Electronic Colloquium on Computational Complexity, Report No. 132 (2005)

Algebraic-geometric generalizations of the Parvaresh-Vardy codes

VENKATESAN GURUSWAMI*

Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195

Abstract

This paper is concerned with a new family of error-correcting codes based on algebraic curves over finite fields, and list decoding algorithms for them. The basic goal in the subject of list decoding is to construct error-correcting codes C over some alphabet Σ which have good rate R, and at the same every Hamming ball of (relative) radius p has few codewords of C, and moreover these codewords can be found in polynomial time.

The trade-off between the rate R and the error-correction radius p is a central one governing list decoding. Traditional "unique decoding" algorithms can achieve $p = (1 - R)/2$, and this was improved in [7] to $p = 1 - \sqrt{R}$ through a new list decoding algorithm for Reed-Solomon (RS) codes. For several years, this remained the best known trade-off between rate and list decoding radius. In a recent breakthrough, Parvaresh and Vardy [11] define a variant of RS codes which can be list decoded beyond the $1 - \sqrt{R}$ radius for rates $R \leqslant 1/16$.

We generalize the PV framework to algebraic-geometric (AG) codes, of which RS codes are an important special case. This shows that their framework applies to fairly general settings, and also better elucidates the key algebraic concepts underlying the new codes. Moreover, since AG codes of arbitrary block length exist over *fixed* alphabets Σ, we are able to almost match the trade-off between p and R obtained in [11] over alphabets of *constant* size. In contrast, the PV codes have alphabet size that is polynomially large in the block length.

Similar to algorithms for AG codes from [7, 8], our encoding/decoding algorithms run in polynomial time assuming a natural polynomial-size representation of the code. For codes based on a specific "optimal" algebraic curve, we also present an expected polynomial time algorithm to construct the requisite representation. This in turn also presents an efficient construction of the representation needed by the list decoding algorithms for AG codes in [7].

*Research supported in part by NSF grant CCF-0343672 and a Sloan Research Fellowship.

ISSN 1433-8092

1 Introduction

1.1 Context and Motivation

In a recent breakthrough in coding theory, Parvaresh and Vardy [11] present a new family of codes that is an elegant variant of Reed-Solomon (RS) codes. For rates $R \leqslant 1/16$, these codes can be efficiently list decoded beyond the fraction $1 - \sqrt{R}$ of errors that was the earlier best trade-off between rate and list decoding radius (achieved for RS codes in [7]). For list decoding up to radius $1-\varepsilon$ for small $\varepsilon \to 0$, their scheme can be used to get codes with rate $\Omega(\varepsilon/\log(1/\varepsilon))$, which is close to the best possible rate of ε.

The new variant of RS codes put forth in [11] is as follows. In a RS code, the message is a polynomial, which is encoded by its evaluations at elements of a field. In the PV-scheme, the message is viewed as a polynomial f, and then a related polynomial h is computed (as a carefully chosen function of f – the details of how this is done are crucial to the success of this approach), and then the encoding comprises of the evaluations of both f and h on the field elements. This gives a non-linear code of half the rate compared to the original Reed-Solomon code. To get better trade-offs between rate and list decoding radius, one can use not a pair but an M-tuple of correlated polynomials for the encoding.

In light of the improvement over RS codes this scheme offers, it is natural to consider whether a similar construction can used to improve upon the list decodability of algebraic-geometric (AG) codes, of which RS codes are but one special case. Such a construction will highlight the generality and promise of this new approach, and in doing so perhaps elucidate its salient features in a general setting unencumbered by specifics of a particular code. Furthermore, algebraic-geometric codes achieve excellent parameters over alphabets of constant size, whereas RS codes require field size at least as large as the alphabet size, and the dependence on alphabet size only worsens in the variant of RS codes considered in [11] (to achieve the above-mentioned codes of rate $\Omega(\varepsilon/\log(1/\varepsilon))$ for decoding radius $1-\varepsilon$, the alphabet size needed is $n^{\Omega(\log(1/\varepsilon))}$).

In this work, using AG codes, we will beat the $1 - \sqrt{R}$ list decoding radius for low rates R over alphabets of *fixed* size. For decoding up to radius $1 - \varepsilon$ for small $\varepsilon \to 0$, we will get codes with alphabet size $(1/\varepsilon)^{O(a)}$ and rate $\Omega(a^{-2}\varepsilon^{1+1/a})$, for every integer $a \geqslant 1$. There is an easy lower bound of $1/\varepsilon$ on the alphabet size and the rate has to be at most $O(\varepsilon)$. Therefore, our result does well simultaneously on both the alphabet size vs. list decoding radius and the rate vs. list decoding radius trade-offs. Our codes also have a nice *list recovering* property which can be used in concatenation schemes with suitable *constant-sized* inner codes to get uniformly constructive binary codes of rate close to ε^3 list-decodable up to radius $(1/2 - \varepsilon)$.

Complexity of encoding/decoding: Since AG codes are a whole family of codes as opposed to a specific code, when we say we give polynomial time encoding and decoding algorithms for them, we mean that every AG-code has a natural (polynomial size) representation given which there are encoding/decoding procedures that run in polynomial time. This is similar to the situation for the original list decoding algorithm for AG codes [7, 8], and is the best one can hope for when we want to decode *every* AG code of a certain type.

However, it makes sense to try to construct this requisite representation efficiently for certain specific AG-codes, ideally the ones which offer the best trade-offs for list decoding. We explicitly address this question in Section 5. For the specific "optimal" AG codes based on a tower of function fields due to Garcia and Stichtenoth [1, 2], we give an expected polynomial time (i.e., *Las*

Vegas) construction of the description of the code. Though not explicit in the sense of deterministic polynomial time constructibility, the representation is guaranteed to be correct and constructing it (a one time job) takes polynomial time with overwhelming probability. This level of explicitness should thus suffice for using the code. We remark that even for the algorithm of Guruswami and Sudan [7, 8] (that achieved a decoding radius of at most $1 - \sqrt{R}$), it was not known how to compute the required representation efficiently. Our construction thus fills an important gap in the literature on efficient decoding of AG codes.

1.2 Generalizing to AG-codes: Ideas and Complications

In this work, we propose a generalization of the Parvaresh-Vardy coding scheme to algebraic-geometric codes. While fairly natural in hindsight (which a "correct" generalization ought to be!), the generalization to algebraic-geometric codes is not immediate, since, as we describe below, the special structure of RS-codes and the rational function field $\mathbb{F}_q(X)$ are used in a more than superficial way in [11].

The ability to view a low-degree polynomial (i.e., the function being evaluated) also as a field element from some field $\mathbb{F}$, and operating on it in the field $\mathbb{F}$ to get another related polynomial is crucial to the PV construction. Indeed, the decoding is performed by solving a system of polynomial equations over the field $\mathbb{F}$ whose solutions contain all possible codewords that must be output. For Reed-Solomon codes, there is a natural way to view polynomials as field elements, since polynomials of degree $< k$ are in one-to-one correspondence with elements of the extension field $\mathbb{F}_q[X]/(E(X)) \approx \mathbb{F}_{q^k}$ (where $E(X)$ is an irreducible polynomial of degree k over $\mathbb{F}_q$). In order to generalize this framework to AG codes, we need an injective homomorphism from the elements of the function field K that are evaluated to give the AG-encoding (i.e., the analog of low-degree polynomials for the RS case) to a suitable field $\mathbb{F}$. We achieve this by associating with an element f of the function field, the field element $\mathbb{F}_{q^\alpha}$ which is the evaluation $f(R)$ of f at a fixed place R of (large enough) degree α. This evaluation is then used to obtain, from the message function f, a correlated function h such that $h(R)$ is a carefully chosen function of $f(R)$. Unlike the RS case, however, for function fields of larger genus this evaluation map restricted to the message functions is only injective and not bijective. Fortunately, we are able to show (Lemma 3) that a correlated function h with the desired evaluation $h(R)$ always exists in a slightly larger space compared to the message space to which f belongs.

The decoding algorithm follows the interpolation followed by root-finding idea that is common to [15, 7, 11]. However, another technical complication arises in the phase when the interpolated polynomial, say Q, with coefficients from the function field, is mapped into a polynomial N with coefficients from $\mathbb{F}_{q^\alpha}$ by evaluating each of its coefficients at the place R. Following [11], we seek to find roots in $\mathbb{F}_{q^\alpha}$ of N, and using the above-mentioned injection from messages into $\mathbb{F}_{q^\alpha}$, map these roots back to obtain the list of messages. It is crucial that in this step N is a nonzero polynomial when Q is. For the Reed-Solomon case, this is easy to achieve, since the coefficients of Q, which are polynomials over $\mathbb{F}_q$ in one variable, come from a *principal ideal domain* (PID), i.e., a ring all of whose ideals are generated by a single element. Therefore, the only way N can be zero when Q is nonzero, is if all coefficients of Q are divisible by the generator of the ideal R (i.e., by a univariate polynomial E of degree α). In this case we can divide Q by the appropriate power of E to get a lower-degree nonzero polynomial $\tilde{Q}$, and then work with it instead (by construction, E doesn't divide $\tilde{Q}$, so it will lead to a nonzero polynomial N upon reduction modulo R).

However, for general function fields, the ring of functions that have poles only at one point

need not be a PID.[1] Thus, the above idea of dividing out Q by the factor from R that divides all coefficients cannot be applied. We do not, therefore, know how to argue in general that N will be nonzero when Q is. We circumvent this problem by restricting the coefficients of Q to come from a much smaller space of functions than is usually done in the interpolation based algorithms of [15, 7, 11]. Specifically, we restrict the pole order of each of the functions to be less than α. This ensures that no nonzero coefficient of Q evaluates to 0 at the place R, which has degree α. Therefore, $Q \neq 0$ implies $N \neq 0$, as desired. This restriction on the coefficients of Q does not come for free, however, and we need to give up a bit on the potential performance in terms of number of errors corrected. Fortunately, this loss is offset by using extra correlated functions at the encoding stage. In effect, the flexibility of using as many correlated functions for the evaluations during encoding trades-off favorably with the loss in performance due to the above complication. This is the reason why our approach begins to give improvements over the decoding of regular AG-codes only when we use 3 or more correlated functions (as opposed to the case of RS codes in [11], where a pair of functions already gives a substantial improvement).

Another fall out of our stringent restriction on the coefficients of Q is that the idea of using *large* "multiplicities" in the interpolation phase actually *degrades* the error correction performance of the algorithm (it does gives minor improvements for small multiplicities, the best one being for multiplicity 3 for the case of three correlated functions). This is in contrast to [7] where the big improvement came by using multiplicities, and also to the Parvaresh-Vardy construction [11] where again large multiplicities are useful. In our presentation, since a small multiplicity gives only a minor improvement, we simply present the interpolation algorithm for the case of multiplicity 1 (i.e., simple zeroes). On the flip side, this greatly helps us in Section 5 since the construction of the requisite representation of the AG code is simpler when one does not have to deal with multiplicities.

1.3 Subsequent work

Subsequent to our work, Patthak [12] also shows how to extend the PV framework to certain AG codes. The best rate we can achieve for list decoding up to radius $(1 - \varepsilon)$ is $\Omega(\varepsilon/\log^2(1/\varepsilon))$ (Corollary 14). Patthak [12] improves this to $\Omega(\varepsilon/\log(1/\varepsilon))$, which matches the bound obtained by Parvaresh and Vardy [11]. However, obtaining a polynomial time construction and decoding algorithm seems harder with the approach put forth in [12]. The main technical differences between this work and [12] lie in how the two complications alluded to above are handled. We now briefly describe these in turn.

Let us consider the first issue of evaluation at R possibly not being a bijection into $\mathbb{F}_{q^\alpha}$. We dealt with this by allowing a slightly larger space from which we pick the correlated functions. This lets us apply our methods to *arbitrary* AG codes, including the explicitly constructed ones based on the Garcia-Stichtenoth function fields [2, 13]. Patthak instead suggests using a specially chosen message space for which the evaluation map is indeed a bijection. Such a space of functions always exists for cases of interest, cf. [14, Prop. I.6.10], but it is not clear how easy explicitly finding and using one is, and in any case one loses the ability to use without any change the preexisting well-studied AG codes like those in [2, 13].

[1] Actually, for our purposes it suffices to stipulate that in this ring, all maximal ideals are principal. However, such a ring, which is an instance of a holomorphy ring [14, Sec. III.2], is always a *Dedekind domain*, and therefore is a PID iff all its maximal ideals are principal.

The second issue was to ensure that the interpolation polynomial Q does not vanish when its coefficients are evaluated at R. It is dealing with this issue that costs us a little in terms of performance compared to [11] — the rate we achieve with m correlated functions is about a factor m smaller, which accounts for the extra factor $\log(1/\varepsilon)$ loss in rate mentioned above. The specific problem is that the coefficients of Q, which come from a space of functions say A, may all vanish at R, and yet we may not be able to remove a common factor from the coefficients of Q to get a new polynomial $\tilde{Q}$ over A that does not have this problem. To deal with this, Patthak suggests dividing all cofficients of Q by a suitable function, say b, such that after division some coefficient of the new polynomial $\tilde{Q}$, say a/b, has a nonzero value at R. However, one is no longer guaranteed that the cofficient $a/b \in A$. So, while such a function b always exists, there remain two issues in implementing this solution in polynomial time (even with access to preprocessed information about the code): (i) how to find such a function b and succinctly represent the necessary information concerning it, and (ii) how to evaluate a/b at R (for $a \in A$ where $a(R) = 0$), since in general a/b lies outside the space A of functions.

Capacity-achieving codes: Recently, explicit codes that approach the capacity of list decoding, i.e., the optimal rate vs. error-correction radius trade-off, were presented in [5]. Specifically, for any desired $R, \varepsilon > 0$, they present codes with rate R that can be list decoded up to a fraction $1 - R - \varepsilon$ of errors. (Though this essentially achieves the optimal error-correction radius for any given rate, the result does not subsume those in this paper or [11], since the list size needed for decoding in [5] is a large polynomial in the block length, whereas we only need constant list size. Also, the decoding complexity and alphabet size we achieve is a lot smaller.)

The underlying codes in [5] are certain *folded Reed-Solomon codes*, which are exactly RS codes but viewed as a code over a larger alphabet by appropriate bundling together of codeword symbols. The ideas in this paper may provide the starting point to generalize such folding schemes to better AG codes. While such a task appears rather intricate, it will be worth the effort as getting close to capacity over a fixed alphabet, say $\mathrm{GF}(2^{12})$, will enable concatenation with a well-understood binary inner code, say the $[24, 12, 8]$ binary Golay code, to get very good binary codes, cf. [9].

2 Our code construction

We now describe a code construction where we use a **triple** of functions in the evaluation. As mentioned above, our scheme does not get an improvement in decoding performance (compared to regular AG-codes) when just two correlated functions are used for the evaluation. The extension of the code, decoding algorithm, and analysis for the case when more than three correlated functions are evaluated as part of encoding, follows in a natural way. The extensions are discussed briefly in Section 4.

2.1 Preliminaries

Most of the notation and terminology we use is standard in the study of algebraic-geometric codes, and can be found in Stichtenoth's book [14]. We briefly recap some key facts concerning algebraic function fields and algebraic-geometric codes that we need for our description. Let K be a function field over $\mathbb{F}_q$, denoted $K/\mathbb{F}_q$, i.e., a finite algebraic extension of the field $\mathbb{F}_q(X)$ of rational functions over $\mathbb{F}_q$. A subring X of K is said to be a valuation ring if for every $z \in K$, either $z \in X$ or

$z^{-1} \in X$. Each valuation ring is a *local ring*, i.e., it has a unique maximal ideal. The set of places of K, denoted $\mathbb{P}_K$, is the set of maximal ideals of all the valuation rings of K. Geometrically, this corresponds to the set of all (non-singular) points on the algebraic curve corresponding to K. The valuation ring corresponding to a place P is called the ring of regular functions at P and is denoted $\mathcal{O}_P$. Associated with a place P is a valuation $v_P : K \to \mathbb{Z}$, that measures the number of zeroes or poles of a function at P (with the convention $v_P(0) = \infty$. In terms of v_P, we have $\mathcal{O}_P = \{x \in K \mid v_P(x) \geqslant 0\}$ and $P = \{x \in K \mid v_P(x) > 0\}$. The quotient $\mathcal{O}_P/P$ is a field since P is a maximal ideal – it is called the residue field at P. The residue field $\mathcal{O}_P/P$ is a finite extension field of degree of $\mathbb{F}_q$; the degree of this extension is called the *degree* of P, and is denoted $\deg(P)$. For every place P, we have an evaluation map $\mathsf{ev}_P : \mathcal{O}_P \to \mathcal{O}_P/P$ defined by $\mathsf{ev}_P(z) = z(P) = z + P$; this map is $\mathbb{F}_q$-linear. We will think of ev_P as a map into $\mathbb{F}_{q^{\deg(P)}}$ using an isomorphism of the residue field to $\mathbb{F}_{q^{\deg(P)}}$. Thus, elements of K can be viewed as functions on $\mathbb{P}_K$ (hence the name *function field* for K); the evaluation of $z \in K$ and $P \in \mathbb{P}_K$, denoted $z(P)$, is either ∞ (if $z \notin \mathcal{O}_P$), or belongs to $\mathbb{F}_{q^{\deg(P)}}$.

The set of divisors D_K of a function field $K/\mathbb{F}_q$ is the (additively written) free abelian group generated by the places $\mathbb{P}_K$. For a divisor $D = \sum_{P \in \mathbb{P}_K} n_P P$ where all but finitely many n_P are 0, its degree, denoted $\deg(D)$, is defined as $\deg(D) = \sum_{P \in \mathbb{P}_K} n_P \deg(P)$ (note that this is a finite sum). For a divisor $D = \sum_P n_P P$, we define the set of functions $\mathcal{L}(D) \stackrel{\text{def}}{=} \{x \in K \mid v_P(x) \geqslant -n_P \ \forall P \in \mathbb{P}_K\}$; this forms a vector space over $\mathbb{F}_q$.

Theorem 1 (Follows from Riemann-Roch)**.** *If $D \in D_K$ is a divisor of $K/\mathbb{F}_q$ of degree at least $2g-1$, then $\dim(\mathcal{L}(D)) = \deg(D) - g + 1$.*

An algebraic-geometric code over $\mathbb{F}_q$ is obtained by evaluating a carefully chosen subset of elements of K at *places of degree one*. For a place P_∞ of degree one and an integer α, the set $\mathcal{L}((\alpha-1)P_\infty)$ consists of all those $z \in K$ for which z has no poles at places other than P_∞, and has less than α poles at P_∞. Typically, an AG-code is defined to be the evaluations of functions in $\mathcal{L}((\alpha-1)P_\infty)$ at n distinct places $P_1, P_2, \ldots, P_n$ (different from P_∞) of degree one. That is,

$$C_{\alpha, P_\infty} = \{\langle f(P_1), f(P_2), \ldots, f(P_n)\rangle \mid f \in \mathcal{L}((\alpha-1)P_\infty)\} .$$

This is a linear code since $\mathcal{L}((\alpha-1)P_\infty)$ is a vector space over $\mathbb{F}_q$. The dimension of C_{α,P_∞} is at least $\alpha - g$ by the Riemann-Roch theorem. Its minimum distance is at least $n - \alpha + 1$ since a nonzero function in $\mathcal{L}((\alpha-1)P_\infty)$ can have at most $(\alpha-1)$ zeroes.

2.2 The code and encoding

We now describe our construction of the code. Let K be a function field over $\mathbb{F}_q$ corresponding to a smooth, irreducible curve. Let g be the genus of K. Suppose K has at least $n+1$ places of degree one, say $P_1, P_2, \ldots, P_n$ and P_∞. Let $k \geqslant g$ be arbitrary (this assumption is mainly for convenience). We will describe a code C of block length n over alphabet $\mathbb{F}_{q^3}$ with q^k codewords. The rate of the code will thus be $r(C) = k/(3n)$. The code will **not** be linear. Let $\{1, \beta_1, \beta_2\}$ be a basis of $\mathbb{F}_{q^3}$ over $\mathbb{F}_q$.

The messages of C will be identified with the vector space $\mathbb{F}_q^k$. We specify the code by specifying its encoding function, Enc, which will be an injective map $\mathsf{Enc} : \mathbb{F}_q^k \to \left(\mathbb{F}_{q^3}\right)^n$.

Let $\alpha = k + g$. Since $\alpha - 1 \geqslant 2g - 1$, by Theorem 1, $\mathcal{L}((\alpha-1)P_\infty)$ is a k-dimensional vector space over $\mathbb{F}_q$ and it is with this space that we identify our messages. Let $\phi_1, \phi_2, \ldots, \phi_k$ be a

basis of $\mathcal{L}((\alpha-1)P_\infty)$. Specifically, a message $(a_1, a_2, \ldots, a_k) \in \mathbb{F}_q^k$ will be viewed as the element $a_1\phi_1 + \cdots + a_k\phi_k \in \mathcal{L}((\alpha-1)P_\infty)$. Therefore, we will describe our encoding function as a map

$$\mathsf{Enc} : \mathcal{L}((\alpha-1)P_\infty) \to (\mathbb{F}_{q^3})^n \,. \tag{1}$$

Let $R \in \mathbb{P}_K$ be a place of degree α. [2] We begin with the following simple lemma, which lets us view our messages as a subset of $\mathbb{F}_{q^\alpha}$, using their evaluations at R. Note that $\mathcal{L}((\alpha-1)P_\infty) \subseteq \mathcal{O}_R$ since functions in $\mathcal{L}((\alpha-1)P_\infty)$ have no poles outside P_∞ and thus certainly do not have a pole at R.

Lemma 2. *The restriction of the map* ev_R *to* $\mathcal{L}((\alpha-1)P_\infty)$ *is injective. Its range is a* k*-dimensional subspace of* $\mathbb{F}_{q^\alpha}$*.*

Proof: Indeed, if $f_1, f_2 \in \mathcal{L}((\alpha-1)P_\infty)$ satisfy $f_1(R) = f_2(R)$, then $f_1 - f_2$ has a zero at R. Hence the zero divisor of $f_1 - f_2$ has degree at least $\deg(R) = \alpha$. However, the pole divisor of $f_1 - f_2$ has degree at most $\alpha - 1$ since $f_1 - f_2 \in \mathcal{L}((\alpha-1)P_\infty)$. Therefore we must have $f_1 - f_2 = 0$. Since ev_R is $\mathbb{F}_q$-linear, and $\mathcal{L}((\alpha-1)P_\infty)$ is a k-dimensional vector space over $\mathbb{F}_q$, the image $\mathsf{ev}_R(\mathcal{L}((\alpha-1)P_\infty))$ is a k-dimensional subspace. ∎

Our plan is to use the above as follows. We can view the message $f \in \mathcal{L}((\alpha-1)P_\infty))$ as the field element $f(R)$. We can attempt to define a correlated message h whose evaluation $h(R)$ is an appropriate function Γ (over $\mathbb{F}_{q^\alpha}$) applied to $f(R)$.[3] However, for the decoding procedure, it seems important that this function Γ be non-linear (over $\mathbb{F}_q$). The the image of ev_R restricted to $\mathcal{L}((\alpha-1)P_\infty)$ is a subspace of $\mathbb{F}_{q^r}$. When Γ is not linear, in general there may not exist $h \in \mathcal{L}((\alpha-1)P_\infty)$ satisfying $h(R) = \Gamma(f(R))$.[4] The following crucial lemma shows that such a h exists provided we allow slightly bigger pole order at P_∞.

Lemma 3. *The image of* $\mathcal{L}((\alpha+2g-1)P_\infty)$ *under* ev_R *equals* $\mathbb{F}_{q^\alpha}$*.*

Proof: Let D be the divisor $(\alpha+2g-1)P_\infty$. We wish to show that the restriction of ev_R to $\mathcal{L}(D)$, denote it by $H : \mathcal{L}(D) \to \mathbb{F}_{q^\alpha}$, is surjective. Note that H is a F_q-linear map, so the the image of H, $\mathsf{Im}(H)$, is a subspace of $\mathbb{F}_{q^\alpha}$. We will show that $\mathsf{Im}(H)$ has dimension α, and this will show that H is surjective.

The image $\mathsf{Im}(H)$ is isomorphic to the quotient $\mathcal{L}(D)/\ker(H)$ where $\ker(H) = \{z \in \mathcal{L}(D) \mid H(z) = 0\}$ is the kernel of H. Recalling that $H(z) = 0$ iff $\mathsf{ev}_R(z) = 0$, we have $\ker(H) = \{z \in \mathcal{L}(D) \mid z \text{ has a zero at } R\}$. Therefore, we have $\ker(H) = \mathcal{L}(D-R)$. Therefore,

$$\dim(\mathsf{Im}(H)) = \dim(\mathcal{L}(D)/\mathcal{L}(D-R)) = \dim(\mathcal{L}(D)) - \dim(\mathcal{L}(D-R)) \,.$$

Now, using Theorem 1, $\dim(\mathcal{L}(D)) = (\alpha+2g-1) - g + 1 = \alpha + g$, and $\dim(\mathcal{L}(D-R)) = \deg(D-R) - g + 1 = ((\alpha+2g-1)-\alpha) - g + 1 = g$. It follows that $\dim(\mathsf{Im}(H)) = \alpha$, as desired. ∎

[2] We note that a place of degree d exists for all d such that $(q^d - 1) > 2q^{d/2}g$, and $d = \alpha \geqslant 2g$ satisfies this condition.

[3] More generally, following the PV-scheme, we can let $(f(R), h(R))$ belong to some curve, but this will improve parameters slightly at best.

[4] For the Reed-Solomon case, $g = 0$, and hence $\alpha = k$ and so the image $\mathsf{ev}_R(\mathcal{L}((\alpha-1)P_\infty)) = \mathbb{F}_{q^\alpha}$, and so such an $h \in \mathcal{L}((\alpha-1)P_\infty)$ satisfying $h(R) = \Gamma(f(R))$ will always exist.

Before we finally describe the encoding function (1), we need one other notation. For each $\gamma \in \mathbb{F}_{q^\alpha}$, we fix an arbitrary preimage in $\mathcal{L}((\alpha+2g-1)P_\infty)$, denote it $I[\gamma]$, that satisfies $ev_R(I[\gamma]) = \gamma$. (Such a preimage exists by Lemma 3.) The code will be parameterized by integers $s_1, s_2 \geqslant 1$ (which will be specified later when we analyze the decoding algorithm). For $f \in \mathcal{L}((\alpha-1)P_\infty)$, we define the i'th coordinate of $\mathsf{Enc}(f)$, for $i = 1, 2, \ldots, n$, by

$$\mathsf{Enc}(f) = f(P_i) \ + \ \beta_1 \cdot I[f(R)^{s_1}](P_i) \ + \ \beta_2 \cdot I[f(R)^{s_2}](P_i) \tag{2}$$

(recall that $\{1, \beta_1, \beta_2\}$ is a basis of $\mathbb{F}_{q^3}$ over $\mathbb{F}_q$). In other words, the encoding consists of the evaluation $f(P_i)$ and also the evaluations $h_1(P_i)$ and $h_2(P_i)$ where h_i is a specific function that satisfies $h_i(R) = f(R)^{s_i}$ for $i = 1, 2$ (the raising to s_i'th power happens in the field $\mathbb{F}_{q^\alpha}$).

Note that the rate of C is $k/(3n)$ and its distance d is at least $n - \alpha + 1 = n - k - g + 1$. Its alphabet size is q^3.

2.3 Encoding complexity

The above encoding can be performed in polynomial time, provided (i) we can efficiently compute functions in $\mathcal{L}((\alpha + 2g - 1)P_\infty)$ at the places $P_1, P_2, \ldots, P_n$ and R, and (ii) we can compute the preimage $I[\gamma] \in \mathcal{L}((\alpha+2g-1)P_\infty)$ of arbitrary $\gamma \in \mathbb{F}_{q^\alpha}$ efficiently. Since the space $\mathcal{L}((\alpha+2g-1)P_\infty)$ is a $\alpha + g$-dimensional $\mathbb{F}_q$-vector space, both of these tasks can be solved in polynomial time using elementary linear algebra, assuming we have a basis for $\mathcal{L}((\alpha + 2g - 1)P_\infty)$ together with the evaluations of the basis functions at P_i as well as at R. This is the representation which we assume for our code. This representation will also suffice to implement our list decoding algorithm in polynomial time, as we will see in Section 3.4.

3 Interpolation based decoding procedure

We now turn to list decoding the above code construction. We recollect the notation of relevant parameters in the construction:

- block length n;
- places $P_1, \ldots, P_n, P_\infty$ of degree 1;
- message length k (over $\mathbb{F}_q$); $\alpha = k + g$; messages correspond to functions in $\mathcal{L}((\alpha - 1)P_\infty)$
- a place R of degree α.
- the powering exponents s_1, s_2 (these will be specified later).

The list decoding problem for radius $n-t$ amounts to solving the following function reconstruction problem:

Input: Triples $(y_i, z_{1i}, z_{2i}) \in \mathbb{F}_q^3$, for $i = 1, 2, \ldots, n$

Output: All functions $f \in \mathcal{L}((\alpha-1)P_\infty)$ for which the triple of functions $(f,\ h_1 = I[f(R)^{s_1}],\ h_2 = I[f(R)^{s_2}])$ satisfies $f(P_i) = y_i$, $h_1(P_i) = z_{1i}$ and $h_2(P_i) = z_{2i}$ for at least t values of $i \in \{1, 2, \ldots, n\}$.

3.1 High level idea behind the algorithm

Let A denote the ring $\cup_{\ell \geqslant 0} \mathcal{L}(\ell P_\infty)$ of all functions in K that have no poles other than possibly at P_∞. The basic idea, following the interpolation based decoding procedure of [15, 7, 11], is to find a nonzero polynomial Q in the polynomial ring $A[Y, Z_1, Z_2]$ such that all triples (f, h_1, h_2) that meet the above output condition are roots of Q. This is done in the same way as in [7] (except even simpler, since we only insist on simple zeroes and not zeroes of higher multiplicities), by finding a nonzero solution to an appropriate homogeneous linear system over $\mathbb{F}_q$. However, the polynomial Q will have exponentially many roots in general, so finding all of them and looking for valid triples (f, h_1, h_2) among them is not an option. Instead, we reduce the polynomial Q modulo the place R, by evaluating each of its coefficients at R, to obtain a polynomial $N \in \mathbb{F}_{q^\alpha}[Y, Z_1, Z_2]$. At this step, as mentioned earlier, we have to be careful that N remains a nonzero polynomial, and for this we depart a bit in the structure we impose on Q.

If (f, h_1, h_2) is a root of Q, clearly the evaluations $(f(R), h_1(R), h_2(R))$ is a root of N. This together with the fact that $h_i(R) = f(R)^{s_i}$ for $i = 1, 2$ implies that $f(R)$ is a root of the univariate polynomial $N[Y, Y^{s_1}, Y^{s_2}]$, call it $T[Y]$. By Lemma 2, the message $f \in \mathcal{L}((\alpha - 1)P_\infty)$ is uniquely recoverable from its evaluation $f(R)$, and so all the solution messages f (and hence the triples (f, h_1, h_2)) can be found by checking amongst the roots of the polynomial T. One additional point to be careful about is that T does not become the zero polynomial (even though $N[Y, Z_1, Z_2]$ is nonzero). This is ensured by a suitable, large enough choice of s_1, s_2.

3.2 Properties required of the interpolated polynomial

We now define the properties we would like from the interpolation polynomial $Q \in A[Y, Z_1, Z_2]$. These properties will guide how the homogeneous linear system will be set up to find Q. Let ℓ be a positive integer to be specified later.

1. Q is nonzero.
2. For all $f, h_1, h_2 \in \mathcal{L}((\alpha + 2g - 1)P_\infty)$, $Q(f, h_1, h_2) \in \mathcal{L}((\ell - 1)P_\infty)$.[5]
3. For every $i = 1, 2, \ldots, n$, for all (f, h_1, h_2) which satisfy $f(P_i) = y_i$, $h_1(P_i) = z_{1i}$ and $h_2(P_i) = z_{2i}$, $Q(f, h_1, h_2)$ has a zero at P_i.

The following simple lemma shows the utility of such a polynomial Q.

Lemma 4. *Let Q satisfy the above conditions. Let $f, h_1, h_2 \in \mathcal{L}((\alpha + 2g - 1)P_\infty)$ satisfy $f(P_i) = y_i$, $h_1(P_i) = z_{1i}$ and $h_2(P_i) = z_{2i}$ for t values of i. If $t \geqslant \ell$, then $Q(f, h_1, h_2) = 0$.*

Proof: The function $Q(f, h_1, h_2)$ has at most $(\ell - 1)$ poles, and it has a zero at P_i for each i for which $f(P_i) = y_i$, $h_1(P_i) = z_{1i}$ and $h_2(P_i) = z_{2i}$, and thus at least t zeroes. If $t \geqslant \ell$, this implies that $Q(f, h_1, h_2)$ is the zero function. ∎

[5] It will actually suffice for us to require that $Q(f, h_1, h_2) \in \mathcal{L}((\ell - 1)P_\infty)$ whenever $f \in \mathcal{L}((\alpha - 1)P_\infty)$ and $h_1, h_2 \in \mathcal{L}((\alpha + 2g - 1)P_\infty)$. But for sake of uniformity and simplicity, we ensure this also for f from the larger space $\mathcal{L}((\alpha + 2g - 1)P_\infty)$.

3.3 The Algorithm

We now specify the decoding algorithm for C formally. Recall the input to the decoding algorithm is a string consisting of triples $(y_i, z_{1i}, z_{2i}) \in \mathbb{F}_q^3$, and the algorithm should find all codewords with agreement t or more with the input string (the parameter t will come out from the analysis). In what follows, $\phi_1, \phi_2, \ldots, \phi_k$ denotes a basis of $\mathcal{L}((\alpha-1)P_\infty)$ (recall that this is a k-dimensional vector space over $\mathbb{F}_q$).

Step 0: Compute integer parameters p, ℓ where

$$\ell \stackrel{\text{def}}{=} p(\alpha + 2g - 1) + \alpha \text{ (so that } \frac{\ell - \alpha}{\alpha + 2g - 1} = p) \,, \tag{3}$$

and p satisfies $k\binom{p+3}{3} > n$, say

$$p \stackrel{\text{def}}{=} \left\lfloor \left(\frac{6n}{k}\right)^{1/3} \right\rfloor . \tag{4}$$

Step 1: Find a **nonzero** trivariate polynomial $Q[Y, Z_1, Z_2]$ with coefficients in $\mathcal{L}((\alpha-1)P_\infty)$ of **total degree** p, i.e., of the form

$$Q[Y, Z_1, Z_2] = \sum_{\substack{j, j_1, j_2 \\ j + j_1 + j_2 \leqslant p}} \Big(\sum_{r=1}^{k} q_{r,j,j_1,j_2} \phi_r \Big) Y^j Z_1^{j_1} Z_2^{j_2} \,,$$

by finding the value of the unknowns $q_{r,j,j_1,j_2} \in \mathbb{F}_q$, such that for each $i = 1, 2, \ldots, n$, the following holds

- The constant term of the polynomial $Q^{(i)}[Y, Z_1, Z_2] \stackrel{\text{def}}{=} Q[Y + y_i, Z_1 + z_{i1}, Z_2 + z_{i2}]$ vanishes at P_i.

Note that the above condition is a homogeneous linear equation over $\mathbb{F}_q$ in the unknowns q_{r,j,j_1,j_2}; hence finding such a polynomial Q amounts to finding a nonzero solution to a homogeneous linear system over $\mathbb{F}_q$.

Step 2: Compute the polynomial $N \in \mathbb{F}_{q^\alpha}[Y, Z_1, Z_2]$ by evaluating each of the coefficients of Q (which are functions in $\mathcal{L}((\alpha-1)P_\infty)$) at the place R.

Step 3: Compute the univariate polynomial $T \in \mathbb{F}_{q^\alpha}[Y]$ where $T[Y] \stackrel{\text{def}}{=} N[Y, Y^{s_1}, Y^{s_2}]$.

Step 4: Compute all the roots in $\mathbb{F}_{q^\alpha}$ of T. (There will be at most $\text{degree}(T) \leqslant \max\{s_1, s_2\}p$ such roots, when T is a nonzero polynomial.)
For each root $\gamma \in \mathbb{F}_{q^\alpha}$ of T, do the following:

- Compute the unique $f \in \mathcal{L}((\alpha-1)P_\infty)$, if any, such that $f(R) = \gamma$ (this can also be accomplished by solving a linear system, with unknowns being the coefficients $a_1, \ldots, a_k$ of the basis elements $\phi_1, \ldots, \phi_k$ where $f = a_1\phi_1 + \cdots + a_k\phi_k$).
- If such a f exists, test if the encoding of f, $\mathsf{Enc}(f)$ that is defined in (2), agrees with the input triples on at least t locations. If so, output f.

3.4 Runtime analysis of the algorithm

The above algorithm can be implemented to run in polynomial time, given an appropriate representation of the code, that consists of:

(i) The evaluation of the basis elements $\phi_1, \ldots, \phi_k$ of $\mathcal{L}((\alpha-1)P_\infty)$ at the places $P_1, P_2, \ldots, P_n$, as well as at a place R of degree α.

(ii) The evaluation of $\psi_1, \psi_2, \ldots, \psi_{2g}$ at the place R, where $\phi_1, \ldots, \phi_k, \psi_1, \ldots, \psi_{2g}$ forms a basis of $\mathcal{L}((\alpha+2g-1)P_\infty)$.

We stress that this is the *same information* that is needed to perform the encoding in polynomial time (see the discussion in Section 2.3). So we do not require any additional precomputation compared to the natural representation used for the encoding.

We now describe why the above information suffices for efficient decoding. With the information in (i), one can perform

- Step 1 using the values of ϕ_i's at $P_1, \ldots, P_n$ by solving a homogeneous linear system over $\mathbb{F}_q$;
- Step 2 using the values of the ϕ_i's at R; and
- The computation of f (if any) satisfying $f(R) = \gamma$ in Step 4, again using the values of the ϕ_i's at R.

Using the information in (ii) one can compute the map $I : \mathbb{F}_{q^\alpha} \to \mathcal{L}((\alpha+2g-1)P_\infty)$ and thus compute $\mathsf{Enc}(f)$ in Step 4 and check which of the f's that are found must be output.

The root-finding in Step 4 can be performed in deterministic $\mathrm{poly}(n, q, \alpha)$ time. Therefore, the overall runtime will be polynomial in the block length n.

We want to point out that the typical representation of an AG-code is a generator matrix, which amounts to the evaluation of the basis functions $\phi_1, \ldots, \phi_k$ at the places $P_1, \ldots, P_n$. The only extra information we require for the algorithm is the evaluation of these basis functions (and up to $2g$ extra ones) at a place R of larger degree. For the tower of function field towers in [2], for which it is known how to compute the generator matrix efficiently [13], we give an expected polynomial time algorithm in Section 5 to compute the evaluations at a high degree place efficiently.

3.5 Analysis of Error-correction Performance

Theorem 5. *For the choice $s_1 = p+1$ and $s_2 = p^2+p+1$, the decoding algorithm of Section 3.3 correctly finds all codewords $c = \langle c_1, \ldots, c_n \rangle$ of C which satisfy $c_i = y_i + \beta_1 z_{i1} + \beta_2 z_{i2}$ for at least t values of $i \in \{1, 2, \ldots, n\}$, for*

$$t = k + g + \left(6\Big(1 + \frac{3g-1}{k}\Big)\right)^{1/3} \cdot \Big((k+3g-1)^2 n\Big)^{1/3}. \tag{5}$$

The number of codewords the algorithm outputs in the worst-case is at most $p(p^2+p+1) \leqslant 3p^3 \leqslant 18n/k$. In particular, for $k \geqslant g$, this implies the algorithm will output all codewords with agreement at least $2k + 4(6k^2 n)^{1/3}$ with a received word.

We will prove the above by a sequence of lemmas.

Lemma 6. *For parameters p, ℓ defined in Step 0 of the algorithm, Step 1 of the algorithm finds a nonzero polynomial $Q[Y, Z_1, Z_2]$ of total degree p that satisfies the interpolation conditions of Section 3.2.*

Proof: Step 1 of the algorithm finds a polynomial $Q[Y, Z_1, Z_2]$ of total degree p whose coefficients lie in $\mathcal{L}((\alpha-1)P_\infty)$. The coefficient of $Y^j Z_1^{j_1} Z_2^{j_2}$ is expressed using the unknowns q_{r,j,j_1,j_2} for $1 \leqslant r \leqslant k$. The total number of unknowns is thus k times the number of trivariate monomials of total degree at most p, which is $\binom{p+3}{3}$, and thus equals $k\binom{p+3}{3}$. The number homogeneous linear conditions imposed on the unknowns is n, one for each place P_i. Therefore, if $k\binom{p+3}{3} > n$, the number of unknowns exceeds the number of constraints, and so a nonzero Q can be found.

It remains to prove that any Q that is found satisfies the following two conditions:

(a) For all $f, h_1, h_2 \in \mathcal{L}((\alpha+2g-1)P_\infty)$, $Q(f, h_1, h_2) \in \mathcal{L}((\ell-1)P_\infty)$, and

(b) For each $i = 1, 2, \ldots, n$, if f, h_1, h_2 evaluate to y_i, z_{i1}, z_{i2} respectively at P_i, then $Q(f, h_1, h_2)$ vanishes at P_i.

Condition (a) is immediate. Indeed, each monomial of Q has degree p and each coefficient of Q belongs to $\mathcal{L}((\alpha-1)P_\infty)$. Therefore, for $f, h_1, h_2 \in \mathcal{L}((\alpha+2g-1)P_\infty)$, $Q(f, h_1, h_2)$ will have at most $(\alpha-1)+p(\alpha+2g-1) = \ell-1$ poles at P_∞, and no poles elsewhere.

For (b), we note the following.

$$\begin{aligned} \mathsf{ev}_{P_i}\big(Q(f, h_1, h_2)\big) &= \mathsf{ev}_{P_i}\big(Q^{(i)}(f - y_i, h_1 - z_{i1}, h_2 - z_{i2})\big) \\ &= \mathsf{ev}_{P_i}\big(Q^{(i)}(f - f(P_i), h_1 - h_1(P_i), h_2 - h_2(P_i))\big) \\ &= 0 \end{aligned}$$

where the last equality follows since by construction of Q, the constant term of $Q^{(i)}(Y, Z_1, Z_2)$ vanishes at P_i, and the functions $f - f(P_i)$, $h_1 - h_1(P_i)$ and $h_1 - h_2(P_i)$ all clearly vanish at P_i. ∎

Lemma 7. *If $Q \not\equiv 0$, then $N \in \mathbb{F}_{q^\alpha}[Y, Z_1, Z_2]$ obtained in Step 2 is a nonzero polynomial of total degree at most p. Moreover, if $Q(f, h_1, h_2) = 0$ for some functions $f, h_1, h_2 \in \mathcal{O}_R$, then $N(f(R), h_1(R), h_2(R)) = 0$.*

Proof: Q has total degree at most p, and hence so does N. Also, any nonzero coefficient of Q evaluates to a nonzero value at R. This is because each coefficient belongs to $\mathcal{L}((\alpha-1)P_\infty)$ and thus has less than α zeroes, while $\deg(R) = \alpha$. Therefore, if $Q \not\equiv 0$, then N is a nonzero polynomial. Since the evaluation map $\mathsf{ev}_R : \mathcal{O}_R \to \mathbb{F}_{q^\alpha}$ is a homomorphism, $N(f(R), h_1(R), h_2(R))$ equals the evaluation of $Q(f, h_1, h_2)$ at R, and so must equal 0 if $Q(f, h_1, h_2) = 0$. ∎

Lemma 8. *If $p \geqslant 1$, $s_1 > p$, $s_2 > s_1 p$, and $N[Y, Z_1, Z_2]$ is a nonzero polynomial of total degree at most p, then the polynomial $T[Y] = N[Y, Y^{s_1}, Y^{s_2}]$ is a nonzero polynomial of degree at most $s_2 p$.*

Proof: The claim about the degree of T is obvious, so we just need to show that T is nonzero. Define the polynomial $S[Y, Z_2] \stackrel{\text{def}}{=} N[Y, Y^{s_1}, Z_2]$. Now $S \equiv 0$ iff $Z_1 - Y^{s_1}$ divides $N[Y, Z_1, Z_2]$. But this is impossible since the total degree of N is at most $p < s_1$. Therefore, S is a nonzero polynomial of total degree at most $s_1 p$. Now, the polynomials T, S are related by $T[Y] = S[Y, Y^{s_2}]$. Therefore, $T \equiv 0$ iff $Z_2 - Y^{s_2}$ divides $S[Y, Z_2]$. Again this is impossible since the degree of S is at most $s_1 p < s_2$. We conclude that T must be a nonzero polynomial. ∎

Combining the above lemmas, it is easy to conclude:

Lemma 9. *For ℓ, p defined as in Step 0, and the choices $s_1 = p + 1$ and $s_2 = p(p + 1) + 1$, for every $f \in \mathcal{L}((\alpha - 1)P_\infty)$, the following holds: If $\mathsf{Enc}(f)$ agrees with the input word on ℓ or more places, then $f(R)$ is a root of T, and thus f will be found and output in Step 4 of the algorithm. Moreover, the algorithm will output at most $p^3 + p^2 + p$ such functions f.*

Proof: By Lemma 6 and Lemma 4, each $f \in \mathcal{L}((\alpha - 1)P_\infty)$ for which $\mathsf{Enc}(f)$ has agreement $\geqslant \ell$ with the input word, satisfies $Q(f, h_1, h_2) = 0$, where $h_1 = I[f(R)^{s_1}]$ and $h_2 = I[f(R)^{s_2}]$. By Lemma 7, $N(f(R), h_1(R), h_2(R)) = 0$. Hence

$$T(f(R)) = N(f(R), f(R)^{s_1}, f(R)^{s_2}) = N(f(R), h_1(R), h_2(R)) = 0 \, .$$

Thus $f(R)$ is a root of T. By Lemma 8, T is a nonzero polynomial of degree at most $s_2 p$. It follows that the number of solutions f output by the algorithm is at most $s_2 p = p^3 + p^2 + p$. ∎

Proof of Theorem 5: Theorem 5 follows immediately from Lemma 9 and the choice of ℓ in (3): $\ell = \alpha + (\alpha + 2g - 1)p$ where $p = (6n/k)^{1/3}$ and $\alpha = k + g$. ∎

For small rates, the result of Theorem 5 improves over the list decoding algorithm for algebraic-geometric codes in [7] which corrects up to $n - \sqrt{(k + g - 1)n}$ errors.

So far our construction applied to any function field. We conclude this section by stating the following corollary to Theorem 5 that records the best trade-off between list decoding radius and rate our scheme offers. This is achieved by using functions fields which have the largest possible ratio of g/n, i.e., the genus to the number of places of degree 1. By the Drinfeld-Vlădut bound, this ratio is at most $1/(\sqrt{q} - 1)$ as q is held fixed and n tends to infinity. For q which is a square, there are sequence of functions fields known with increasing genus known for which g/n approaches $1/(\sqrt{q} - 1)$ [17, 1]. Plugging in these function fields in the result of Theorem 5, and recalling that the rate of our code construction equals $k/(3n)$, we get the following final result for this section. (The claim about the polynomial time constructibility of the codes follows from Section 5.)

Theorem 10. *For q a square prime power and every r, $\frac{1}{\sqrt{q}-1} < 3r < 1 - \frac{1}{\sqrt{q}-1}$, there is a family of codes over alphabet size q^3 of rate r, relative distance at least $1 - 3r - \frac{1}{\sqrt{q}-1}$, and which is list decodable up to a fraction*

$$1 - 3r - \frac{1}{\sqrt{q} - 1} - 6\Big(r + \frac{1}{\sqrt{q} - 1}\Big)^{2/3}$$

of errors using lists of size at most $6/r$.[6] Furthermore, there is a natural representation of the codes, computable in expected polynomial time, for which the encoding as well as list decoding up to this radius can be performed in polynomial time.

For decoding up to a fraction of errors approaching 1, we get the following corollary. (We say a code of block length n is (ρ, L)-list decodable for every received word there are at most L codewords within distance ρn from it.)

Corollary 11. *For all small enough $\varepsilon > 0$, there is a family of Q-ary codes for $Q = O(1/\varepsilon^9)$ which has rate $\Omega(\varepsilon^{3/2})$ and which is $(1 - \varepsilon, O(1/\varepsilon^{3/2}))$-list decodable. Furthermore, the codes have a representation, computable in expected polynomial time, that permits polynomial time encoding and list decoding up to radius $(1 - \varepsilon)$.*

[6] When the stated fraction of errors is non-positive, the stated bound of course becomes trivial. So the result is meaningful only for small rates r.

The above corollary can be contrasted with regular AG codes that are list decodable up to radius $(1-\varepsilon)$ using the algorithm in [7]. Those codes had worse rate of $\Theta(\varepsilon^2)$, but their alphabet size was $O(1/\varepsilon^4)$.

4 Extension to codes using higher order correlations and list recovering

Evidently, we can extend our construction to use more than three correlated functions for the encoding. For low rates this leads to much better, and in fact near-optimal, rate vs. list decoding radius trade-offs. In this section, we state the formal results that follow from such a generalization. We also state a generalization of our results to *list recovering*, a setting where for each position we have not just one but several possible values, and the goal is to find codewords which agree with at least one of the suggested values on t or more locations, for some agreement parameter t. Since these generalizations are quite easy to work out, we skip the proofs, and simply state the results with brief explanations of the key changes.

4.1 Using more than three correlated functions

We can modify the construction of Section 2.2 by using $m \geqslant 4$ correlated functions $f, h_1, h_2, \ldots, h_{m-1}$ to perform the encoding. The function $f \in \mathcal{L}((\alpha-1)P_\infty)$ will be the message, and the functions $h_i \in \mathcal{L}((\alpha+2g-1)P_\infty)$ will be defined by $h_i = I[f(R)^{s_i}]$ for suitable choices of $s_1, s_2, \ldots, s_{m-1}$. The rate of the code is $k/(mn)$ and its distance at least $n-k-g+1$.

For the decoding, in order to find a nonzero interpolation polynomial Q, the parameter p in (4) must now satisfy

$$k\binom{p+m}{m} > n$$

since the number of monomials in a total degree p m-variate polynomial equals $\binom{p+m}{m}$. The above condition is satisfied for the choice $p = \lfloor (m!n/k)^{1/m} \rfloor$. The choice of ℓ remains the same as in (3). For the choice $s_1 = p+1$ and $s_i = ps_{i-1}+1$ for $i \leqslant 2 \leqslant m-1$, a decoding algorithm similar to the one in Section 3.3 finds all codewords with agreement at least t with any input word, where

$$t = k + g + \left(m!\Big(1+\frac{3g-1}{k}\Big)\right)^{1/m} \cdot \Big((k+3g-1)^{m-1}n\Big)^{1/m}. \tag{6}$$

The size of list output will be at most $s_{m-1}p = \sum_{i=1}^{m} p^i \leqslant mp^m$. Using the above with the function fields of best possible g/n ratio, we get the following generalization of Theorem 10, which we view as our main result.

Theorem 12 (Main). *For q a square prime power, an integer $m \geqslant 3$, and every r satisfying $\frac{1}{\sqrt{q}-1} < mr < 1 - \frac{1}{\sqrt{q}-1}$, the following holds. There is a family of codes over alphabet size q^m of rate r, relative distance at least $1 - mr - \frac{1}{\sqrt{q}-1}$, and which is list decodable up to a fraction*

$$1 - mr - \frac{1}{\sqrt{q}-1} - (4m!)^{1/m}\Big(mr + \frac{3}{\sqrt{q}-1}\Big)^{1-1/m}$$

of errors using lists of size at most $m!/r$. Moreover, there is a natural representation of the codes, computable in expected polynomial time, for which the encoding as well as list decoding up to this radius can be performed in polynomial time.

Likewise, we get the following generalization of Corollary 11.

Corollary 13. *For all $\varepsilon > 0$ and every integer $m \geqslant 3$, there is a family of Q-ary codes for $Q = O((m/\varepsilon)^{2m^2/(m-1)})$ which has rate $\Omega(\frac{1}{m^2} \cdot \varepsilon^{m/(m-1)})$ and which is $(1-\varepsilon, O(m^2 m!(1/\varepsilon)^{m/(m-1)}))$-list decodable. Moreover, the codes have a representation, computable in expected polynomial time, that permits polynomial time list decoding up to radius $(1-\varepsilon)$.*

To maximize the rate as a function of ε (which we think of as a small constant), we can pick $m = \Theta(\log(1/\varepsilon))$ in the above corollary. This leads to the following corollary.

Corollary 14. *For all $\varepsilon > 0$, there is a family of Q-ary codes for $Q = (1/\varepsilon)^{O(\log(1/\varepsilon))}$ which has rate $\Omega(\varepsilon/\log^2(1/\varepsilon))$ and which is $(1-\varepsilon, (1/\varepsilon)^{O(\log\log(1/\varepsilon))})$-list decodable. Moreover, the codes have a natural representation, computable in expected polynomial time, that permits polynomial time encoding as well as polynomial time list decoding up to radius $(1-\varepsilon)$.*

4.2 Extension to list recovering

Definition 15. A code $C \subseteq \Sigma^n$ is said to be (γ, l, L)-*list recoverable* if for every sequence of sets $S_1, S_2, \ldots, S_n$, where each $S_i \subseteq \Sigma$ has at most l elements, the number of codewords $c \in C$ which satisfy $c_i \in S_i$ for at least γn values of $i \in \{1, 2, \ldots, n\}$ is at most L.

Note a code being (ρ, L)-list decodable is the same thing as it being $(1-\rho, 1, L)$-list recoverable, so the above notion is more general than list decoding. The notion of list recovering actually sits somewhere in between hard decoding where for each position we have a unique symbol in the received word and soft decoding where for each position we have appropriate non-negative weights for all symbols in Σ. List recovering was first explicitly studied in work on extractor codes [16]; the name was coined in [3], and it has played a crucial role in combinatorial constructions of list decodable codes, including those with linear complexity algorithms [4].

We now make the following observation. The algorithm in Section 3.3 can be trivially generalized to handle the case when there is a set S_i consisting of possibly more than one triple (y_i, z_{i1}, z_{i2}) for each location i. We simply need to add a constraint for each such triple in the interpolation of Step 2, so that the total number of constraints will now be the total number of triples N (or in other words the total size of all the S_i's). It immediately follows that we get an algorithm for list recovering that works with agreement t as in (5) with N replacing the block length n. Of course, a similar generalization also holds for the m-variate decoding algorithm and the agreement bound of (6).

Plugging this into function fields with $g/n = 1/(\sqrt{q}-1)$, and performing some straightforward computations, we can get the following. We note that Corollary 13 is a special case obtained by setting $l = 1$ and $\gamma = \varepsilon$.

Theorem 16. *For all integers $l \geqslant 2$, for all $\gamma > 0$ and all integers $m \geqslant 3$, there is a family of Q-ary codes for $Q = O((ml^{1/m}/\gamma)^{2m^2/(m-1)})$ which has rate $\Omega(\gamma/m^2 \cdot (\gamma/l)^{1/(m-1)})$ and which is (γ, l, L)-list recoverable for $L = O(m^2 \cdot m! \cdot (l/\gamma)^{m/(m-1)})$. Moreover, the codes have a natural representation, computable in expected polynomial time, that permits polynomial time encoding as well as polynomial time (γ, l, L)-list recovering.*

Using the choice $m = \lceil \log_2(l/\gamma) \rceil$ in the above we get:

Corollary 17. *For all integers $l \geqslant 2$ and all $\gamma > 0$, there is a family of Q-ary codes for $Q = l^{O(\log\log(l/\gamma))} \cdot 2^{O(\log^2(1/\gamma))}$ which has rate $\Omega\Big(\frac{\gamma}{\log^2(l/\gamma)}\Big)$ and which is (γ, l, L)-list recoverable for $L = (l/\gamma)^{O(\log\log(l/\gamma))}$. Moreover, the codes have a natural representation, computable in expected polynomial time, that permits polynomial time encoding as well as polynomial time (γ, l, L)-list recovering.*

4.3 Binary codes for list decoding up to radius $(1/2 - \varepsilon)$

We now consider the problem of constructing binary codes for list decoding up to radius $(1/2-\varepsilon)$, for small $\varepsilon > 0$. Using the list recoverable codes of Corollary 17 with parameters $l = O(1/\varepsilon^2)$ and $\gamma = \varepsilon/2$ as the outer code in a concatenation scheme with a constant-sized binary inner code with Q codewords and rate $\Omega(\varepsilon^2)$ and that is $(1/2 - \varepsilon/2, l)$-list decodable, we can show the following.

Theorem 18. *For every $\varepsilon > 0$, there is a family of binary codes of rate $\Omega(\varepsilon^3/\log^2(1/\varepsilon))$ that is $(1/2 - \varepsilon, (1/\varepsilon)^{O(\log\log(1/\varepsilon))})$-list-decodable. The codes can be constructed in expected polynomial time and admit a polynomial time encoding algorithm as well a polynomial time list decoding algorithm for radius $(1/2 - \varepsilon)$.*

We remark that a similar result, in fact with a slightly better rate of $\Omega(\varepsilon^3/\log(1/\varepsilon))$, can be obtained by using the Parvaresh-Vardy codes as the outer code. But the construction time of such a code will be at least $n^{\Omega(\log(1/\varepsilon))}$ since the inner code we need to find by brute-force will have at least $n^{\Omega(\log(1/\varepsilon))}$ codewords. Similarly, the recent construction of [5] achieve a rate of $\Omega(\varepsilon^3)$ for $(1/2 - \varepsilon, L)$-list-decodable codes, but their construction time as well as list size is $n^{\Omega(1/\varepsilon)}$. In contrast, our codes are uniformly constructive, i.e., can be constructed and decoded in time $f(\varepsilon)n^{O(1)}$ with exponent of n independent of ε.

5 Constructing the representation of the code

We now show how for a specific family of AG codes, we can construct in (expected) polynomial time, the representation outlined in Section 3.4 that is needed for the encoding and decoding algorithms. The particular codes are based on a tower of function fields proposed by Garcia and Stichtenoth [2]. This tower meets the Drinfeld-Vlăduţ bound and thus gives the best possible AG codes (in terms of the rate vs. distance trade-off). We next describe this tower.

Let q_0 be a prime power and $F = \mathbb{F}_{q_0^2}$. The tower of function fields F_i, $i = 0, 1, 2, \ldots$ is defined as a sequence of *Artin-Schreier* extensions. We begin with $F_0 = F(x_0)$, the field of rational functions in x_0. For $i \geqslant 1$, F_i is an algebraic extension of F_{i-1} of degree q_0:

$$F_i = F_{i-1}(x_i) \qquad \text{where } x_i^{q_0} + x_i = \frac{x_{i-1}^{q_0}}{x_{i-1}^{q_0-1} + 1}\,. \tag{7}$$

The above tower meets the Drinfeld-Vlăduţ bound, and thus leads to AG codes with best rate vs distance trade-offs. In [13], a polynomial time algorithm is presented to compute the generator matrix of this code. All we need to add to this to achieve the representation needed in Section 3.4 are the evaluations of the basis elements at some place R of a specified large degree, and the evaluations of $2g$ extra functions at the code places $P_1, P_2, \ldots, P_n$ and at R. This turns out to be not so straightforward. We begin with a description of some of the basic facts about the function fields F_m. The description assumes some basic knowledge of splitting of places in field extensions.

The genus $g(F_m)$ of F_m satisfies $g(F_m) \leqslant q_0^{m+1}$. Let $\Omega = \{\gamma \in F \mid \gamma^{q_0} + \gamma = 0\}$ denote the set of trace zero elements. For $\theta \in F$, let $P_\theta^{(0)}$ denote the unique zero of $x_0 - \theta$ in F_0. Let $P_\infty^{(0)}$ denote the unique pole of x_0 in F_0. The place $P_\infty^{(0)}$ is totally ramified in the tower, i.e., in each F_m there is precisely one place, $P_\infty^{(m)}$, that lies above $P_\infty^{(0)}$ and moreover this place has degree one. We will use AG codes based on F_m by using as message space $\mathcal{L}((\alpha-1)P_\infty^{(m)})$ — functions in F_m that have $< \alpha$ poles at $P_\infty^{(m)}$ and no poles elsewhere.

We now describe the places where the message functions are evaluated for the encoding. Each of the $q_0^2 - q_0$ places $P_\theta^{(0)}$ for $\theta \in F \setminus \Omega$ splits completely in the tower and thus has q_0^m places of degree one lying above it in F_m. Let $n = (q_0^2 - q_0)q_0^m$ and let $P_1, P_2, \ldots, P_n$ be the set of all places of F_m that lie above $P_\theta^{(0)}$ for $\theta \in F \setminus \Omega$. We use the places $P_1, P_2, \ldots, P_n$ as the evaluation places for encoding. Note that $n/g(F_m) \geqslant (q_0 - 1)$ and hence the code meets the Drinfeld-Vlădut bound.

Let R_m be the ring of functions that have a pole only at $P_\infty^{(m)}$. As shown in [13], every function R_m has an expression of the form

$$x_0^l \cdot \left(\sum_{e_0=0}^{(m-1)q_0+1} \sum_{e_1=0}^{q_0-1} \cdots \sum_{e_m=0}^{q_0-1} c_{\mathbf{e}} g_0 \frac{x_0^{e_0} x_1^{e_1} \cdots x_m^{e_m}}{\pi_1 \cdots \pi_{m-1}} \right) \tag{8}$$

where $l \geqslant 0$, $c_{\mathbf{e}} \in F$, and for $0 \leqslant k < m$, $g_k = x_k^{q_0-1} + 1$ and $\pi_k = g_0 g_1 \cdots g_k$. Moreover, for any n', Shum et al [13] present an algorithm running in time polynomial in n', n that outputs a basis of $\mathcal{L}(n' P_\infty^{(m)})$ in the above form, together with evaluations of the basis elements at $P_1, P_2, \ldots, P_n$. We note that this latter evaluation part is easily done once the basis elements are represented in the form (8), since for each P_i, evaluating at P_i amounts to substituting appropriate values from $F \setminus \Omega$ for $x_0, x_1, \ldots, x_m$.[7] Likewise, it suffices to find out the evaluations of $x_0, x_1, \ldots, x_m$ at a place R of degree α in F_m. We now proceed towards this goal, and will soon prove Theorem 20 that will show how this can be done.

The places of degree α in $F_0 = F(x_0)$ are in one-one correspondence with irreducible polynomials of degree α over F. The place corresponding to an irreducible polynomial $p_0(x_0) \in F[x_0]$ is equal to

$$P_{p_0(x_0)} \stackrel{\text{def}}{=} \left\{ \frac{a(x_0)}{b(x_0)} \mid a(x_0), b(x_0) \in F[x_0],\ p_0(x_0) | a(x_0),\ p_0(x_0) \not| b(x_0) \right\} .$$

The following lemma shows that one can find a place of degree α in F_m by finding a place of degree α in F_0 that has a place of degree α lying above it in the extension $[F_m : F_0]$.

Lemma 19. *For every $D \geqslant \max\{m+5, 10\}$, there are at least $\frac{q_0^{2D}}{2D \cdot q_0^m}$ places of degree D in F_0 that have a place of degree D lying above them in F_m.*

Proof: Let $g = g(F_m)$ be the genus of of F_m; we know $g \leqslant q_0^{m+1}$. Let $\mathcal{T}_D$ denote the set of places in F_m of degree D. By the Hasse-Weil bound, it is known that the number of places $B_D = |\mathcal{T}_D|$ of degree D in F_m satisfies $|B_D - q_0^{2D}/D| < (2 + 7g)q_0^D/D$, cf. [14, Corollary V.2.10]. It follows that $B_D \geqslant q_0^{2D}/D - 8gq_0^D/D \geqslant q_0^{2D}/D - 8q_0^{m+1+D}/D \geqslant \frac{q_0^{2D}}{2D}$.

Let $\mathcal{T}'_D \subseteq \mathcal{T}_D$ be those places of degree D in F_m that do **not** lie above a place of degree D in F_0. Let $B'_D = |\mathcal{T}'_D|$. The number N_D of places of degree D in F_0 which have a place of degree D lying

[7] If we begin with $x_0 = \gamma \in F \setminus \Omega$, and solve the equations in (7) in sequence for $x_1, x_2, \ldots, x_m$, then for all solutions, we will have each $x_i \in F \setminus \Omega$, cf. [2, Lemma 3.9].

above them in F_m satisfies $N_D \geqslant (B_D - B'_D)/q_0^m$, since the degree of the extension $[F_m : F_0] = q_0^m$ and so at most q_0^m places of F_m lie above any place of F_0.

Now, if $\tilde{P} \in \mathcal{T}'_D$, then the place $\tilde{P}_0$ lying below it in F_0 must have degree D_0 at most $D/2$ (since D_0 must divide D and is not equal to D). It follows that $B'_D \leqslant q_0^m n_{D/2}$ where $n_{D/2}$ is the number of places of degree at most $D/2$ in F_0. Clearly $n_{D/2} \leqslant \sum_{i=1}^{D/2+1} (q_0^2)^i \leqslant q_0^{D+4}$.

Hence $q_0^m N_D \geqslant B_D - B'_D \geqslant q_0^{2D}/D - 8q_0^{m+1+D}/D - q_0^{D+4}$ and this latter quantity is easily seen to be at least $\frac{q_0^{2D}}{2D}$ when $D \geqslant \max\{m+5, 10\}$. ∎

We are now ready to prove that the evaluations of the basis functions of $\mathcal{L}(n'P_\infty^{(m)})$ at some place of large degree in F_m can be efficiently found. Recall that the block length n of the code is $n = (q^2 - q)q^m$.

Theorem 20. *There is a randomized algorithm that on input integers n', α with $5 \log n \leqslant \alpha \leqslant n$, outputs in expected $\mathrm{poly}(n, n')$ time the evaluations of a set of basis functions of $\mathcal{L}(n'P_\infty^{(m)})$ at some place $R \in \mathbb{P}_{F_m}$ with $\deg(R) = \alpha$.*

Proof: Applying Lemma 19, when $5 \log n \leqslant \alpha \leqslant n$, if we pick a monic polynomial $p_0(x_0)$ over $\mathbb{F}_{q_0^2}$ of degree α, then with probability at least $\frac{1}{2\alpha q_0^m} \geqslant \frac{1}{2n^2}$, the degree α place $P_{p_0(x_0)} \in \mathbb{P}_{F_0}$ will have a place of degree α above it in F_m. Suppose that given an irreducible polynomial $p_0(x_0)$ of degree α we could check in $\mathrm{poly}(n)$ time whether the place $P_{p_0(x_0)}$ has some place $R_{p_0(x_0)}$ of degree α above it in F_m, and if so also output the evaluations of $x_0, x_1, \ldots, x_m$ at the place $R_{p_0(x_0)}$. Then we can simply pick a random monic polynomial of degree α, check it is irreducible (which can be done in deterministic polynomial time), and run the above check, and repeat the process till we succeed in finding a place of degree α in F_m together with the evaluations of $x_0, x_1, \ldots, x_m$ at that place. This process will succeed in expected $O(n^2)$ trials of the initial monic polynomial. Using the form (8) of the basis elements of $\mathcal{L}(n'P_\infty^{(m)})$, we can also compute the evaluations of the basis functions at $R_{p_0(x_0)}$ from the evaluations of $x_0, x_1, \ldots, x_m$ at $R_{p_0(x_0)}$.

Therefore, it remains to check whether a given degree α irreducible $p_0(x_0) \in F[x_0]$ has a place of degree α above it in F_m, and if it does, to find the evaluations of $x_0, \ldots, x_m$ at one of those places. Let $L = F[x_0]/(p_0(x_0))$; L is isomorphic to the finite field $\mathbb{F}_{q_0^{2\alpha}}$. Let $\zeta_0 \in L$ be the residue of $\frac{x_0^{q_0}}{x_0^{q_0-1}+1}$ modulo $p_0(x_0)$. A well-known theorem of Kummer, cf. [14, Theorem III.3.7], when applied to the tower (7), implies that $P_{p_0(x_0)}$ has some place of degree α above it in F_m iff the sequence of equations $x_1^{q_0} + x_1 = \zeta_0$ and $x_{i+1}^{q_0} + x_{i+1} = \frac{x_i^{q_0}}{x_i^{q_0-1}+1}$ for $1 \leqslant i < m$ has a solution $x_i = \zeta_i \in L$ for $1 \leqslant i \leqslant m$. Moreover, in such a case, there is a place $R_{p_0(x_0)}$ of degree α in F_m such that the evaluation of x_i at the place equals ζ_i for $0 \leqslant i \leqslant m$.

Now, for any $\zeta \in L$ represented in the basis $\{1, x_0, \ldots, x_0^{\alpha-1}\}$ over $\mathbb{F}_{q_0^2}$, one can find all solutions in L of a single equation $z^q + z = \zeta$ in $\mathrm{poly}(\alpha)$ time by solving a linear system with 2α unknowns over $\mathbb{F}_{q_0}$. This is because $z^{q_0} + z$ is a *linearized polynomial* and is a $\mathbb{F}_{q_0}$-linear function on L, cf. [10, Chap. 3, Sec. 4]. It follows that one can find all solutions $(\zeta_1, \ldots, \zeta_m) \in L^m$ to $x_1, \ldots, x_m$ that satisfy the above equations in $q_0^m \cdot \mathrm{poly}(\alpha) = \mathrm{poly}(n)$ time, by solving at most q_0^m such linearized polynomial equations. If no such solution exists, the particular choice $p_0(x_0)$ fails. Otherwise, we can use an arbitrary one of those solutions $(\zeta_0, \zeta_1, \ldots, \zeta_m)$ as the evaluations of $x_0, \ldots, x_m$ respectively at a place of degree α. ∎

6 Concluding Remarks

We have generalized the Parvaresh-Vardy approach to all algebraic-geometric codes. These new codes are obtained by evaluating several functions from a function field, which are correlated in a carefully specified way, at some rational points on the algebraic curve. Some complications arise in the higher genus case compared to RS codes (the genus 0 case), but we showed how to handle these with some loss in error-correction performance.

The scheme of evaluating correlated functions/messages to perform the encoding is quite general and can also be applied to Chinese Remainder codes (in fact for these codes there is a precise parallel with Reed-Solomon codes), and perhaps more generally to "ideal-based" codes [6]. The details should be quite straightforward now that we have abstracted the salient features of the algorithm for general AG-codes.

Acknowledgments

Many thanks to Farzad Parvaresh and Alexander Vardy for sending me an early draft of their paper. I thank Henning Stichtenoth for useful discussions on the splitting behavior of places in towers of function fields.

References

[1] Arnaldo Garcia and Henning Stichtenoth. A tower of Artin-Schreier extensions of function fields attaining the Drinfeld-Vlăduţ bound. *Inventiones Mathematicae*, 121:211–222, 1995.

[2] Arnaldo Garcia and Henning Stichtenoth. On the asymptotic behavior of some towers of function fields over finite fields. *Journal of Number Theory*, 61(2):248–273, 1996.

[3] Venkatesan Guruswami and Piotr Indyk. Expander-based constructions of efficiently decodable codes. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 658–667, 2001.

[4] Venkatesan Guruswami and Piotr Indyk. Linear-time encodable and list decodable codes. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 126–135, June 2003.

[5] Venkatesan Guruswami and Atri Rudra. Explicit capacity-achieving list-decodable codes. *Manuscript*, October 2005.

[6] Venkatesan Guruswami, Amit Sahai, and Madhu Sudan. Soft-decision decoding of Chinese Remainder codes. In *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 159–168, 2000.

[7] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45:1757–1767, 1999.

[8] Venkatesan Guruswami and Madhu Sudan. On representations of algebraic-geometric codes. *IEEE Transactions on Information Theory*, 47(4):1610–1613, May 2001.

[9] Ralf Koetter and Alexander Vardy. Soft decoding of Reed Solomon codes and optimal weight assignments. In *ITG Fachtagung*, Berlin, Germany, January 2002.

[10] Rudolf Lidl and Harald Niederreiter. *Introduction to Finite Fields and their applications*. Cambridge University Press, Cambridge, MA, 1986.

[11] Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *Proceedings of the 46th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 285–294, 2005.

[12] Anindya Patthak. Reducing the alphabet size of the Parvaresh-Vardy code using Algebraic Geometry codes. *Manuscript*, September 2005.

[13] Kenneth Shum, Ilia Aleshnikov, P. Vijay Kumar, Henning Stichtenoth, and Vinay Deolalikar. A low-complexity algorithm for the construction of algebraic-geometric codes better than the Gilbert-Varshamov bound. *IEEE Transactions on Information Theory*, 47(6):2225–2241, 2001.

[14] Henning Stichtenoth. *Algebraic Function Fields and Codes*. Universitext, Springer-Verlag, Berlin, 1993.

[15] Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.

[16] Amnon Ta-Shma and David Zuckerman. Extractor codes. *IEEE Transactions on Information Theory*, 50(12):3015–3025, 2004.

[17] Michael A. Tsfasman, Serge G. Vlădut, and Thomas Zink. Modular curves, Shimura curves, and codes better than the Varshamov-Gilbert bound. *Math. Nachrichten*, 109:21–28, 1982.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'