List decoding of binary codes

(A brief survey of some recent results)

Venkatesan Guruswami[⋆]

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

Abstract. We briefly survey some recent progress on list decoding algorithms for *binary* codes. The results discussed include:

– Algorithms to list decode binary Reed-Muller codes of any order up to the minimum distance, generalizing the classical Goldreich-Levin algorithm for RM codes of order 1 (Hadamard codes). These algorithms are "local" and run in time polynomial in the *message* length.

– Construction of binary codes efficiently list-decodable up to the Zyablov (and Blokh-Zyablov) radius. This gives a factor two improvement over the error-correction radius of traditional "unique decoding" algorithms.

– The existence of binary linear *concatenated* codes that achieve list decoding capacity, i.e., the optimal trade-off between rate and fraction of worst-case errors one can hope to correct.

– Explicit binary codes mapping k bits to $n \leqslant \mathrm{poly}(k/\varepsilon)$ bits that can be list decoded from a fraction $(1/2-\varepsilon)$ of errors (even for $\varepsilon = o(1)$) in $\mathrm{poly}(k/\varepsilon)$ time. A construction based on concatenating a variant of the Reed-Solomon code with dual BCH codes achieves the best known (cubic) dependence on $1/\varepsilon$, whereas the existential bound is $n = O(k/\varepsilon^2)$. (The above-mentioned result decoding up to Zyablov radius achieves a rate of $\Omega(\varepsilon^3)$ for the case of *constant* ε.)

We will only sketch the high level ideas behind these developments, pointing to the original papers for technical details and precise theorem statements.

[⋆] Currently visiting the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. Research supported by NSF grants CCF-0343672 and CCF-0835814, and a Packard Fellowship.

1 Introduction

Shannon's capacity theorem states that for the binary symmetric channel BSC_p with crossover probability p,[1] there exist codes using which one can reliably communicate at any rate less than $1-H(p)$, and conversely, no larger rate is possible. (Here $H(\cdot)$ is the binary entropy function.) But what if the errors are *worst-case* and not random and independent? Specifically, if the channel is modeled as a "jammer" than can corrupt up to a fraction $p < 1/2$ of symbols in an arbitrary manner, what is the largest rate possible for error-free communication?

If we want a guarantee that the original message can be correctly and uniquely recovered, then this is just the question of the largest rate of a binary code every pair of whose codewords have different bits in at least a fraction $2p$ of the positions. This remains one of the biggest open questions in combinatorial coding theory. It is known, however, that in this case the rate has to be much smaller than $1 - H(p)$. The best rate known to be possible (even by non-constructive random coding methods) is the much smaller $1 - H(2p)$. (Note, in particular, that the rate is 0 already for $p > 1/4$.)

The above limitation arises due to the fact that, for rates close to Shannon capacity, there will be two codewords that both differ from some string in less than a fraction p of bits. However, the closest codeword is usually unique, and in those cases, it makes sense to decode up to a radius p. A clean way to model this algorithmic task is to relax the requirement on the error-correction algorithm and allow it to output a *small list* of messages in the worst-case (should there be multiple close-by codewords). This model is called *list decoding*. Perhaps surprisingly (and fortunately), using list decoding, one achieve a rate approaching the Shannon capacity $1-H(p)$, even if the errors are worst-case. Formally, there *exist* binary codes $C \subseteq \{0,1\}^n$ of rate $1-H(p)-\frac{1}{L}$ which are (p, L)-list-decodable, i.e., every Hamming ball of radius pn has at most L codewords of C [19, 1]. In fact, such binary *linear* codes exist [6]. If a codeword from such a code is transmitted and corrupted by at most a fraction p of errors, there will be at most L possible codewords that could have resulted in the received word. Thus it can be used for error recovery with an ambiguity of at most L. By allowing the worst-case list size L to grow, one can approach the best possible rate of $1 - H(p)$, which we call the *list decoding capacity*.

Thus, list decoding offers the potential of realizing the analog of Shannon's result for worst-case errors. However, the above is a *non-constructive* result. The codes achieving this trade-off are shown to exist via a random coding argument and are not explicitly specified. Further, for a code to be useful, the decoding algorithm must be efficient, and for a random, unstructured code only brute-force decoders running in exponential time are known.

[1] The BSC_p is a communication channel that transmits bits, and flips each bit independently with probability p.

Therefore, the grand challenge in the subject of list decoding binary codes is to give an explicit (polynomial time) construction of binary codes approaching list decoding capacity, together with an efficient list decoding algorithm. This remains a challenging long term goal that seems out of the reach of currently known techniques. For *large* alphabets, recent progress in algebraic list decoding algorithms [16, 8, 5] has led to the construction of explicit codes that achieve list decoding capacity — namely, they admit efficient algorithms to correct close to the optimal fraction $1 - R$ of errors with rate R. This in turn has spurred some (admittedly modest) progress on list decoding of binary codes, using code concatenation, soft decoding, and other techniques.

We give an informal discussion of some of this progress in this paper. The specific problems we discuss are those mentioned in the abstract of the paper, and are based on results in [3, 8, 10, 7, 9]. The second and third results (discussed in Sections 3 and 4) have straightforward extensions to codes over the field $\mathbb{F}_q$ with q elements for any fixed prime power q. The exact list decodability of Reed Muller codes over non-binary fields $\mathbb{F}_q$ remains an intriguing open problem (some progress is made in [3] but the bounds are presumably not tight). Our technical discussion below assumes that the reader is familiar with the basic background material and terminology of coding theory.

2 List decoding Reed-Muller codes

The first non-trivial algorithm for list decoding was the Goldreich-Levin algorithm for Hadamard codes (or first order Reed-Muller codes) [2]. The messages of the (binary) Reed-Muller code $\mathsf{RM}(m, r)$ of order r consist of m-variate multilinear polynomials of degree at most r over the binary field $\mathbb{F}_2$. The encoding of such a polynomial f consists of the evaluations $f(\mathbf{x})$ at all $\mathbf{x} \in \mathbb{F}_2^m$. The length of the encoding is thus 2^m. The minimum distance of $\mathsf{RM}(m, r)$ is 2^{m-r}.

The order 1 RM code corresponds to evaluations of *linear* polynomials on $\mathbb{F}_2^m$ and is often called the Hadamard code.[2] The Goldreich-Levin algorithm list decodes the Hadamard code up to a fraction $(1/2 - \varepsilon)$ of errors in time $\text{poly}(m/\varepsilon)$, outputting a list of size $O(1/\varepsilon^2)$. Note that the decoding radius approaches the relative distance, and further the runtime of the decoder is polynomial in the message length and *sub-linear* (in fact just polylogarithmic) in the length of the code (which is 2^m). The decoder is a "local" algorithm that only randomly probes a small portion of the received word in order to recover the messages corresponding to the close-by codewords.

An extension of the Goldreich-Levin algorithm to higher order RM codes was open for a long time. In recent work, Gopalan, Klivans, and Zuckerman [3]

[2] To be accurate, the Hadamard code only encodes linear polynomials with no constant term but this is a minor difference.

solved this problem, giving a local list decoding algorithm to correct a fraction $(2^{-r} - \varepsilon)$ errors for $\mathsf{RM}(m,r)$, i.e., arbitrarily close to the relative distance 2^{-r}. The algorithm runs in time $(m/\varepsilon)^{O(r)}$ and outputs a list of size at most $(1/\varepsilon)^{O(r)}$. This list size bound is also shown to be tight (up to constant factors in the exponent $O(r)$). Reed-Muller codes of constant order have only polynomially many codewords and thus have vanishing rate. Thus, this result is not directly related to the program of constructing binary codes with good trade-off between rate and list decoding radius. It is nevertheless an exciting development since Reed-Muller codes are one of the classical and most well-studied binary code constructions.

We now describe the approach behind the algorithm at a very informal level. Suppose we are given oracle access to a received word R which we think of as a function $R : \mathbb{F}_2^m \to \mathbb{F}_2$. The goal is to find all degree r polynomials f which differ from R on at most a fraction $(2^{-r} - \varepsilon)$ of points. The algorithm picks a random subspace A of $\mathbb{F}_2^m$ of size $a = 2^{O(r)}/\varepsilon^2$. Then it guesses the correct value of $f_{|A}$, the target polynomial f restricted to A (we remark on the number of such guesses shortly). Then given a point $\mathbf{b} \in \mathbb{F}_2^m$, the algorithm determines $f(\mathbf{b})$ as follows: Consider the subspace $B = A \cup (A + \mathbf{b})$. Run a unique decoding algorithm for a Reed-Muller code of order r restricted to B (correcting up to a fraction $\frac{1}{2} \cdot 2^{-r}$ of errors), to find a degree r polynomial $g_{|B}$, if one exists. Note that if the error rate on $A + \mathbf{b}$ w.r.t f is less than 2^{-r}, the error rate on B will be less than $2^{-(r+1)}$ and thus $g_{|B}$ must be $f_{|B}$. We will recover $f(\mathbf{b})$ correctly in this case from the corresponding value of g.

Since A is a random subspace of size a, by pairwise independence, with high probability (at least $1 - \frac{1}{a\varepsilon^2} = 1 - 2^{-\Omega(r)}$), the error rate on $A + \mathbf{b}$ is within ε of the original error rate, and therefore less than 2^{-r}. Therefore, with high probability over the choice of the subspace A, when the algorithm guesses $f_{|A}$ correctly, it will correctly compute $f(\mathbf{b})$ for all but a $2^{-\Omega(r)}$ fraction of points $\mathbf{b}$. The function computed by the algorithm is thus within fractional distance at most $2^{-(r+1)}$ from the encoding of f. Since the unique decoding radius of $\mathsf{RM}(m,r)$ is $2^{-(r+1)}$, running a local unique decoder on the function computed by the algorithm then returns f.

The list size is governed by the number of guesses for $f_{|A}$. When $r = 1$, since f is a linear polynomial, it suffices to guess the value on a basis for the subspace A which consists of $\log a = 2\log(1/\varepsilon) + O(1)$ points. This leads to the $O(1/\varepsilon^2)$ list-size bound for Hadamard codes. For $r > 1$, the total number of guesses becomes quasi-polynomial in $1/\varepsilon$. Using additional ideas, it is possible to improve this to $(1/\varepsilon)^{O(r)}$. The above description was meant to only give the flavor of the algorithm, and hides many subtleties. The reader can find the formal details and the arguments for the improved list size bound in [3]. List size bounds for decoding binary Reed-Muller codes *beyond* the minimum distance are established in [14].

3 List decoding concatenated codes up to Zyablov radius

As mentioned in the introduction, the problem of constructing explicit binary codes achieving list decoding capacity, i.e., binary codes of rate R list-decodable up to radius $H^{-1}(1-R)$, remains wide open. In this section, we report on recent constructions that achieve the best known trade-offs between rate and list decoding radius.

Guruswami and Rudra [8] construct a variant of Reed-Solomon codes, called folded Reed-Solomon codes, over an alphabet of size polynomial in the block length, that can list decode a fraction $(1-R_0-\varepsilon)$ of errors with rate R_0 for any desired $0 < R_0 < 1$ and any constant $\varepsilon > 0$. This achieves the list decoding capacity over large alphabets. A natural approach to construct binary codes is to concatenate these codes with optimal binary list-decodable codes at the inner level. Since the inner codes have only polynomially codewords, one can find by a greedy "brute-force" search such codes close to list decoding capacity, i.e., with rate r which are list-decodable up to a fraction $H^{-1}(1-r-\varepsilon)$ of errors (with list-size $O(1/\varepsilon)$). The search for the inner code is not based on the most obvious brute-force algorithm (which would take quasi-polynomial time in the block length of the outer code), but rather a greedy derandomization of the probabilistic method; see [6] for details.

The resulting concatenated code has rate rR_0, and an algorithm to list decode the code up to a fraction $(1-R_0)H^{-1}(1-r)-\varepsilon$ of errors is given in [8]. This leads to binary codes of rate R list-decodable up to the so-called *Zyablov radius* given by

$$\text{Zyablov}(R) = \max_{\substack{0<R_0,r\leqslant 1\\ R_0 r = R}} (1-R_0)H^{-1}(1-r) \ .$$

We note that the Zyablov radius is also the standard product bound on the relative distance of concatenated codes with an outer Maximum Distance Separable code and an inner binary code meeting the Gilbert-Varshamov bound. The above result is able to *decode* up to this radius. In comparison, traditional algorithms based on Generalized Minimum Distance decoding are able to (unique) decode up to *half* the Zyablov radius.

The idea behind the above algorithm is very natural. The various inner blocks are first decoded using a brute-force search algorithm up to a radius of $H^{-1}(1-r-\varepsilon)$. By the assumed list decoding properties of the inner code, this step returns a set of at most $\ell = O(1/\varepsilon)$ candidate symbols for each possible symbol of the folded Reed-Solomon codeword. If the fraction of errors is at most $(1-R_0)H^{-1}(1-r)-O(\varepsilon)$, then at most a $(1-R_0-\varepsilon)$ fraction of the sets returned by the inner decodings will fail to contain the correct outer symbol. Now comes the part where a crucial, powerful feature of the Guruswami-Rudra list decoder comes in handy — a fraction $(1-R_0-\varepsilon)$ of errors can be list decoded even if the input is not a received word but a collection of sets of possible symbols (of some bounded size, such as ℓ), one for each codeword position, and where a codeword position is counted as an error if the correct codeword symbol does not belong to the set of candidates corresponding to that position. This generalization of list decoding is called *list recovery* in the literature. The details of list recovering folded Reed-Solomon codes and formal details about the above algorithm can be found in [8].

In [10], the authors use multilevel concatenated codes to improve the above trade-off and construct codes list-decodable up to the *Blokh-Zyablov* radius. The outer codes are folded Reed-Solomon codes, and the inner codes are picked (via a careful brute-force search, guided by the derandomization of a probabilistic argument) to satisfy a certain "nested" list-decodability property.

Folded codes based on cyclotomic function fields are constructed in [5] with list decoding properties similar to folded Reed-Solomon codes but over an alphabet of size polylogarithmic (instead of polynomial) in the block length. This is useful to give a *Justesen-style* explicit binary concatenated code list-decodable up to the Zyablov radius *without a brute-force search* for a good inner code, by using *all* possible linear codes at the inner level.

4 Concatenated codes can achieve list decoding capacity

Despite achieving some good trade-offs, the above concatenated code constructions fall well short of achieving the list-decoding capacity for binary codes. Given the almost exclusive stronghold of concatenated codes on progress in explicit constructions of list-decodable codes over small alphabets, a natural question that arises is the following: Do there exist binary *concatenated* codes that achieve list-decoding capacity, or does the stringent structural restriction imposed on the code by concatenation preclude such codes achieving list-decoding capacity?

In [7], the authors prove that there *do* exist binary linear concatenated codes that achieve list-decoding capacity for any desired rate. In fact, it is shown that a random concatenated code drawn from a certain ensemble achieves capacity with overwhelming probability. This is somewhat encouraging news for the eventual goal of achieving list-decoding capacity (or at least, going beyond the Blokh-Zyablov radius) for binary codes with polynomial time decodable codes, since code concatenation has been the preeminent method for constructing good codes over small alphabets.

The outer codes in this construction are folded Reed-Solomon codes of rate R_0 over an extension field $\mathbb{F}_{2^m}$ with near-optimal list-recovering properties (these were constructed in [8]). The inner codes for the various positions are random binary linear codes of dimension m and rate r (which can even be chosen to equal 1), with a completely *independent* random choice for each outer codeword position. This gives independence across coordinates of the outer code, which is crucially exploited in the analysis. To prove the desired list decoding property, the goal is to show that a large number of codewords of the concatenated code are unlikely to lie in some Hamming ball of fractional radius $H^{-1}(1 - rR_0) - \varepsilon$.

Using the list recovering properties of folded Reed-Solomon codes, it is shown that for every integer J, any large enough collection (compared to J) of outer codewords has a "good" subset, say $c_1, \ldots, c_J$, of size J with the property that each c_i has at least a fraction $(1 - R_0 - \varepsilon)$ of symbols which are linearly independent (over $\mathbb{F}_2$) of the corresponding symbols of $c_1, c_2, \ldots, c_{i-1}$. Since the inner encoding at each position is a random linear code, each such linearly independent symbol of c_i is mapped to a random binary string that is independent of where the corresponding symbols of $c_1, c_2, \ldots, c_{i-1}$ were mapped. This is to used to upper bound the probability that c_i also falls inside a fixed Hamming ball, even conditioned on $c_1, \ldots, c_{i-1}$ belonging to that ball.

Finally, a union ball over all centers for the Hamming ball and all choices of "good" (in the above sense) J tuples of outer codewords is used to show that with high probability (over the choice of the independent inner encodings) the concatenated code will not have too many codewords in *any* Hamming ball of fractional radius $H^{-1}(1 - rR_0) - \varepsilon$. We refer the reader to [7] for the formal details and the precise probability calculations.

5 List decoding up to $1/2 - o(1)$ radius

In this section, we consider the problem of binary codes that can correct close to the information-theoretically maximum possible fraction 1/2 of errors. Consider the task of communicating k bits of information over a channel that can flip an arbitrary subset of up to a fraction $1/2 - \varepsilon$ of the transmitted bits. Here we think of $\varepsilon \to 0$ as very small, and even allow $\varepsilon = o(1)$ (as k grows). We are interested in the following question: What is the fewest (asymptotic) number $n = n(k, \varepsilon)$ of bits we need to communicate so that no matter which subset of at most $(1/2 - \varepsilon)n$ bits are corrupted, we can recover the k message bits *efficiently* (in time polynomial in k and $1/\varepsilon$)? This question arises in many applications of list decoding in complexity theory and cryptography such as the construction of hardcore predicates from one-way functions, constructions of randomness extractors and pseudorandom generators, approximability of the VC dimension, membership comparability of NP-complete sets, approximating NP-witnesses, etc. We refer the reader to [18] and [4, Chap. 12] for a survey of some of these applications of list decoding.

It can be shown by a random coding argument that there exist codes with $n = O(k/\varepsilon^2)$ such that every Hamming ball of radius $(1/2 - \varepsilon)n$ has at most poly(k/ε) codewords. Further, this bound is tight — for any code with $n < k/\varepsilon^a$ for $a < 2$, there must exist some error pattern for which a super-polynomial number of codewords need to be output. The upper bound of $n = O(k/\varepsilon^2)$ above is non-constructive, and all known explicit constructions achieve weaker bounds on n. In the sequel, we focus on codes that can be constructed in time polynomial in k/ε, as well as list decoded from a fraction $(1/2 - \varepsilon)$ of errors in poly(k/ε) time. (The above mentioned applications in complexity theory demand such efficiency of the construction and the decoding algorithms.) In particular, brute-force search that takes time exponential in $1/\varepsilon$ is not permitted. The construction of codes list-decodable up to the Zyablov radius due to [8] that we discussed in Section 3 achieves an encoding length $n = O(k/\varepsilon^3)$; however, it has construction and decoding complexity exponential in $1/\varepsilon$. Prior to [8], an encoding length of $n = O(k/\varepsilon^4)$ was obtained in [6], but the construction time was once again exponential in $1/\varepsilon$.

We should remark that in the complexity-theoretic applications, the exact dependence of n on k often does not matter as long as the exponent of k is some constant. In the "traditional" setting of coding theory, the most interesting regime is when the exponent of k equals one as this corresponds to constant rate codes (when ε is thought of as a constant). On the other hand, in some complexity theory settings, one would like ε to be as small a function of n as possible. This makes the goal of approaching the optimal $1/\varepsilon^2$ dependence of the block length n on ε important.

The approach to construct such binary codes is again to concatenate an outer algebraic code with good distance/list decoding properties with some special low-rate inner code. 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]. This quartic dependence on ε was the best known bound, till a recent improvement by Guruswami and Rudra [9] who constructed codes with encoding length $n = O\left(\frac{k^3}{\varepsilon^{3+\gamma}}\right)$ for any constant $\gamma > 0$. Their construction was based on concatenation of an outer folded Reed-Solomon code (or their precursor, the Parvaresh-Vardy codes [16]), with a dual BCH code as inner code.

We remark that the Parvaresh-Vardy codes are a generalization of Reed-Solomon codes and the dual BCH codes are generalizations of Hadamard codes. Thus, the above result from [9] generalizes both the outer and inner codes in the Reed-Solomon concatenated with Hadamard construction. This seems necessary for reasons we will sketch shortly.

Consider a Reed-Solomon code of block length $n = 2^m$ over an extension field $\mathbb{F}_{2^m}$ concatenated with a binary Hadamard code of dimension m and block length 2^m. The block length of the concatenated code is thus $N = 2^{2m} = n^2$. Let k' denote the dimension of the Reed-Solomon code, and let $k = k'm$ denote the dimension of the binary concatenated code. If the total fraction of errors w.r.t a codeword is at most $(1/2 - \varepsilon)$, at least $\varepsilon/2$ of the inner Hadamard blocks have at most a fraction $(1/2 - \varepsilon/2)$ of errors. List decoding the Hadamard code up to radius $(1/2 - \varepsilon/2)$ will return a list of at most $1/\varepsilon^2$ candidate field elements for each position of the Reed-Solomon code. At the outer level, it suffices to output all polynomials which agree with one of these candidate symbols for at least

$\varepsilon n/2$ positions. This can be accomplished using Sudan's list decoding algorithm for Reed-Solomon codes [17] provided the rate k'/n of the Reed-Solomon code is at most $\varepsilon^4/16$. For $k' = \Omega(\varepsilon^4 n)$, the block length of the concatenated code satisfies $N = n^2 \leqslant O(k'^2/\varepsilon^8) \leqslant O(k^2/\varepsilon^8)$.

To improve this to the $O(k^2/\varepsilon^4)$ bound attained in [12], one must pass more sophisticated information from the Hadamard decoding stage to the outer Reed-Solomon decoder, and use a *soft* list decoding algorithm for Reed-Solomon codes [11, 15]. Informally, for each inner block i the Hadamard decoder returns every field symbol α as a candidate symbol along with a confidence estimate $w_{i,\alpha}$ which is a (decreasing) linear function of the distance of that inner block to the Hadamard encoding of α. The second moment of these confidence estimates is at most $O(1)$ (this follows from the Parseval's identity for Fourier coefficients, and was the key to the result in [12]). When plugged into the soft decoding bound [11] for Reed-Solomon codes, this implies that the outer decoding succeeds for the larger rate $k'/n = \Omega(\varepsilon^2)$, leading to a block length $N \leqslant O(k^2/\varepsilon^4)$.

The improvement in [9] is based on using the Parvaresh-Vardy codes, which have better list-decoding guarantees, for the outer encoding. For this construction, the Hadamard code is too wasteful to be used at the inner level to encode the PV codeword symbols, since PV codes have a much larger alphabet size than Reed-Solomon codes. Dual BCH 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, and this found use as appropriate confidence estimates to pass to the Reed-Solomon soft decoder. For duals of BCH codes with distance $(2t+1)$, an analogous bound holds only for the $2t$'th moment (this bound also arose in the work of Kaufman and Litsyn [13] 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 obtaining a near-cubic dependence on $1/\varepsilon$ in [9]. We refer the reader to [9] for further details and the precise calculations. We stress that the power of the soft decoding algorithm for the newly discovered algebraic codes [16, 8], and the ability to exploit it via meaningful weights passed from the dual BCH decoder, is crucial for this result.

References

1. P. Elias. Error-correcting codes for list decoding. *IEEE Transactions on Information Theory*, 37:5–12, 1991.
2. O. Goldreich and L. Levin. A hard-core predicate for all one-way functions. In *Proc. of the 21st ACM Symp. on Theory of Computing*, pages 25–32, 1989.
3. P. Gopalan, A. Klivans, and D. Zuckerman. List-decoding Reed-Muller codes over small fields. In *Proc.* 40^{th} *ACM Symposium on Theory of Computing (STOC'08)*, pages 265–274, 2008.

4. V. Guruswami. *List decoding of error-correcting codes.* Number 3282 in Lecture Notes in Computer Science. Springer, 2004.
5. V. Guruswami. Artin automorphisms, cyclotomic function fields, and folded list-decodable codes. arXiv:0811.4139, November 2008.
6. V. Guruswami, J. Håstad, M. Sudan, and D. Zuckerman. Combinatorial bounds for list decoding. *IEEE Transactions on Information Theory*, 48(5):1021–1035, 2002.
7. V. Guruswami and A. Rudra. Concatenated codes can achieve list decoding capacity. In *Proceedings of 19th ACM-SIAM Symposium on Discrete Algorithms*, pages 258–267, January 2008.
8. V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008. Preliminary version in *STOC'06*.
9. V. Guruswami and A. Rudra. Soft decoding, dual BCH codes, and better list-decodable ε-biased codes. In *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity*, pages 163–174, 2008.
10. V. Guruswami and A. Rudra. Better binary list-decodable codes via multilevel concatenation. *IEEE Transactions on Information Theory*, 55(1):19–26, January 2009.
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. 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.
14. T. Kaufman and S. Lovett. The list-decoding size of Reed-Muller codes. arXiv:0811.2356, November 2008.
15. R. Koetter and A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. *IEEE Transactions on Information Theory*, 49(11):2809–2825, 2003.
16. F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *Proc. of the 46th IEEE Symposium on Foundations of Computer Science*, pages 285–294, 2005.
17. M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.
18. M. Sudan. List decoding: Algorithms and applications. *SIGACT News*, 31:16–27, 2000.
19. 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.