that treats all the absolute values from points in S evenhandedly.

One we have identified a suitable embedding of $\mathbb{F}_q[z]$ into $\mathcal{O}_S$, we are faced with two more difficulties. The first is that we must consider lattices with more general non-Archimedean norms than those studied in the literature, because we must take into account all the absolute values from S, and the known algorithms for basis reduction no longer apply. However, we can prove the needed results in our more general framework.

The final difficulty comes from attempting to control the zeros and poles of functions in K. In the simplest function field, namely the rational function field $\mathbb{F}_q(z)$, we can specify the (finitely many) zeros and poles arbitrarily, subject to just one constraint, that the total order of all the zeros must equal that of the poles. For example, $z^2/(z-1)$ has a zero of order two at 0, a pole of order one at 1, and a pole of order one at ∞ (because the function grows linearly as z becomes large).

In more complicated function fields, there are additional subtle constraints on the zeros and poles, which interfere with our ability to construct auxiliary functions in the proof (specifically, the placeholder X that measures the size of the desired solution of the equation). We circumvent this difficulty by using a technique based on the strong approximation theorem. This allows us to control the behavior of a function at all the points in S except one, if we are willing to allowed uncontrolled behavior at that single point. Furthermore, we can uniformly bound the bad behavior at the uncontrolled point in terms of the genus of the function field. This approach introduces error terms into our bounds, but they are small enough that they disappear entirely in the final result.

Once we have overcome these obstacles, the proof of Theorem 1.4 is analogous to the previous results. See [8] for the full proof, as well as more background about function fields and their application to algebraic-geometric codes.

Acknowledgements

We are grateful to Amanda Beeson, Keith Conrad, Abhinav Kumar, Victor Miller, Chris Peikert, Bjorn Poonen, Nigel Smart, and Madhu Sudan for helpful conversations, comments, and references. N.H. was supported by an internship at Microsoft Research New England and an NSF Graduate Research Fellowship.

References

[1] M. Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing* (Dallas, Texas, United States, May 24–26, 1998), pages 10–19. ACM, New York, NY, 1998.

[2] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing* (Hersonissos, Greece, July 6–8, 2001), pages 601–610. ACM, New York, NY, 2001.

[3] D. J. Bernstein. List decoding for binary Goppa codes. Preprint, 2008, http://cr.yp.to/ codes/ goppalist-20081107.pdf.

[4] D. Bleichenbacher and P. Q. Nguyen. Noisy polynomial interpolation and noisy Chinese remaindering. In *Advances in Cryptology – EUROCRYPT 2000*, pages 53–69. Lecture Notes in Computer Science 1807. Springer-Verlag, Berlin, Heidelberg, 2000.

[5] J. Blömer and A. May. New partial key exposure attacks on RSA. In *Advances in Cryptology – CRYPTO 2003*, pages 27–43. Lecture Notes in Computer Science 2729. Springer-Verlag, Berlin, Heidelberg, 2003.

[6] D. Boneh. Finding smooth integers in short intervals using CRT decoding. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing* (Portland, Oregon, United States, May 21–23, 2000), pages 265–272. ACM, New York, NY, 2000.

[7] D. Boneh, G. Durfee, and Y. Frankel. An attack on RSA given a small fraction of the private key bits. In *Advances in Cryptology–ASIACRYPT'98*, pages 25–34. Lecture Notes in Computer Science 1514. Springer-Verlag, Berlin, Heidelberg, 1998.

[8] H. Cohn and N. Heninger. Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding. Preprint, 2010, http://arxiv. org/ abs/ 1008.1284.

[9] D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. Cryptology* 10: 233–260, 1997.

[10] D. Coppersmith. Finding small solutions to small degree polynomials. In *Cryptography and Lattices*, pages 20–31. Lecture Notes in Computer Science 2146. Springer-Verlag, Berlin, Heidelberg, 2001.

[11] D. Coppersmith, N. Howgrave-Graham, and S. V. Nagaraj. Divisors in residue classes, constructively. *Math. Comp.* 77: 531–545, 2008.

[12] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symb. Comput.* 9: 251–280, 1990.

[13] C. Fieker and D. Stehlé. Short bases of lattices over number fields. In *Algorithmic Number Theory*, pages 157–173. Lecture Notes in Computer Science 6197. Springer-Verlag, Berlin, Heidelberg, 2010.

[14] J. von zur Gathen. Hensel and Newton methods in valuation rings. *Math. Comp.* 42: 637–661, 1984.

[15] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Second edition. Cambridge University Press, Cambridge, England, 2003.

[16] J. von zur Gathen and E. Kaltofen. Factorization of multivariate polynomials over finite fields. *Math. Comp.* 45: 251–261, 1985.

[17] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the complexity of polynomial matrix computations. In *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation* (Philadelphia, Pennsylvania, United States, August 3–6, 2003), pages 135–142. ACM, New York, NY, 2003.

[18] V. Guruswami, A. Sahai, and M. Sudan. "Soft-decision" decoding of Chinese remainder codes. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science* (Redondo Beach, California, United States, November 12–14, 2000), pages 159–168. IEEE Computer Society, Los Alamitos, CA, 2000.

[19] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Trans. Inform. Theory* 45: 1757–1767, 1999.

[20] N. Howgrave-Graham. Approximate integer common divisors. In *Cryptography and Lattices*, pages 51–66. Lecture Notes in Computer Science 2146. Springer-Verlag, Berlin, Heidelberg, 2001.

[21] M.-D. Huang and D. Ierardi. Efficient algorithms for the Riemann-Roch problem and for addition in the Jacobian of a curve. *J. Symb. Comput.* 18: 519–539, 1994.

[22] T. Kailath. *Linear Systems*. Prentice-Hall, Inc., Upper Saddle River, NJ, 1980.

[23] S. V. Konyagin and T. Steger. On polynomial congruences. *Math. Notes* 55: 596–600, 1994.

[24] A. K. Lenstra. Factoring multivariate polynomials over algebraic number fields. *SIAM J. Comput.* 16: 591–598, 1987.

[25] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coeficients. *Math. Ann.* 261: 515–534, 1982.

[26] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *Advances in Cryptology – EUROCRYPT 2010*, pages 1–23. Lecture Notes in Computer Science 6110. Springer-Verlag, Berlin, Heidelberg, 2010.

[27] K. Manders and L. Adleman. NP-complete decision problems for quadratic polynomials. In *Proceedings of the Eighth Annual ACM Symposium on Theory of Computing* (Hershey, Pennsylvania, United States, May 3–5, 1976), pages 23–29. ACM, New York, NY, 1976.

[28] R. C. Mason. *Diophantine Equations over Functions Fields*. London Mathematical Society Lecture Note Series 96. Cambridge University Press, Cambridge, England, 1984.

[29] A. May. New RSA vulnerabilities using lattice reduction methods. Ph.D. thesis, University of Paderborn, 2003.

[30] A. May. Using LLL-reduction for solving RSA and factorization problems. In P. Q. Nguyen and B. Vallée, editors, *The LLL Algorithm*, pages 315–348. Springer-Verlag, Berlin, Heidelberg, 2010.

[31] M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing* (Atlanta, Georgia, United States, May 1–4, 1999), pages 245–254. ACM, New York, NY, 1999.

[32] C. Peikert and A. Rosen. Lattices that admit logarithmic worst-case to average-case connection factors. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing* (San Diego, California, United States, June 11–13, 2007), pages 478–487. ACM, New York, NY, 2007.

[33] R. M. Roth and G. Ruckenstein. Efficient decoding of Reed-Solomon codes beyond half the minimum distance. *IEEE Trans. Inform. Theory* 46: 246–257, 2000.

[34] M. A. Shokrollahi and H. Wasserman. List decoding of algebraic-geometric codes. *IEEE Trans. Inform. Theory* 45: 432–437, 1999.

[35] V. Shoup. OAEP reconsidered. In *Advances in Cryptology–CRYPTO 2001*, pages 239–259. Lecture Notes in Computer Science 2139. Springer-Verlag, Berlin, Heidelberg, 2001.

[36] M. Sudan. Ideal error-correcting codes: Unifying algebraic and number-theoretic algorithms. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pages 36–45. Lecture Notes in Computer Science 2227. Springer-Verlag, Berlin, Heidelberg, 2001.

[37] P. Trifonov. Efficient interpolation in the Guruswami-Sudan algorithm. Preprint, 2008, http:// arxiv. org/ abs/0812.4937v3.