Electronic Colloquium on Computational Complexity, Report No. 36 (2008)

# Soft decoding, dual BCH codes, and better list-decodable $\varepsilon$-biased codes*

Venkatesan Guruswami† Atri Rudra‡

**Abstract**

We construct binary linear codes that are efficiently list-decodable up to a fraction $(1/2-\varepsilon)$ of errors. The codes encode $k$ bits into $n = \mathrm{poly}(k/\varepsilon)$ bits and are constructible and list-decodable in time polynomial in $k$ and $1/\varepsilon$ (in particular, in our results $\varepsilon$ need not be constant and can even be polynomially small in $n$). Our results give the best known polynomial dependence of $n$ on $k$ and $1/\varepsilon$ for such codes. Specifically, we are able to achieve $n \leqslant O(k^3/\varepsilon^{3+\gamma})$ or, if a linear dependence on $k$ is required, $n \leqslant O(k/\varepsilon^{5+\gamma})$, where $\gamma > 0$ is an arbitrary constant. The best previously known constructive bounds in this setting were $n \leqslant O(k^2/\varepsilon^4)$ and $n \leqslant O(k/\varepsilon^6)$. Non-constructively, a random linear encoding of length $n = O(k/\varepsilon^2)$ suffices, but no sub-exponential algorithm is known for list decoding random codes.

Our construction with a cubic dependence on $\varepsilon$ is obtained by concatenating the recent Parvaresh-Vardy (PV) codes with dual BCH codes, and crucially exploits the soft decoding algorithm for PV codes. This result yields better hardness results for the problem of approximating NP witnesses in the model of Kumar and Sivakumar. Our result with the linear dependence on $k$ is based on concatenation of the PV code with an arbitrary inner code of good minimum distance.

In addition to being a basic question in coding theory, codes that are list-decodable from a fraction $(1/2-\varepsilon)$ of errors for $\varepsilon \to 0$ have found many uses in complexity theory. In addition, our codes have the property that all nonzero codewords have relative Hamming weights in the range $(1/2-\varepsilon, 1/2+\varepsilon)$; this *$\varepsilon$-biased* property is a fundamental notion in pseudorandomness.

*Preliminary version to appear in *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity (CCC 2008).*

†Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195. Work done while on leave at the School of Mathematics, Institute for Advanced Study, Princeton, NJ. Research supported in part by a Packard fellowship, and NSF grants CCF-0343672, CCR-0324906. `venkat@cs.washington.edu`

‡Department of Computer Science and Engineering, University at Buffalo, State University of New York, Buffalo, NY, 14620. Research supported in part by startup funds from University at Buffalo. `atri@cse.buffalo.edu`

ISSN 1433-8092

# 1 Introduction

Consider the task of communicating $k$ bits of information over a channel that is going to adversarially corrupt a fraction $1/2 - \varepsilon$ of the transmitted bits. What is the fewest number $n = n(k, \varepsilon)$ bits we need to communicate so that no matter which of the $(1/2 - \varepsilon)n$ bits are corrupted, we can recover the $k$ message bits efficiently, i.e., in time polynomial in $k$ and $1/\varepsilon$? The main goal of this paper is to understand the asymptotic dependence of $n$ on $k$ and $\varepsilon$. This natural problem from coding theory also arises in many contexts in complexity theory including: construction of hardcore predicates from one-way functions [7, 26], approximating the VC dimension [20], constructions of extractors [23], membership comparability of NP-complete sets [25] and approximating NP-witnesses [17, 6].[1]

The mapping from message bits to a larger string of redundant bits is called an error-correcting code (or just a *code*). Each message of $k$ bits is mapped to an $n$-bit *codeword*. The process of recovering the transmitted message from the received corrupted codeword is called *decoding*. If we want the decoder to *uniquely* recover the transmitted codeword, then it is well known that one cannot recover from more than a $1/4$ fraction of errors. Unfortunately, all the applications mentioned above need to work with a fraction $1/2 - \varepsilon > 1/4$ of errors. In fact, in some cases $\varepsilon$ could be polynomially small in $n$. In such high noise regimes, one has to settle with a relaxation of unique decoding called *list decoding*. Under list decoding, the decoder is allowed to output a (small) list of codewords with the guarantee that the transmitted codeword is present in the list. The fact that a list decoder can output a small list of codewords could be problematic since we need to recover the *actual* transmitted codeword. Fortunately, in all the applications mentioned above, the decoder has access to some side information, which allows it to go through the list and pick the transmitted codeword. The only constraint that this introduces is that the size of the list needs to be bounded by a polynomial in $n$. Note that this is also an *a priori* requirement for the list decoding algorithm to run in polynomial time.

Thus, the natural question to ask is: what is the smallest $n$ one can achieve with a code that needs to be list decoded from a $1/2 - \varepsilon$ fraction of errors? It is well known that there exist codes with $n = O(k/\varepsilon^2)$ such that for every error pattern only $O(1/\varepsilon^2)$ many codewords need to be output; in fact a random code with $2^k$ codewords of such a block length has this property with high probability [28, 4]. Further, this bound is tight, that is, for $n = o(k/\varepsilon^2)$, for *any* code, there exists an error pattern for which a super-polynomial number of codewords need to be output. Thus, $n = \Theta(k/\varepsilon^2)$ is information-theoretically best possible. The upper bound of $n = O(k/\varepsilon^2)$ above is non-constructive: a code that is guaranteed to have these properties is not known to be efficiently computable, and even if we are content with a randomized Monte Carlo construction of such a code, there is no polynomial time list decoding algorithm known for such codes. This is a serious drawback since the applications need efficient constructions and decoding algorithms.

In summary, the applications of these codes demand that they be list-decodable from a $1/2 - \varepsilon$ fraction of errors in time polynomial in $k/\varepsilon$. Further, these codes need to be constructed in time polynomial in $k/\varepsilon$. In particular, the focus of this paper is on codes with construction and list decoding time complexity of $\mathrm{poly}(k/\varepsilon)$ with as small a block length $n$ as possible. We should remark that in the complexity-theoretic applications, the exact dependence of $n$ on $k$ often does

[1]There are some other applications such as hardness amplifications of boolean functions [27] and hardness amplification of the permanent [3] which use additional properties of the mapping from $k$ bits to $n$ bits.

not matter as long as the exponent of $k$ is some constant. However, in the "traditional" setting of coding theory and asymptotically good codes, the interesting regime is when the exponent of $k$ equals one. On the other hand, in some complexity theory settings (including the hardness of approximating NP witnesses which we discuss in more detail in Section 1.2), one would like $\varepsilon$ to be as small a function of $n$ as possible can manage. This makes the goal of approaching the optimal $1/\varepsilon^2$ dependence of the block length $n$ on $\varepsilon$ important.

### 1.1 Our bounds and comparison with previously best known bounds

Prior to this work, the following were the best known dependence of $n$ on $k$ and $\varepsilon$ for binary codes with $\text{poly}(k/\varepsilon)$ construction and list decoding time (for correcting a fraction $1/2 - \varepsilon$ of errors):

(a) By concatenating an outer Reed-Solomon code with an inner Hadamard code, a bound of $n = O(k^2/\varepsilon^4)$ was achieved in [12].

(b) By concatenating Reed-Solomon codes with a binary linear code of relative distance $(1/2 - O(\varepsilon^2))$, a bound of $n = O(k/\varepsilon^6)$ was obtained in [13]. This was a polynomial time *Las Vegas* construction (where the list decoding property of the code is *certified*).

Both the above results exploit the powerful soft decoding algorithm for Reed-Solomon codes (aka, weighted polynomial reconstruction) from [11]. In the case when the time for construction and list decoding are allowed to be *exponential* in $1/\varepsilon$, the recent work on binary codes list-decodable up to the Zyablov bound yields a bound of $n = O(k/\varepsilon^3)$ [10]. Earlier, an encoding length of $n = O(k/\varepsilon^4)$ was obtained in [9], but the construction time was once again exponential in $1/\varepsilon$.

In this work, we improve the previous best known results by designing codes that achieve $n$ to be about $O(k^3/\varepsilon^3)$ as well as $O(k/\varepsilon^5)$, each with a construction and decoding complexity of $\text{poly}(k/\varepsilon)$. In particular,

(1) By concatenating the Parvaresh-Vardy codes [22] with dual BCH codes [19], we achieve $n = O\left(\frac{k^3}{\varepsilon^{3+\gamma}}\right)$ for any constant $\gamma > 0$.

(2) By concatenating the Parvaresh-Vardy codes [22] with a binary linear code of relative distance $(1/2 - O(\varepsilon^2))$, a bound of $n = O(k/\varepsilon^{5+\gamma})$ for any constant $\gamma > 0$. As with the $n = O(k/\varepsilon^6)$ result from [13], this is a polynomial time *Las Vegas* construction with a certified guaranteed on the list decoding property (plus, of course, a list decoding algorithm of complexity $\text{poly}(k/\varepsilon)$).

We remark that the Parvaresh-Vardy (PV) codes are a generalization of Reed-Solomon codes and the dual BCH codes are generalizations of Hadamard codes. Thus, for our result (1), we generalized both the outer and inner codes used in result (a) from [12] mentioned earlier. Interestingly, our improvement seems to crucially require generalizing *both* the outer and the inner codes. The Hadamard code is too wasteful and we cannot afford to use it as inner code when the outer code is a PV code, since PV codes have a much larger alphabet size than Reed-Solomon codes. Dual BCH codes, which are generalized Hadamard codes, are much more efficient in encoding length. On the other hand, they necessarily provide weaker guarantees. In particular, the second moment of a certain coset weight distribution is bounded in the case of Hadamard codes (this follows from the Parseval's identity for Fourier coefficients, and was the key to the result in [12]). For duals of

BCH codes with distance $(2t+1)$, an analogous bound holds only for the $2t$'th moment (this was shown in the work [14] on property testing of dual BCH codes). But quite remarkably, there is a *soft decoding* algorithm for PV codes that works under the weaker guarantee of bounded higher order moments, and this enables us to obtain our result (1). We stress that the power of the soft decoding algorithm for PV codes, and the ability to exploit it via meaningful weights passed from the dual BCH decoder, is crucial for our result (1).

Our result (2) achieving $n = O(k/\varepsilon^{5+\gamma})$ is simpler. It is based on list decoding the inner codes up to a radius of $(1/2 - O(\varepsilon))$ (each output list will have size at most $O(1/\varepsilon^2)$ by the Johnson bound). The resulting lists from the inner decoding stage are used by a list recovery algorithm for PV codes to complete the decoding.

While we state our results only for binary codes for sake of convenience, analogous bounds also hold for codes over $\mathbb{F}_q$ for any fixed $q$, for the corresponding problem of decoding a fraction $(1-1/q)(1-\varepsilon)$ of errors.

**$\varepsilon$-biased codes.** Our constructions also yield codes that are $\varepsilon$-biased, namely every nonzero codeword has Hamming weight in the range $[(1/2-\varepsilon)n, (1/2+\varepsilon)n]$. The rows of a generator matrix of an $\varepsilon$-biased code are called an $\varepsilon$-biased set of length $k$ strings. Our constructions yield explicit $\varepsilon$-biased sets of size $O\left(k^3/\varepsilon^{3+\gamma}\right)$ or $O\left(k/\varepsilon^{5+\gamma}\right)$ which are in addition list-decodable up to a fraction $(1/2-\varepsilon)$ of errors. The concept of an $\varepsilon$-biased set, which originated in the seminal work [21], is a fundamental one in pseudorandomness with a rich body of work and numerous applications. The best non-constructive bound on the size of an $\varepsilon$-biased set of length $k$ strings equals $O(k/\varepsilon^2)$. (It is also known that the size needs to be *at least* $\Omega(k/(\varepsilon^2 \log(1/\varepsilon)))$.) The best known explicit bounds for $\varepsilon$-biased sets equal $O(k^2/\varepsilon^2)$ [2] and $O(k/\varepsilon^3)$ [1], but these have weaker decoding guarantees than the $(1/2-\varepsilon)$ radius we achieve.

### 1.2 Approximating NP-witnesses

Every language $L$ in $NP$ comes equipped with a polynomial-time decidable witness predicate $R_L$ such that $L = \{x | \exists y, |w| = |x|^c \wedge R_L(x,y)\}$. Note that $L$ might have multiple witness predicates. For an NP-complete language $L$ and given a string $x \in L$, unless $P = NP$, one cannot find in polynomial-time a witness $y$ such that $R_L(x,y)$ is true. One natural approximation (considered in [6, 17]) is to try and find a string $z$ that does not differ much from a satisfying witness $y$.

Kumar and Sivakumar [17] showed the following connection of approximating NP witness to list decodable codes.

**Theorem 1.** *Let $\{C_i\}_i$ be a family of binary codes such that for every integer $i > 1$, $C_i$ is an $[n_i, i]_2$ code[2] that can be list decoded from a fraction $1/2 - \varepsilon$ of errors (for some $\varepsilon = \varepsilon(n_i) > 0$) in $\mathrm{poly}(i, 1/\varepsilon)$ time. Then for any language $L$ in $NP$, there exists a polynomial-time decidable witness predicate $R'_L$ such that for $x \in L$ the following holds. If one can compute a string $z \in \{0,1\}^{|y|}$ such that $\Delta(z,y) \leqslant |y|/2 - \varepsilon|y|$ in polynomial time, where $R'_L(x,y)$ is true, then one can compute a witness $y'$ that satisfies $R'_L(x,y')$ in polynomial time.*

We now briefly sketch the idea behind the proof of the theorem above. Let $R_L$ be a polynomial-

[2] See Section 2 for definitions related to codes.

time decidable witness predicate for $L$. We define $R'_L$ as follows:

$$R'_L(x,y) = \Big[(|y| = |x|^d) \wedge \big(\exists z \left[R_L(x,z) \wedge |z| = |x|^c \wedge y = C_{|z|}(z)\right]\big)\Big].$$

Consider an $x \in \{0,1\}^n$ such that $x \in L$ and define $N = n^d$. Let $y' \in \{0,1\}^N$ be such that $\Delta(y', C_N(z')) \leqslant N/2 - \varepsilon N$ for some witness $z' \in \{0,1\}^{n^c}$ such that $R_L(x,z')$ is satisfied. By running the list decoding algorithm for $C_N$, one can obtain a polynomial-sized list $\{z'_1, z'_2, \ldots, z'_\ell\}$ that contains $z'$. One can then prune the list to find some satisfying witness for $R_L$ by checking if $R_L(x, z'_i)$ is satisfied for some $1 \leqslant i \leqslant \ell$.

Recall that our result implies that there are codes with $n = O(k^3/\varepsilon^{3+\gamma})$ that are list decodable in $\mathrm{poly}(n)$ time up to $1/2-\varepsilon$ fraction of errors. Note that if we choose $\varepsilon = n^{-1/3+\gamma}$, then $n = \mathrm{poly}(k)$ (assuming $\gamma$ is a constant). Thus, Theorem 1 immediately implies the following corollary:

**Corollary 2.** *For every $\gamma > 0$ the following holds. For every language in $NP$, there exists a polynomial time decidable witness predicate $R'_L$ such that for every $x \in L$, given an arbitrary string of length $N$ that agrees some unknown $N$-bit witness $y$ that satisfies $R'_L(x,y)$ in at least $N/2 + N^{2/3+\gamma}$ positions, one can, in* $\mathrm{poly}(|x|)$ *time, compute a witness $y'$ that satisfies $R'_L(x,y')$.*

Thus, for any NP-complete language, it is NP-hard to compute a string that agrees in at least $N/2 + N^{2/3+\gamma}$ (for $\gamma > 0$) positions with some $N$-bit witness (for some witness predicate). This improves upon the previous "hardness of approximation" result of $N/2+N^{3/4+\gamma}$ from [12]. The best one could hope for via these methods is a bound of $N/2 + O(\sqrt{N \log N})$, as we show in Section 6.

**Related work on NP witness approximation.** The model of approximating NP-witness of [17] has the slight disadvantage that the polynomial-time decidable witness predicate $R'_L$ for the language $L \in$ NP is somewhat unnatural. For example, for the $SAT$ problem, we would like the witness for $R'_L$ (and $R_L$) to be a truth assignment. This model was first considered by Gál, Halevi, Lipton and Petrank [6] (though their original notion of approximation was to correctly compute some bits of a satisfying witness). Feige, Langberg and Nissim in [5] show that it is NP-hard to compute an assignment that agrees with $N/2 + N^\eta$ many bits of a satisfying assignment for a 3CNF formula (for some $\eta > 0$). Sheldon and Young extended this result to work for every $\eta > 0$ [24]. Similar results were shown for other specific NP-complete problems in [5, 24].

We now discuss the result of Sheldon and Young for SAT in some more detail. As was mentioned earlier, one cannot hope to go beyond a "hardness of approximation" result of $N/2 + \sqrt{N}$ using Theorem 1. By contrast, the result for SAT in [24] shows a hardness of approximation result for $N/2 + N^\eta$ for *any* $\eta > 0$ (and hence, "beats" the list decoding approach). Sheldon-Young *do not* use list decodable codes in their reduction. In particular, their reductions works by repeating an input variable enough number of times (and suitably changing the SAT formula) so that assigning the "wrong" value to the chosen bit will result in an assignment that agrees with any satisfying assignment in strictly less than $N/2 + N^\eta$ many positions. Thus, if there exists an algorithm that can output an assignment that agrees in at least $N/2 + N^\eta$ positions with some satisfying assignment, then the assignment computed by the algorithm will assign the "correct" value to the chosen bit. Repeating this procedure recovers a satisfying assignment (assuming one exists).

We note that the reduction outline above is specific to SAT (though the idea can be generalized to some other "natural" NP-complete problems). By contrast, the reduction of Theorem 1 works for *arbitrary* NP language $L$. (Further, the witness predicate $R'_L$ of Theorem 1 can be constructed from *any* witness predicate $R_L$ for $L$.)

### 1.3 Organization of the Paper

We start with the basic definitions from coding theory that will be needed in this paper in Section 2. We review some properties of our "outer" codes in Section 3 and our "inner" code in Section 4. We then present and analyze binary codes that are list decodable from a very close to 50% of errors in Section 5. We present a simple lower bound showing a quadratic dependence of $n$ on $\varepsilon$ in Section 6. Finally, we conclude with some open questions in Section 7.

## 2 Coding Basics

In this section, we will quickly review the basic concepts from coding theory that will be needed for this work. A code of *dimension* $k$ and *block length* $n$ over an alphabet $\Sigma$ is a subset of $\Sigma^n$ of size $|\Sigma|^k$. The *rate* of such a code equals $k/n$. Each vector in $C$ is called a codeword. In this paper, we will focus on the case when $\Sigma$ is a finite field. We will denote by $\mathbb{F}_q$ the field with $q$ elements. A code $C$ over $\mathbb{F}_q$ is called a linear code if $C$ is a subspace of $\mathbb{F}_q^n$. In this case the dimension of the code coincides with the dimension of $C$ as a vector space over $\mathbb{F}_q$. By abuse of notation we will also think of a code $C$ as a map from elements in $\mathbb{F}_q^k$ to their corresponding codeword in $\mathbb{F}_q^n$. If $C$ is linear, then the linear code corresponding to its null space is called the dual of $C$ and is denoted by $C^{\perp}$.

The Hamming distance between two vectors $\mathbf{u}, \mathbf{v} \in \Sigma^n$, denoted by $\Delta(\mathbf{u}, \mathbf{v})$ is the number of places they differ in. The (minimum) distance of a code $C$ is the minimum Hamming distance between any two pairs of distinct codewords from $C$. The relative distance is the ratio of the distance to the block length. The Hamming weight of a vector $\mathbf{u} \in \Sigma^n$, denoted by $wt(\mathbf{u})$, is the number of non-zero positions in $\mathbf{u}$. A linear code over $\mathbb{F}_q$ of block length $n$, dimension $k$, and minimum distance $d$ will be denoted compactly as an $[n, k, d]_q$ code; when the distance $d$ is omitted, such a code will be referred to as an $[n, k]_q$ code.

A list decoding algorithm for a code $C$ of block length $n$ needs to do the following. Given an error parameter $0 \leqslant \rho < 1$ and a received word $\mathbf{y} \in \Sigma^n$ the decoder needs to output all codewords $\mathbf{c} \in C$ such that $\Delta(\mathbf{c}, \mathbf{y}) \leqslant \rho n$. We say that a code $C$ of block length $n$ is $(\rho, L)$-list-decodable, if for every such $\mathbf{y}$, there are at most $L$ codewords which satisfy $\Delta(\mathbf{c}, \mathbf{y}) \leqslant \rho n$.

**Code Concatenation.** Concatenated codes are constructed from two different kinds of codes that are defined over alphabets of different sizes. Say we are interested in a code over a finite field $\mathbb{F}_r$ (in this paper, we will present all our results for $r = 2$). Then the *outer code* $C_{\text{out}}$ is defined over $\mathbb{F}_q$, where $q = r^k$ for some positive integer $k$ and has block length $N$. The second type of code, called the *inner code*, which is denoted by $C_{\text{in}}$ is defined over $\mathbb{F}_r$ and is of dimension $k$ (note that the message space of $C_{\text{in}}$ and the alphabet of $C_{\text{out}}$ have the same size). The concatenated code, denoted by $C = C_{\text{out}} \circ C_{\text{in}}$, is defined as follows. Let the rate of $C_{\text{out}}$ be $R$ and let the block length of $C_{\text{in}}$ be $n$. Define $K = RN$ and $r = k/n$. The input to $C$ is a vector $\mathbf{m} = \langle m_1, \ldots, m_K \rangle \in (\mathbb{F}_{r^k})^K$.

Let $C_{\text{out}}(\mathbf{m}) = \langle x_1, \ldots, x_N \rangle$. The codeword in $C$ corresponding to $\mathbf{m}$ is defined as follows

$$C(\mathbf{m}) = \langle C_{\text{in}}(x_1), C_{\text{in}}(x_2), \ldots, C_{\text{in}}(x_N) \rangle.$$

It can be verified that $C$ has rate, dimension and block length of $rR$, $kK$ and $nN$. Further, the minimum distance of $C$ is at least the product of the distances of $C_{\text{out}}$ and $C_{\text{in}}$.

# 3 Algebraic outer codes and soft decoding

### Reed-Solomon codes and Parvaresh-Vardy construction

The following theorem can be deduced via a straightforward adaptation of the work of Parvaresh and Vardy [22] to the "soft-decoding" setting. The case $s = 1$ of the statement was shown by Guruswami and Sudan [11] for Reed-Solomon codes. The Parvaresh-Vardy (PV) code for $s > 1$ encodes a polynomial $f(X)$ of degree less than $K$ by the evaluations of $f(X)$, and $s - 1$ correlated polynomials $f_j(X) = (f(X))^{h^j} \pmod{E(X)}$ for $1 \leqslant j \leqslant s - 1$, at $N$ distinct elements of $\mathbb{F}_q$. Here $E(X)$ is an arbitrary irreducible polynomial over $\mathbb{F}_q$ of degree $K$, and $h$ is an integer parameter (which has to be large enough for the list decoding guarantee). Note that when $h$ is a power of $p^l$ where $p = \text{char}(\mathbb{F}_q)$ and $l \geqslant 1$ is an arbitrary integer, the above encoding is $\mathbb{F}_{p^l}$-linear (this will be important for us to get *linear* codes in the end, after concatenation with a suitable linear inner code).

**Theorem 3.** *For all integers $s \geqslant 1$, for all prime powers $r$ and all powers $q$ of $r$, every pair of integers $1 < K \leqslant N \leqslant q$, there is an explicit $\mathbb{F}_r$-linear map $E : \mathbb{F}_q^K \to \mathbb{F}_{q^s}^N$ such that:*

1. *The image of $E$, $C \subseteq \mathbb{F}_{q^s}^N$, is a code of minimum distance at least $N - K + 1$.*
2. *Let $W = (s+1)! \sum_{i,\alpha} \binom{w_{i,\alpha}+s}{s+1}$ and $L = \left( r + \left( \frac{W}{K-1} \right)^{1/(s+1)} \right)^s$. There is a list decoding algorithm, that given a collection of nonnegative* integers *$\{w_{i,\alpha}\}$, $1 \leqslant i \leqslant N$, $\alpha \in \mathbb{F}_{q^s}$, runs in time* $\text{poly}(q, L)$, *and outputs a list of size at most $L$ that includes precisely the set of codewords $(c_1, \ldots, c_N) \in C$ that satisfy*

$$\sum_{i=1}^{N} w_{i,c_i} > ((K-1)^s W)^{\frac{1}{s+1}} \quad . \tag{1}$$

We now state a version that holds for *all* nonnegative weights (and eliminates the pseudopolynomial dependence on the weights). It is obtained by a suitable scaling and rounding of the weights to convert them to integer weights and then applying the above statement. A similar proof for RS codes ($s = 1$) appears in [8, Chap. 6]. For the sake of completeness, we include a proof below.

**Theorem 4.** *For all integers $s \geqslant 1$, for all prime powers $r$ and all powers $q$ of $r$, every pair of integers $1 < K \leqslant N \leqslant q$, there is an explicit $\mathbb{F}_r$-linear map $E : \mathbb{F}_q^K \to \mathbb{F}_{q^s}^N$ such that:*

1. *The image of $E$, $C \subseteq \mathbb{F}_{q^s}^N$, is a code of minimum distance at least $N - K + 1$.*

2. *Let $\{w_{i,\alpha}\}$, $1 \leqslant i \leqslant N$, $\alpha \in \mathbb{F}_{q^s}$ be a collection of non-negative weights. Let $w_{max} = \max_{i,\alpha} w_{i,\alpha}$ and $W' = \left(\sum_{i,\alpha} w_{i,\alpha}^{s+1}\right)/w_{\max}^{s+1}$. Then for every $\zeta > 0$ there is an algorithm, that given $\{w_{i,\alpha}\}$, $1 \leqslant i \leqslant N$, $\alpha \in \mathbb{F}_{q^s}$, runs in time bounded by $(rs/\zeta)^{O(s)}(W'q)^{O(1)}$, and outputs a list of size at most $W'/\zeta^{s+1}(rs)^{O(s)}$ that includes precisely the set of codewords $c \in C$ satisfying*

$$\sum_{i=1}^{N} w_{i,c_i} > (s+1)\left((K-1)^s \sum_{i,\alpha} w_{i,\alpha}^{s+1}\right)^{\frac{1}{s+1}} + \zeta N w_{max}\ .^{3} \tag{2}$$

***Proof*.** The linear map $E$ is the one from Theorem 3. If all the weights equal 0, then the condition (2) cannot be satisfied for any codeword, so assume that not all the $w_{i,\alpha}$'s are zero. Let $B$ be a large enough integer (the choice $B = \lceil \frac{1}{\zeta} \rceil$ will do). Set $w'_{i,\alpha} = \left\lfloor \frac{Bw_{i,\alpha}}{w_{\max}} \right\rfloor$. Note that $\frac{Bw_{i,\alpha}}{w_{max}} - 1 \leqslant w'_{i,\alpha} \leqslant \frac{Bw_{i,\alpha}}{w_{max}}$. Each $w'_{i,\alpha}$ is a nonnegative integer that is at most $B$. Clearly the condition (1) will be satisfied by a certain codeword $\mathbf{c} = (c_1, \ldots, c_N)$ with respect to the integer weights $w'_{i,\alpha}$ if the following condition is met:

$$\sum_{i=1}^{N} \left(\frac{Bw_{i,c_i}}{w_{\max}} - 1\right) > \sqrt[s+1]{(K-1)^s (s+1)! \sum_{i,\alpha} \binom{w'_{i,\alpha} + s}{s+1}}.$$

Now if $w'_{i,\alpha} = 0$ then $\binom{w'_{i,\alpha}+s}{s+1} = 0$. Since $w'_{i,\alpha}$ is an integer $(s+1)!\binom{w'_{i,\alpha}+s}{s+1} \leqslant \left(w'_{i,\alpha}(1+s)\right)^{s+1}$. Thus, the condition above is met if

$$\sum_{i=1}^{N} \left(\frac{Bw_{i,c_i}}{w_{\max}} - 1\right) > (s+1) \sqrt[s+1]{(K-1)^s \sum_{i,\alpha} \left(\frac{Bw_{i,\alpha}}{w_{max}}\right)^{s+1}},$$

which in turn is equivalent to the condition

$$\sum_{i=1}^{N} w_{i,c_i} > (s+1) \sqrt[s+1]{(K-1)^s \sum_{i,\alpha} w_{i,\alpha}^{s+1}} + \frac{N w_{max}}{B}.$$

By the choice of $B$, this is implied by (2).

It remains to justify the claims about the runtime and output list size. As as we have seen before, $W = (s+1)! \sum_{i,\alpha} \binom{w'_{i,\alpha}+s}{s+1}$ is at most $(s+1)^{s+1} \sum_{i,\alpha} (w'_{i,\alpha})^{s+1}$. As $w'_{i,\alpha} \leqslant Bw_{i,\alpha}/w_{max}$,

$$W \leqslant \left(\frac{B(s+1)}{w_{max}}\right)^{s+1} \sum_{i,\alpha} w_{i,\alpha}^{s+1} \leqslant \left(\frac{2s}{\zeta}\right)^{s+1} W'$$

by the choice of $B$ and definition of $W'$. The claimed bounds on runtime and list size now follow from those of Theorem 3. ■

[3]We can make the multiplicative factor of $(s+1)$ a constant arbitrarily close to 1 with suitable choice of parameters. For the low-rate regime we are interested in, this does not make much difference so we state here a crude bound that suffices.

### A list recovery bound for PV codes

The following is a corollary of Theorem 3 using only 0, 1 weights $w_{i,\alpha}$, namely $w_{i,\alpha} = 1$ if $\alpha \in S_i$ (for some subsets $S_i \subseteq \mathbb{F}_{q^s}$ for $1 \leqslant i \leqslant N$) and 0 otherwise. This variant is called list recovery in the literature.

**Corollary 5.** *For all integers $s \geqslant 1$, for all prime powers $r$ and all powers $q$ of $r$, every pair of integers $1 < K \leqslant N \leqslant q$, there is an explicit $\mathbb{F}_r$-linear map $E : \mathbb{F}_q^K \to \mathbb{F}_{q^s}^N$ such that:*

1. *The image of $E$, $C \subseteq \mathbb{F}_{q^s}^N$, is a code of minimum distance at least $N - K + 1$.*
2. *There is an algorithm, that given a collection of subsets $S_i \subseteq \mathbb{F}_{q^s}$ for $i = 1, 2, \ldots, N$, each of size at most $\ell$, runs in $\mathrm{poly}((rs)^s, q, \ell)$ time, and outputs a list of size at most $O((rs)^s N\ell/K)$ that includes precisely the set of codewords $(c_1, \ldots, c_N) \in C$ that satisfy $c_i \in S_i$ for at least $\alpha N$ values of $i$, provided*
$$\alpha > (s+1)(K/N)^{s/(s+1)}\ell^{1/(s+1)} . \tag{3}$$

### Johnson bound

The following well known result called the Johnson bound states that a code of good minimum distance also has good list-decodability. We state it only for binary codes. Proofs can be found, for example, in [12] and [8, Chap. 3].

**Lemma 6.** *Let $C$ be a binary code of block length $n$ and relative distance $\delta < 1/2$. Then, for every $L \geqslant 1$, $C$ is*
$$\left(\frac{1}{2}\Big(1 - \sqrt{1 - 2\delta + \frac{2\delta}{L}}\Big), L\right) \textit{-list-decodable} .$$

In particular, the above implies that a code of relative distance $(1/2 - \varepsilon^2)$ is $(1/2 - \varepsilon, \frac{1}{\varepsilon^2})$-list-decodable, a fact we will make use of later on (in Theorem 14). Note that this is a combinatorial guarantee– such a code need not be efficiently list decodable.

## 4 Inner dual BCH codes

### 4.1 Weight distribution and duals of linear codes

In this subsection, we will recollect some known results on linear codes. Given a binary linear code $C$ of block length $n$, and $\mathbf{y} \in \mathbb{F}_2^n$, let $C_{\mathbf{y}}$ denote the code $C \cup (C + \mathbf{y})$, where $\mathbf{y} + C \stackrel{def}{=} \{\mathbf{c} + \mathbf{y} | \mathbf{c} \in C\}$ is a *coset* of $C$. For any $0 \leqslant w \leqslant n$, we will use $A_w(C)$ to denote the number of codewords of Hamming weight $w$.

If $f(X)$ is a univariate polynomial over the reals, then we have
$$\mathop{\mathbf{E}}_{\mathbf{c} \in C}[f(wt(\mathbf{c}))] = \frac{1}{|C|} \cdot \sum_{i=0}^{n} A_i(C) f(i). \tag{4}$$

Our goal will be to compute the expectation of a certain low-degree polynomial over a coset of a linear code. The following lemma is handy for this task.

**Lemma 7** ([15])**.** *Let $C$ be a binary linear code of block length $n$ and let $f(X)$ be a polynomial over the reals. The following holds for every $\mathbf{y} \in \mathbb{F}_2^n$:*

$$\mathop{\mathbf{E}}_{\mathbf{c}\in C}\left[f(wt(\mathbf{y}+\mathbf{c})\right] = 2\mathop{\mathbf{E}}_{\mathbf{c}'\in C_{\mathbf{y}}}\left[f(wt(\mathbf{c}')\right] - \mathop{\mathbf{E}}_{\mathbf{c}\in C}\left[f(wt(\mathbf{c})\right].$$

***Proof*.** Indeed, by definition

$$\mathop{\mathbf{E}}_{\mathbf{c}'\in C_{\mathbf{y}}}\left[f(wt(\mathbf{c}')\right] = \frac{1}{2}\mathop{\mathbf{E}}_{\mathbf{c}\in C}\left[f(wt(\mathbf{c})\right] + \frac{1}{2}\mathop{\mathbf{E}}_{\mathbf{c}\in C+\mathbf{y}}\left[f(wt(\mathbf{c})\right].$$

∎

We will make crucial use of the following result from [14], which states that the weight distribution of a linear code appears binomial to all polynomials of degree bounded by the distance of its dual. For the sake of completeness, we present its proof in the appendix.

**Lemma 8** ([14])**.** *Let $t \geqslant 1$ be an integer. Let $C$ be a binary linear code of block length $n$ such that $C^{\perp}$ has minimum distance at least $2t+1$. Then for every polynomial $f(X)$ of degree at most $2t$ over the reals:*

$$\sum_{i=0}^{n} A_i(C) f(i) = \frac{|C|}{2^n}\sum_{i=0}^{n}\binom{n}{i} f(i).$$

The above result implies the following result:

**Lemma 9** ([14])**.** *Let $t \geqslant 1$ be an integer. Let $C$ be a binary linear code of block length $n$ such that $C^{\perp}$ has minimum distance at least $2t+1$. Then the following holds for every $\mathbf{y} \in \mathbb{F}_2^n$:*

$$\sum_{\mathbf{c}\in C}\left(1 - 2\frac{wt(\mathbf{y}+\mathbf{c})}{n}\right)^{2t} \leqslant 2\cdot\frac{|C|}{n^t}\cdot\frac{(2t)!}{2^t t!}.$$

***Proof*.** By definition, $C \subseteq C_{\mathbf{y}}$ and thus, $C_{\mathbf{y}}^{\perp} \subseteq C^{\perp}$. In other words, $C_{\mathbf{y}}^{\perp}$ has a minimum distance of at least $2t+1$. This implies that by Lemma 8, the following are true:

$$\sum_{i=0}^{n} A_i(C_{\mathbf{y}})\left(1-\frac{2i}{n}\right)^{2t} = \frac{|C_{\mathbf{y}}|}{2^n}\sum_{i=0}^{n}\binom{n}{i}\left(1-\frac{2i}{n}\right)^{2t},$$

$$\sum_{i=0}^{n} A_i(C)\left(1-\frac{2i}{n}\right)^{2t} = \frac{|C|}{2^n}\sum_{i=0}^{n}\binom{n}{i}\left(1-\frac{2i}{n}\right)^{2t}.$$

The above along with Lemma 7 and (4), implies that

$$\mathop{\mathbf{E}}_{\mathbf{c}\in C}\left[\left(1-2\frac{wt(\mathbf{y}+\mathbf{c})}{n}\right)^{2t}\right] = \frac{1}{2^n}\sum_{i=0}^{n}\left(1-\frac{2i}{n}\right)^{2t}.$$

The lemma follows from the above relation and the following inequality (which is proved in Appendix A of [14]):

$$\frac{1}{2^n}\sum_{i=0}^{n}\binom{n}{i}(n-2i)^{2t} \leqslant 2n^t\frac{(2t)!}{2^t t!}.$$

∎

### 4.2 Moments of coset weight distribution of dual BCH codes

Let $t, m \geqslant 1$ be integers such that $2t - 2 < 2^{m/2}$. Recall that an $[2^m - 1, 2^m - 2t - 1, 2t + 1]_{2^m}$ Reed-Solomon code $C_1$ is the evaluation of degree $2^m - 2t - 1$ polynomials over all the non-zero elements in $\mathbb{F}_{2^m}$. Let $C_{BCH(t,m)}$ denote the code $C_1 \cap \mathbb{F}_2^{2^m-1}$. $C_{BCH(t,m)}$ is the well known BCH code.

**Theorem 10** ([19]). *Let $m, t \geqslant 1$ be integers such that $2t - 2 < 2^{m/2}$. Then $C_{BCH(t,m)}$ is an $[2^m - 1, 2^m - mt - 1, d]_2$ code, where $d \geqslant 2t + 1$.*

Let $C_{dBCH(t,m)} = (C_{BCH(t,m)})^{\perp}$. Thus, $C_{dBCH(t,m)}$ is an $[n \stackrel{def}{=} 2^m - 1, k \stackrel{def}{=} mt]_2$ code. Note that Lemma 9 can be applied to $C_{dBCH(t,m)}$. Further, $|C| = 2^{mt} \leqslant (2(2^m - 1))^t = 2^t n^t$. Thus, by Lemma 9 we have

**Lemma 11.** *Let $m, t \geqslant 1$ be integers such that $2t - 2 < 2^{m/2}$. Then the following is true for any $\mathbf{y} \in \mathbb{F}_2^n$:*

$$\sum_{\mathbf{c} \in C_{dBCH(t,m)}} \left(1 - 2\frac{\Delta(\mathbf{y}, \mathbf{c})}{n}\right)^{2t} \leqslant \frac{2(2t)!}{t!}.$$

We will also need the following property of the dual BCH code.

**Theorem 12** (Weil-Carlitz-Uchiyama Bound, [19]). *Let $m, t \geqslant 1$ be integers such that $2t - 2 < 2^{m/2}$. Then for every non-zero $\mathbf{c} \in C_{dBCH(t,m)}$,*

$$2^{m-1} - (t-1)2^{m/2} \leqslant wt(\mathbf{c}) \leqslant 2^{m-1} + (t-1)2^{m/2}.$$

# 5 Binary codes list decodable in presence of high noise

### 5.1 Construction with near-cubic dependence on $\varepsilon$

We will prove the following result in this section.

**Theorem 13.** *Given an integer $\mathcal{K} > 1$ and reals $\gamma > 0$ and $0 < \varepsilon < 1/2$, the following holds. There exits an explicit binary linear code of dimension at least $\mathcal{K}$ and block length $\mathcal{N}$ at most $\left(\frac{1}{\gamma}\right)^{O(1)} \cdot \left(\frac{\mathcal{K}^3}{\varepsilon^{3+\gamma}}\right)$ with the following properties:*

*(i) There is an algorithm that can list decode the code from a fraction $(1/2 - \varepsilon)$ of errors in time at most $\left(\frac{\mathcal{K}}{\gamma\varepsilon}\right)^{O(1/\gamma)}$ and outputs a list of size at most $\left(\frac{1}{\gamma\varepsilon}\right)^{O(1/\gamma)}$.*

*(ii) The code is $\frac{5\varepsilon}{6}$-biased, i.e., all the codewords have Hamming weights in the range $\left[(\frac{1}{2} - \frac{5\varepsilon}{6})\mathcal{N}, (\frac{1}{2} + \frac{5\varepsilon}{6})\mathcal{N}\right]$.*

In the rest of this section, we will prove this theorem. Given $\mathcal{K}$, $\varepsilon$ and $\gamma$ we will pick positive integer parameters $r$, $q$, $K$, $N$, $s$, $t$ and $m$ (for use in Theorem 4 and choosing the inner dual BCH code) as follows.

We pick $K = \left\lceil \frac{2\mathcal{K}}{\log \mathcal{K}} \right\rceil$. Let $t = \left\lceil \frac{1+\gamma}{2\gamma} \right\rceil$, and $s = 2t - 1$.

Let $r = 2$ and $q$ be the smallest power of 2 larger than $\left(\frac{2(2t)!}{t!}\right)^{1/s} \frac{K}{(\varepsilon/(2s+2))^{1+1/s}}$. In particular,

$$\left(\frac{2(2t)!}{t!}\right)^{1/s} \frac{K}{(\varepsilon/(2s+2))^{1+1/s}} \leqslant q \leqslant 2\left(\frac{2(2t)!}{t!}\right)^{1/s} \frac{K}{(\varepsilon/(2s+2))^{1+1/s}}. \quad (5)$$

The reason for the specific choice of $q$ is to ensure a decoding radius of $(1/2 - \varepsilon)$ and will become clear at the end of the proof.

Finally we take $N = q$ and define $m = \left\lceil \frac{s \log q}{t} \right\rceil$.

We first verify that with these setting of parameters, $2t - 2 < 2^{m/2}$. Indeed, $2^{m/2} \geqslant q^{s/(2t)} \geqslant (2s+2)^{(1+1/s)\cdot s/(2t)} = 2s + 2 > 2t - 2$, where the relationships follow from the choices of $m, q$ and $s$ respectively.

Let $C_{\text{out}}$ be the $\mathbb{F}_2$-linear code over $\mathbb{F}_q$ from Theorem 4 with dimension $K$ and block length $N$ for parameter $s$. Let $C_{dBCH(t,m)}$ be the $[n = 2^m - 1, k = mt]_2$ dual BCH code (note that the condition of Theorem 10 is satisfied). Note that $k \geqslant s \log q$, and we take $C_{\text{in}}$ to be an arbitrary subspace of $C_{dBCH(t,m)}$ of dimension $k' = s \log q$.

The binary code $C$ with the claimed properties of Theorem 13 will be the concatenated code $C_{\text{out}} \circ C_{\text{in}}$. The claim on explicitness of $C$ follows from the fact that both $C_{\text{out}}$ and $C_{\text{in}}$ are explicit codes. We begin by verifying the parameters of $C$.

The dimension of $C$ equals $Ks \log q$ which is clearly at least $\mathcal{K}$ by the choice of $K$ and $q$. The block length $n = 2^m - 1$ of $C_{\text{in}}$ satisfies $q^{2-1/t} - 1 \leqslant n \leqslant 2q^{2-1/t}$. Now, the block length of $C$ satisfies

$$\mathcal{N} = nN \leqslant 2q^{3-1/t} \leqslant 16\left(\frac{2(2t)!}{t!}\right)^{(3t-1)/(ts)} \frac{K^{3-1/t}}{(\varepsilon/(2s+2))^{(3-1/t)(1+1/s)}}. \quad (6)$$

It can be verified that by our choices of $s, t$ and $K$, $K^{3-1/t}$ is $O(\mathcal{K}^3)$, $(3-1/t)(1+1/s) \leqslant 3 + \gamma$ and $16\left(\frac{2(2t)!}{t!}\right)^{(3t-1)/(ts)} (2s+2)^{(3-1/t)(1+1/s)}$ is $(1/\gamma)^{O(1)}$, which proves the claimed bound on $\mathcal{N}$.

We now verify the claim that $C$ is $\frac{5\varepsilon}{6}$-biased. First, note that since $C_{\text{out}}$ is $\mathbb{F}_2$-linear and $C_{\text{in}}$ is linear, the concatenated code is a binary linear code. (5) and the fact that $N = q$ implies that $K/N \leqslant \varepsilon$. This along with Theorem 4 implies that $C_{\text{out}}$ has relative distance at least $1 - \varepsilon$. Thus, by Theorem 12, all the non-zero codewords in $C$ have relative Hamming weights in the range $\left[(1-\varepsilon)\left(\frac{2^{m-1}}{2^m-1} - \frac{(t-1)2^{m/2}}{2^m-1}\right), \frac{2^{m-1}}{2^m-1} + \frac{(t-1)2^{m/2}}{2^m-1}\right]$. We first claim that $\left[\frac{2^{m-1}}{2^m-1} - \frac{(t-1)2^{m/2}}{2^m-1}, \frac{2^{m-1}}{2^m-1} + \frac{(t-1)2^{m/2}}{2^m-1}\right] \subseteq \left[\frac{1}{2} - \frac{t+1}{2^{m/2}}, \frac{1}{2} + \frac{t+1}{2^{m/2}}\right]$. One can check that this inclusion is satisfied if

$$2^{m/2+1} \geqslant \frac{t+1}{2^{m/2}} + \frac{1}{2}.$$

Using the fact that $2t - 2 < 2^{m/2}$, the above is satisfied if $2^{m/2+1} \geqslant 1 + 2/2^{m/2}$, which is true for $m \geqslant 1$. The claim on $C$ being $\frac{5\varepsilon}{6}$-biased follows from the fact that $\frac{t+1}{2^{m/2}} \leqslant \varepsilon/3$. The latter inequality is easily checked, since

$$2^{m/2} \geqslant q^{s/(2t)} = q^{1-1/(2t)} \geqslant \frac{1}{(\varepsilon/(2s+2))^{(1+1/s)(1-1/(2t))}} = 4t/\varepsilon \geqslant 3(t+1)/\varepsilon\ ,$$

where the last inequality follows from the fact that $t \geqslant 1$.

Finally, we turn to the claims on list decodability of $C$. The list decoding algorithm for $C$ is as follows. Let $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_N) \in (\mathbb{F}_2^n)^N$ be the received word. We now define appropriate weights for use in the soft decoding algorithm from Theorem 4. For every $1 \leqslant i \leqslant N$ and $\alpha \in \mathbb{F}_{q^s}$ define:

$$w_{i,\alpha} = \max\left(1 - \frac{2\Delta(\mathbf{y}_i, C_{\mathrm{in}}(\alpha))}{n}, 0\right). \tag{7}$$

Define $w_{\max} = \max_{i,\alpha} w_{i,\alpha}$. (Since $\mathbf{y}$ is promised to have a codeword of $C$ within distance $\mathcal{N}/2$, $w_{\max} > 0$.) Finally, run the soft decoder from Theorem 4 with the choice

$$\zeta = \frac{\varepsilon}{2w_{\max}} . \tag{8}$$

We will defer the analysis of the running time and worst case list size to the end of this section. Next, we show that the algorithm above outputs a list that contains $\mathbf{c}'$, for any codeword $\mathbf{c}' \in C$ such that $\Delta(\mathbf{c}', \mathbf{y}) \leqslant (1/2 - \varepsilon)\mathcal{N}$.

As $s+1 = 2t$ is even, $w_{i,\alpha}^{s+1} \leqslant \left(1 - 2\frac{\Delta(\mathbf{y}_i, C_{\mathrm{in}}(\alpha))}{n}\right)^{s+1}$. Since $C_{\mathrm{in}}$ is a subcode of the dual BCH code $C_{dBCH(t,m)}$ (and $2t - 2 < 2^{m/2}$), Lemma 11 implies the following property of these weights (for every $1 \leqslant i \leqslant N$):

$$\sum_{\alpha \in \mathbb{F}_{q^s}} w_{i,\alpha}^{s+1} \leqslant \frac{2(2t)!}{t!}. \tag{9}$$

Let $\mathbf{c} = (c_1, \ldots, c_N) \in C_{\mathrm{out}}$ be the intermediate codeword corresponding to $\mathbf{c}'$. Thus, if we use the soft decoding algorithm from Theorem 4 with the choice $\zeta = \frac{\varepsilon}{2w_{max}}$, it will output $\mathbf{c}$ if the following condition is satisfied:

$$\sum_{i=1}^{N} w_{i,c_i} \geqslant (s+1) \sqrt[s+1]{K^s \sum_{i=1}^{N} \sum_{\alpha \in \mathbb{F}_{q^s}} w_{i,\alpha}^{s+1}} + \frac{\varepsilon N}{2}$$

Now as $w_{i,\alpha} \geqslant \left(1 - 2\frac{\Delta(\mathbf{y}_i, C_{\mathrm{in}}(\alpha))}{n}\right)$, the above condition is satisfied if the following is true:

$$\sum_{i=1}^{N} \left(1 - 2\frac{\Delta(\mathbf{y}_i, C_{\mathrm{in}}(c_i))}{n}\right) \geqslant (s+1) \sqrt[s+1]{K^s \sum_{i=1}^{N} \sum_{\alpha \in \mathbb{F}_{q^s}} w_{i,\alpha}^{s+1}} + \frac{\varepsilon}{2} N .$$

By (9) the above will be satisfied for $\mathbf{c}$ if (where we have used the fact that $\sum_{i=1}^{N} \Delta(\mathbf{y}_i, C_{\mathrm{in}}(c_i)) = \Delta(\mathbf{y}, \mathbf{c}')$):

$$N - 2\frac{\Delta(\mathbf{c}', \mathbf{y})}{n} \geqslant (s+1) \sqrt[s+1]{\frac{2(2t)!}{t!} K^s N} + \varepsilon N/2,$$

or equivalently

$$1 - 2\frac{\Delta(\mathbf{c}', \mathbf{y})}{\mathcal{N}} \geqslant (s+1) \sqrt[s+1]{\frac{2(2t)!}{t!} \left(\frac{K}{N}\right)^s} + \frac{\varepsilon}{2} .$$

From the lower bound in (5) and the fact that $N = q$, the above condition is met as long as $\Delta(\mathbf{c}', \mathbf{y}) \leqslant \frac{1}{2}(1-\varepsilon)\mathcal{N}$. This proves the claim that $C$ can be list decoded from a fraction $1/2 - \varepsilon$ of errors.

We now bound the worst case list size output by our list decoding algorithm. By Theorem 4, the worst case list size is at most $(2s)^{O(s)}\left(\frac{1}{w_{max}\zeta}\right)^{s+1} \cdot \sum_{i,\alpha} w_{i,\alpha}^{s+1}$, which by (9) and the choice of $\zeta$ in (8) is upper bounded by $(s/\varepsilon)^{O(s)}$. The claimed bound on the worst case list size follows from the choice of $s$.

To complete the proof, we bound the running time of the list decoding algorithm, which has two parts. The first part is the time needed to compute the weights as defined by (7). Computing the weights in a brute-force manner takes time $O(nNq^s)$, which by the choices of $n$, $N$, $q$, $\mathcal{N}$ and $s$ is upper bounded by $\mathcal{N}^{O(1/\gamma)}$. The second component of the running time, by Theorem 4 and the choice of $q$ and $\mathcal{N}$ is upper bounded by $(s/\varepsilon)^{O(s)}\mathcal{N}^{O(1)}$. The total running time is thus bounded by $\left(\frac{\mathcal{N}}{\gamma\varepsilon}\right)^{O(1/\gamma)}$. The bound on $\mathcal{N}$ implies a running time of $O\left(\frac{\mathcal{K}}{\gamma\varepsilon}\right)^{O(1/\gamma)}$, as desired. This completes the proof of Theorem 13.

### 5.2 Construction with linear dependence on dimension

Using an alternate concatenated code construction, with a different inner code but again the PV code as outer code, we prove the following result in this section.

**Theorem 14.** *Given an integer $\mathcal{K} > 1$ and reals $0 < \gamma, \varepsilon < 1/2$, the following holds. There exists a binary linear code of dimension at least $\mathcal{K}$ and block length $\mathcal{N} \leqslant O\left(\frac{\mathcal{K}\log(1/\varepsilon)}{\gamma^{1+\gamma}\varepsilon^{5+\gamma}}\right)$ with the following properties:*

*(i) There is an algorithm that can list decode the code from a fraction $1/2 - \varepsilon$ of errors in time at most $(\mathcal{K}/(\gamma\varepsilon))^{O(1/\gamma)}$ and outputs a list of size at most $O\Big((1/\gamma)^{O(1/\gamma)} \cdot (1/\varepsilon)^{3+\gamma}\Big)$.*

*(ii) There is a randomized* Las Vegas *construction of the code in time $(\mathcal{K}/(\gamma\varepsilon))^{O(1/\gamma)}$ that provides a* certified *guarantee of the list decoding property claimed in (i).*

*(iii) The code is $\frac{3\varepsilon}{4}$-biased, that is, all nonzero codewords have Hamming weights in the range $\left[(\frac{1}{2} - \frac{3\varepsilon}{4})\mathcal{N}, (\frac{1}{2} + \frac{3\varepsilon}{4})\mathcal{N}\right]$.*

***Proof.*** Given $\mathcal{K}, \varepsilon$ and $\gamma$, pick parameters $r, q, K, N, s$ for the codes guaranteed by Corollary 5 as follows. Let $K = \left\lceil \frac{2\mathcal{K}}{\log \mathcal{K}} \right\rceil$, and $s = \lceil \frac{3}{\gamma} \rceil$. Let $r = 2$ and take $q$ to be the smallest power of 2 greater than $\frac{32K(s+1)^{1+1/s}}{\varepsilon^{1+3/s}}$, and pick $N = q$.

Let $C_{\text{out}}$ be the $\mathbb{F}_2$-linear code guaranteed by Corollary 5 with dimension $K$ and block length $N$ for parameter $s$. For the inner code $C_{\text{in}}$, we will use any $[n,k]_2$ code that is $\varepsilon^2/4$-biased (that is, all non zero-codewords have relative Hamming weights in $[\frac{1}{2} - \frac{\varepsilon^2}{4}, \frac{1}{2} + \frac{\varepsilon^2}{4}]$) for $k = s\log_2 q$. It is known that for $n = \Theta(k/\varepsilon^4)$, a random linear $[n,k]_2$ code is $\varepsilon^2/4$-biased with high probability (and in particular, lies on the Gilbert-Varshamov bound). Thus, one can construct a generator matrix for $C_{\text{in}}$ in randomized $O(k^2/\varepsilon^4)$ time by sampling a random matrix and confirming that the resulting code has the needed property by brute-force in $2^{O(k)} = q^{O(s)} = (K/(\gamma\varepsilon))^{O(1/\gamma)}$ time. This gives a polynomial time *Las Vegas* construction of our final code $C^*$, which is the concatenation of $C_{\text{out}}$ and $C_{\text{in}}$.

The proof that $C^*$ is $\varepsilon$-biased is similar to the argument in Theorem 13. By the choices of $K$ and $N$, $K/N \leqslant \varepsilon$, which by Theorem 4 implies that $C_{\text{out}}$ has relative distance $1-\varepsilon$. Thus, all the non-zero codewords have relative Hamming weights in the range $\left[(1-\varepsilon)\left(\frac{1}{2}-\frac{\varepsilon^2}{4}\right), \frac{1}{2}+\frac{\varepsilon^2}{4}\right] \subseteq \left[\frac{1}{2}-\frac{3\varepsilon}{4}, \frac{1}{2}+\frac{3\varepsilon}{4}\right]$, as required.

The dimension of $C^*$ is $Kk = Ks\log q$ which is easily seen to be at least $\mathcal{K}$. The block length $\mathcal{N} = Nn$ of $C^*$ satisfies

$$\begin{aligned}\mathcal{N} \leqslant \frac{64K(s+1)^{1+1/s}}{\varepsilon^{1+3/s}} \cdot O(\log K + \log(1/\varepsilon)) \cdot O(1/\varepsilon^4) &\leqslant O(s^{1+1/s}\mathcal{K}\log(1/\varepsilon)/\varepsilon^{5+3/s}) \\ &\leqslant O(1/\gamma^{1+\gamma} \cdot \mathcal{K}\log(1/\varepsilon)/\varepsilon^{5+\gamma}) \ .\end{aligned}$$

It remains to give a list decoding algorithm for $C^*$ that can correct a fraction $(1/2-\varepsilon)$ of errors. This follows from a rather standard method, and we give the details for completeness below. Let $\mathbf{y} \in \mathbb{F}_2^{\mathcal{N}}$ be a received word. If $\mathbf{c} \in C^*$ agrees with $\mathbf{y}$ on at least $(1/2+\varepsilon)\mathcal{N}$ positions, clearly on at least $\varepsilon N/2$ of the inner encoding blocks, $\mathbf{c}$ and $\mathbf{y}$ agree on at least $\left(\frac{1}{2}+\frac{\varepsilon}{2}\right)n$ positions. Since $C_{\text{in}}$ has relative distance at least $\left(\frac{1}{2}-\frac{\varepsilon^2}{4}\right)$, by the Johnson bound (Lemma 6), we know that $C_{\text{in}}$ is $((1-\varepsilon)/2, 4/\varepsilon^2)$-list-decodable. Therefore, we can decode $\mathbf{y}$ by decoding each inner block to a radius of $(1-\varepsilon)n/2$, returning a set $S_i$ of at most $4/\varepsilon^2$ symbols of $\mathbb{F}_{q^s}$ for each position of $C_{\text{out}}$. We can then run the list recovery algorithm guaranteed by Corollary 5 for $C_{\text{out}}$ and output all outer codewords whose $i$'th symbols belong to $S_i$ for at least $\varepsilon N/2$ positions.

All that remains to be done is to check that the list recovery Condition (3) of Corollary 5 is met with the choice $\alpha = \varepsilon/2$, $\ell = 4/\varepsilon^2$, and the above choice of $K, N, s$. Of course the specific choice above was made precisely in order to satisfy this. Indeed, we have $K/N \leqslant \varepsilon^{1+3/s}/(32(s+1)^{1+1/s})$, so the required condition is met if

$$\frac{\varepsilon}{2} > \varepsilon^{\frac{s+3}{s+1}}(1/32)^{\frac{s}{s+1}}(4/\varepsilon^2)^{\frac{1}{s+1}} = \frac{\varepsilon}{2^{\frac{5s-2}{s+1}}} \ ,$$

which is true for every $s \geqslant 1$. The bound on the list size follows from Corollary 5 and noting that $s = O(1/\gamma)$, $\ell = O(1/\varepsilon^2)$, $r = 2$ and $N/K = O(s^2/\varepsilon^{1+3/s})$. The decoding complexity consisting of two parts: the time to decode the inner codes by brute-force which takes $q^{O(s)}$ time, and the time to run the list recovering algorithm for the outer code, which takes time $s^{O(s)}(q\ell)^{O(1)}$ by Corollary 5. The combined runtime is thus bounded by $(\mathcal{N}/(\gamma\varepsilon))^{O(1/\gamma)}$, which in turn is bounded by $(\mathcal{K}/(\gamma\varepsilon))^{O(1/\gamma)}$. ■

**Remark 1.** One shortcoming of the above result is that the construction is only randomized (Las Vegas) polynomial time. In many applications this is okay since the accompanying list decoding algorithm is polynomial time and is *certified* to be able to correct a fraction $(1/2-\varepsilon)$ of errors.

If we knew a deterministic construction of an $[n,k]_2$ binary linear that meets (or gets close to) the Gilbert-Varshamov bound and further had a running time of $2^{O(k)}$ (as opposed to the $2^{O(n-k)}$ time construction known by finding a good $(n-k)\times n$ parity check matrix column-by-column), then we will be able to achieve the result of Theorem 14 with a deterministic construction. Note that such a result will be interesting for the regime $k = o(n)$, which is the case for the application in Theorem 14. Obtaining such a construction seems like a very interesting question.

## 6 Negative Result

In this section, we will state the following limit on codes that are list decodable from a very high fraction of errors. For the sake of completeness, we present its (standard) proof below.

**Theorem 15.** *Let $a > 0$ be real. Let $C$ be a binary code of block length $n$. If $C$ is a $\left(\frac{1}{2} - a\sqrt{\frac{\log n}{n}}, L\right)$-list decodable code then*

$$|C| \leqslant L \cdot n^{4a^2 \log e + 1}.$$

To prove Theorem 15, we will need the following lemma.

**Lemma 16.** *Let $\mathbf{x} \in \{0,1\}^n$ be a fixed vector. Then for a vector $\mathbf{y}$ chosen uniformly at random from $\{0,1\}^n$, the following is true for large enough $n$:*

$$\Pr\left[\Delta(\mathbf{x},\mathbf{y}) \leqslant \frac{n}{2} - a\sqrt{n\log n}\right] \geqslant \frac{1}{n^{4a^2 \log e + 1}}.$$

The lemma follows (with slightly different constants) from a lower bound on the tail of a Binomial distribution (cf. [18, Chap. 4]). For the sake of completeness, we present a proof in Appendix B. Armed with the lemma above, let us prove Theorem 15.

**Proof of Theorem 15:** Fix an arbitrary codeword $\mathbf{c} \in C$. Pick a received word $\mathbf{y} \in \{0,1\}^n$ uniformly at random. By Lemma 16,

$$\Pr_{\mathbf{y}}\left[\mathbf{c} \in B\left(\mathbf{y}, \frac{n}{2} - a\sqrt{n\log n}\right)\right] \geqslant \frac{1}{n^{4a^2 \log e + 1}},$$

where $B(\mathbf{y}, e')$ is the Hamming ball of radius $e'$ centered at $\mathbf{y}$. Since the choice of $\mathbf{c}$ was arbitrary,

$$\mathbb{E}_{\mathbf{y}}\left[\left|C \cap B\left(\mathbf{y}, \frac{n}{2} - a\sqrt{n\log n}\right)\right|\right] \geqslant \frac{|C|}{n^{4a^2 \log e + 1}}.$$

By the assumption on the list-decodability of $C$, for every $\mathbf{y} \in \{0,1\}^n$,

$$\left|C \cap B\left(\mathbf{y}, \frac{n}{2} - a\sqrt{n\log n}\right)\right| \leqslant L,$$

which implies that

$$\frac{|C|}{n^{4a^2 \log e + 1}} \leqslant L,$$

as desired. $\square$

Note that if a code $C$ has polynomial time list decoding algorithm, then the worst case list size has to be bounded by a polynomial in the block length of the code. Thus, we have the following corollary of Theorem 15:

**Corollary 17.** *Let $C$ be a binary code of block length $n$ that can be list decoded from a fraction $1/2 - O\left(\sqrt{\frac{\log n}{n}}\right)$ of errors in time polynomial in $n$. Then $C$ has a dimension of $O(\log n)$.*

## 7 Open Questions

The most obvious question left open by our work is to design binary codes that are list decodable from a fraction $1/2 - \varepsilon$ of errors in polynomial time with $n = k^{O(1)}/\varepsilon^a$ for some $2 \leqslant a < 3$. One might hope that using a better inner code in our construction might lead to such an improvement. We now informally argue that such an improvement will most probably not come from *only* choosing another inner code. To obtain a concatenated code that can correct a fraction $1/2 - \varepsilon$ of errors, we will need to come up with an outer and inner code that can list decode from a fraction $(1 - \varepsilon)$ and $(1/2 - \varepsilon)$ of errors respectively. These imply an upper bound of $O(\varepsilon)$ and $O(\varepsilon^2)$ on the rates of such codes (this follows from known upper bounds on rates of such codes). Thus, the resulting concatenated code will have a rate of at most $O(\varepsilon^3)$.

The argument above uses the fact that most of the currently known list decoding algorithms for concatenated codes run in two stages. In the first stage the inner codes are decoded while ignoring the structure of the outer code. In the second stage one uses the intermediate information from the first stage to decode the outer code. It seems likely that carefully chosen soft information to pass from the inner decodings to the outer algebraic decoder (see [16, 13] for examples of such decoders) may hold the key to further progress in list decoding concatenated codes.

### Acknowledgments

The second author would like to thank Ken Regan and Alan Selman for helpful discussions. We thank the anonymous reviewers of CCC'08 for useful suggestions that improved the presentation of the paper.

## References

[1] N. Alon, J. Bruck, J. Naor, M. Naor, and R. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38:509–516, 1992.

[2] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple constructions of almost $k$-wise independent random variables. *Random Structures and Algorithms*, 3:289–304, 1992.

[3] J.-Y. Cai, A. Pavan, and D. Sivakumar. On the hardness of the permanent. *Proceedings of the 16th International Symposium on Theoretical Aspects of Computer Science*, March 1999.

[4] P. Elias. Error-correcting codes for list decoding. *IEEE Transactions on Information Theory*, 37:5–12, 1991.

[5] U. Feige, M. Langberg, and K. Nissim. On the hardness of approximating NP witnesses. *Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pages 120–131, September 2000.

[6] A. Gál, S. Halevi, R. J. Lipton, and E. Petrank. Computing from partial solutions. *Proceedings of the 14th Annual IEEE Conference on Computation Complexity*, pages 34–45, 1999.

[7] O. Goldreich and L. Levin. A hard-core predicate for all one-way functions. *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 25–32, May 1989.

[8] V. Guruswami. *List decoding of error-correcting codes.* Number 3282 in Lecture Notes in Computer Science. Springer, 2004.

[9] V. Guruswami, J. Hastad, M. Sudan, and D. Zuckerman. Combinatorial bounds for list decoding. *IEEE Transactions on Information Theory*, 48(5):1021–1035, 2002.

[10] V. Guruswami and A. Rudra. Explicit capacity-achieving list-decodable codes. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 1–10, May 2006.

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

[12] V. Guruswami and M. Sudan. List decoding algorithms for certain concatenated codes. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 181–190, 2000.

[13] V. Guruswami and M. Sudan. Decoding concatenated codes using soft information. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity (CCC)*, pages 148–157, 2002.

[14] T. Kaufman and S. Litsyn. Almost orthogonal linear codes are locally testable. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 317–326, 2005.

[15] M. Kiwi. Testing and weight distributions of dual codes. *ECCC Technical Report TR-97-010*, 1997.

[16] R. Koetter. On optimal weight assignments for multivariate interpolation list-decoding. In *Proc. 2006 IEEE Information Theory Workshop*, pages 37–41, March 2006.

[17] S. R. Kumar and D. Sivakumar. Proofs, codes, and polynomial-time reducibilities. *Proceedings of the 14th Annual IEEE Conference on Computation Complexity*, 1999.

[18] M. Ledoux and M. Talagrand. *Probability in Banach Spaces.* Springer, 1991.

[19] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes.* Elsevier/North-Holland, Amsterdam, 1981.

[20] E. Mossel and C. Umans. On the complexity of approximating the VC dimension. *J. Comput. Syst. Sci.*, 65(4):660–671, 2002. Preliminary version in the 16th Annual Conference on Computational Complexity (CCC), 2001.

[21] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

[22] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 285–294, 2005.

[23] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudo-random generator. *Journal of the ACM*, 52(2):172–216, 2005.

[24] D. Sheldon and N. E. Young. Hamming approximation of NP witnesses, November 2003. Manuscript.

[25] D. Sivakumar. On membership comparable sets. *J. Comput. Syst. Sci.*, 59(2):270–280, October 1999.

[26] M. Sudan. List decoding: Algorithms and applications. *SIGACT News*, 31:16–27, 2000.

[27] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom generators without the XOR lemma. *J. Comput. Sys. Sci.*, 62(2):236–266, 2001.

[28] V. V. Zyablov and M. S. Pinsker. List cascade decoding. *Problems of Information Transmission*, 17(4):29–34, 1981 (in Russian); pp. 236-240 (in English), 1982.

# A Proof of Lemma 8

The proof uses the MacWilliams transform and Krawtchouk polynomials, which we define next.

**Definition 1** (Krawtchouk Polynomial)**.** *The Krawtchouk polynomial of degree $j$ is defined as follows:*

$$P_j^n(X) = \sum_{i=0}^{j} (-1)^i \binom{X}{j}\binom{n-X}{j-i}.$$

The MacWilliams transform relates the weight distribution of of a binary linear code to the weight distribution of its dual.

**Theorem 18** (MacWilliams Transform [19])**.** *For a binary linear code $C$ of block length $n$,*

$$A_j(C^\perp) = \frac{1}{|C|} \cdot \sum_{i=0}^{n} A_i(C) P_j^n(i).$$

We will also need the fact that Krawtchouk polynomials are orthogonal:

**Lemma 19** ([15])**.** *Let $i, j, r, s$ and $n$ be non-negative integers. Then the following holds:*

$$\sum_{i=0}^{n} \binom{n}{i} P_r^n(i) P_s^n(i) = 2^n \binom{n}{r} \delta_{r,s},$$

*where $\delta_{r,s} = 1$ if $r = s$ and is $0$ otherwise.*

We will now present the proof of Lemma 8.

As the Krawtchouk polynomials form an orthogonal basis and $f(X)$ is a polynomial of degree at most $2t$,

$$f(X) = \sum_{j=0}^{2t} \alpha_j P_j^n(X), \tag{10}$$

for some suitable coefficients $\alpha_j$. In particular, Lemma 19 implies that

$$\alpha_0 = \frac{1}{2^n} \sum_{i=0}^{n} \binom{n}{i} f(i). \tag{11}$$

Thus, we have

$$\begin{aligned} \sum_{i=0}^{n} A_i(C) f(i) &= \sum_{j=0}^{2t} \alpha_j \sum_{i=0}^{n} A_i(C) P_j^n(i) && (12) \\ &= |C| \sum_{j=0}^{2t} \alpha_j A_j(C^\perp) && (13) \\ &= |C| \cdot \alpha_0 && (14) \\ &= \frac{|C|}{2^n} \cdot \sum_{i=0}^{n} \binom{n}{i} f(i). && (15) \end{aligned}$$

In the above (12) follows from (10). (13) follows from Theorem 18. (14) follows from the assumption that $C^\perp$ has distance of at least $2t+1$. Finally, (15) follows from (11). The proof is complete.

# B Proof of Lemma 16

Note that we need to prove that

$$\Pr_{\mathbf{y}} \left[ \mathbf{y} \in B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right) \right] \geqslant \frac{1}{n^{4a^2 \log e + 1}}.$$

Since $\mathbf{y}$ is chosen uniformly at random,

$$\Pr_{\mathbf{y}} \left[ \mathbf{y} \in B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right) \right] = \frac{\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right|}{2^n}.$$

To complete the proof, we will show that for large enough $n$:

$$\log \left( \left| B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right) \right| \right) \geqslant n - (4a^2 \log e + 1) \log n. \tag{16}$$

It is easy to check that

$$\left| B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right) \right| \geqslant \binom{n}{\frac{n}{2} - a\sqrt{n \log n}} = \frac{n!}{(n/2 - a\sqrt{n \log n})!(n/2 + a\sqrt{n \log n})!}.$$

Before we proceed, we recall Stirling's approximation for $b!$:

$$\sqrt{2\pi b} \left(\frac{b}{e}\right)^b e^{\lambda_1(b)} < b! < \sqrt{2\pi b} \left(\frac{b}{e}\right)^b e^{\lambda_2(b)}.$$

where

$$\lambda_1(b) = \frac{1}{12b+1} \text{ and } \lambda_2(b) = \frac{1}{12b}.$$

Thus,

$$\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n\log n}\right)\right| \geqslant E(n) \cdot \frac{n^n}{\left(\frac{n}{2} - a\sqrt{n\log n}\right)^{\frac{n}{2} - a\sqrt{n\log n}} \left(\frac{n}{2} + a\sqrt{n\log n}\right)^{\frac{n}{2} + a\sqrt{n\log n}}},$$

where

$$E(n) = \frac{1}{\sqrt{2\pi\left(\frac{n}{4} - a^2\log n\right)}} \cdot \frac{e^{\lambda_1(n)}}{e^{\lambda_2\left(\frac{n}{2} - a\sqrt{n\log n}\right) + \lambda_2\left(\frac{n}{2} + a\sqrt{n\log n}\right)}}.$$

For large enough $n$, $E(n) \geqslant \frac{1}{n}$, which implies that

$$\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n\log n}\right)\right| \geqslant \frac{n^{n-1}}{\left(\frac{n}{2} - a\sqrt{n\log n}\right)^{\frac{n}{2} - a\sqrt{n\log n}} \left(\frac{n}{2} + a\sqrt{n\log n}\right)^{\frac{n}{2} + a\sqrt{n\log n}}},$$

Thus,

$$\begin{aligned}
\log\left(\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n\log n}\right)\right|\right) &\geqslant (n-1)\log n - \left(\frac{n}{2} - a\sqrt{n\log n}\right)\left(\log\left(\frac{n}{2}\right) + \log\left(1 - 2a\sqrt{\frac{\log n}{n}}\right)\right) \\
&\quad - \left(\frac{n}{2} + a\sqrt{n\log n}\right)\left(\log\left(\frac{n}{2}\right) + \log\left(1 + 2a\sqrt{\frac{\log n}{n}}\right)\right) \\
&= n - \log n - \left(\frac{n}{2} - a\sqrt{n\log n}\right)\log\left(1 - 2a\sqrt{\frac{\log n}{n}}\right) \\
&\quad - \left(\frac{n}{2} + a\sqrt{n\log n}\right)\log\left(1 + 2a\sqrt{\frac{\log n}{n}}\right) \\
&\geqslant n - \log n + 2a\log e\sqrt{\frac{\log n}{n}} \cdot \left(\frac{n}{2} - a\sqrt{n\log n}\right) \\
&\quad - 2a\log e\left(\frac{n}{2} + a\sqrt{n\log n}\right)\sqrt{\frac{\log n}{n}} \qquad (17) \\
&= n - (4a^2\log e + 1)\log n.
\end{aligned}$$

In the above (17) follows from the fact that for any $0 \leqslant x < 1$, $\ln(1+x) \leqslant x$, $\ln(1-x) \leqslant -x$ and for large enough $n$, $2a\sqrt{\frac{\log n}{n}} < 1$. The proof is complete. □



ECCC ISSN 1433-8092

http://eccc.hpi-web.de/