Error Correcting Codes: Combinatorics, Algorithms and Applications **(Fall 2007)**

Lecture 13: List Decoding

October 1, 2007

Lecturer: Atri Rudra *Scribe: Thanh-Nhan Nguyen & Atri Rudra*

In previous lectures, we have seen the following bound for unique decoding (for worst-case errors):

$$p \leq \frac{1-R}{2}$$

and the capacity bound for qSC_p (for stochastic errors):

$$p \leq H_q^{-1}(1-R) \sim 1-R \text{ (for large } q\text{).}$$

Note that there is a gap between what we can achieve for worst-case errors and stochastic errors. In this lecture, we extend the notion of unique decoding to give the decoder the flexibility to output a *list* of candidate transmitted codewords. This will allow us to bridge the gap in the Shannon world and the Hamming world.

1 List Decoding

The new notion of decoding that we will discuss is called *list decoding* as the decoder is allowed to output a list of answers. We now formally define (the combinatorial version of) list decoding:

Definition 1.1. *Given $0 \leq \rho \leq 1, L \geq 1$, a code $C \subseteq \Sigma^n$ is (ρ, L)-list decodable if for every received word $\mathbf{y} \in \Sigma^n$,*

$$|\{c \in C | \Delta(\mathbf{y}, c) \leq \rho n\}| \leq L$$

Given an error parameter ρ, a code C and a received word $\mathbf{y}$, a list-decoding algorithm should output all codewords in C that are within (relative) Hamming distance ρ from $\mathbf{y}$. Note that if the fraction of errors that occurred during transmission is at most ρ then the transmitted codeword is *guaranteed* to be in the output list. Further, note that if C is (ρ, L)-list decodable then the algorithm will always output at most L codewords for any received word. In other words, for efficient list-decoding algorithm, L should be a polynomial in the block length n (as otherwise the algorithm will have to output a super-polynomial number of codewords and hence, cannot have a polynomial running time). Thus, the restriction of L being at most some polynomial in n is an *a priori* requirement enforced by the fact that we are interested in efficient polynomial time decoding algorithms. Another reason for insisting on a bound on L is that otherwise the decoding problem can become trivial: for example, one can output all the codewords in the code. Finally, it is worthwhile to note that one can always have an exponential time list-decoding algorithm: go through all the codewords in the code and pick the ones that are within ρ Hamming distance of the received word.

Note that in the communication setup, we need to recover the transmitted message. In such a scenario, outputting a list might not be useful. There are two ways to get around this "problem":

1. Declare a decoding error if list size > 1. Note that this generalizes unique coding (as when the number of errors is at most half the distance of the code then there is a unique codeword and hence, the list size will be at most one). However, the gain over unique decoding would be substantial only if for most error patterns (of weight significantly more than half the distance of the code) the output list size is at most one. Fortunately, it can be show that:

    - For random codes, with high probability, for most received words, the list size is at most one. In other words, for *most* codes, we can hope to see a gain over unique decoding. The proof of this fact follows from Shannon's proof for the capacity for qSC: the details are left as an exercise.
    - The result above is for random codes, which lack structure and it seems difficult to come up with efficient algorithm for such codes. It would be nice, if we could have a result similar to the above for some explicit code. Such results are known and in particular, it was shown by McEliece that a similar result holds for Reed-Solomon codes [2].

    Thus, using this option to deal with multiple answers, we still deal with worse case errors but can correct more error patterns than unique decoding.

2. If decoder has access to some side information, then it can use that to prune the list. In applications of list decoding in complexity theory (see for example [3],[1, Chap. 12]), side information is used crucially.

2 List-Decoding Capacity

Next, we will prove the following result concerning the optimal trade-off between rate of a code and the fraction of errors that can be corrected via list decoding.

Theorem 2.1. *Let $q \geq 2$, $0 \leq p < 1 - \frac{1}{q}$, and $\varepsilon > 0$. Then the following holds for codes of large enough block length n:*

(i) If $R \leq 1 - H_q(p) - \varepsilon$, then there exists a $(p, O(\frac{1}{\varepsilon}))$-list decodable code.

(ii) If $R > 1 - H_q(p) + \varepsilon$, every (ρ, L)-list decodable code has $L \geq q^{\Omega(n)}$.

Thus, the *List-decoding capacity*[1] is $1 - H_q(p)$ (where p is the fraction of errors). Note that this exactly matches capacity for qSC_p and hence, list decoding can be seen as a bridge between Shannon's world and Hamming's world.

We will now give the basic idea behind the proof of part (i) of the theorem. The actual proof will be done in the next lecture.

[1] Actually the phrase should be something like "capacity of worst case noise model under list decoding" as the capacity is a property of the channel. However, in the interest of brevity we will only use the term list-decoding capacity.

As in Shannon's proof for capacity of BSC_p, we will pick a random code and show that it satisfies the required property with non-zero probability. In fact, we will show that a random code is (ρ, L)-list decodable with high probability as long as:

$$R \leq 1 - H_q(p) - \frac{1}{L}$$

The analysis will proceed by proving that probability of a "bad event" is small. "Bad event" means there exist messages $\mathbf{m}_0, \mathbf{m}_1, \cdots, \mathbf{m}_L \in [q]^{Rn}$ and a received code $\mathbf{y} \in [q]^n$ such that:

$$\Delta\left(C(\mathbf{m}_i), \mathbf{y}\right) \leq \rho n, \text{for every } 0 \leq i \leq L$$

The probability of any bad event happening will then be calculated by an application of the union bound.

References

[1] Venkatesan Guruswami. *List decoding of error-correcting codes*. Number 3282 in Lecture Notes in Computer Science. Springer, 2004. (Winning Thesis of the 2002 ACM Doctoral Dissertation Competition).

[2] Robert J. McEliece. On the average list size for the Guruswami-Sudan decoder. In *7th International Symposium on Communications Theory and Applications (ISCTA)*, July 2003.

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