Foreword

This chapter is based on lecture notes from coding theory courses taught by Venkatesan Guruswami at University at Washington and CMU; by Atri Rudra at University at Buffalo, SUNY and by Madhu Sudan at MIT.

This version is dated **May 1, 2013**. For the latest version, please go to

http://www.cse.buffalo.edu/ atri/courses/coding-theory/book/

The material in this chapter is supported in part by the National Science Foundation under CAREER grant CCF-0844796. Any opinions, findings and conclusions or recomendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

Chapter 12

Efficiently Achieving the Capacity of the BSC_p

Table 12.1 summarizes the main results we have seen so far for (binary codes).

	Shannon	Hamming	
		Unique Decoding	List Decoding
Capacity	$1-H(p)$ (Thm 6.3.1)	$\geq$ GV (Thm 4.2.1) $\leq$ MRRW (Sec 8.2)	$1-H(p)$ (Thm 7.4.1)
Explicit Codes	?	Zyablov bound (Thm 9.2.1)	?
Efficient Algorithms	?	$\frac{1}{2}\cdot$ Zyablov bound (Thm 11.3.3)	?

Table 12.1: An overview of the results seen so far

In this chapter, we will tackle the open questions in the first column of Table 12.1. Recall that there exist linear codes of rate $1-H(p)-\varepsilon$ such that decoding error probability is not more than $2^{-\delta n}$, $\delta=\Theta(\varepsilon^2)$ on the BSC_p (Theorem 6.3.1 and Exercise 6.5.1). This led to Question 6.3.1, which asks if we can achieve the BSC_p capacity with explicit codes and efficient decoding algorithms?

12.1 Achieving capacity of BSC_p

We will answer Question 6.3.1 in the affirmative by using concatenated codes. The main intuition in using concatenated codes is the following. As in the case of construction of codes on the Zyablov bound, we will pick the inner code to have the property that we are after: i.e. a code that achieves the BSC_p capacity. (We will again exploit the fact that since the block length of the inner code is small, we can construct such a code in a brute-force manner.) However, unlike the case of the Zyablov bound construction, we do not know of an explicit code that is optimal over say the $q\mathrm{SC}_p$ channel. The main observation here is that the fact that the BSC_p

noise is memory-less can be exploited to pick the outer code that can correct from some small but constant fraction of *worst-case* errors.

Before delving into the details, we present the main ideas. We will use an outer code C_{out} that has rate close to 1 and can correct from some fixed constant (say γ) fraction of worst-case errors. We pick an inner code C_{in} that achieves the BSC_p capacity with parameters as guaranteed by Theorem 6.3.1. Since the outer code has rate almost 1, the concatenated code can be made to have the required rate (since the final rate is the product of the rates of C_{out} and C_{in}). For decoding, we use the natural decoding algorithm for concatenated codes from Algorithm 7. Assume that each of the inner decoders has a decoding error probability of (about) γ. Then the intermediate received word $\mathbf{y}'$ has an expected γ fraction of errors (with respect to the outer codeword of the transmitted message), though we might not have control over where the errors occur. However, we picked C_{out} so that it can correct up to γ fraction of worst-case errors. This shows that everything works in expectation. To make everything work with high probability (i.e. achieve exponentially small overall decoding error probability), we make use of the fact that since the noise in BSC_p is independent, the decoding error probabilities of each of the inner decodings is independent and thus, by the Chernoff bound (Theorem 3.1.6), with all but an exponentially small probability $\mathbf{y}'$ has $\Theta(\gamma)$ fraction of errors, which we correct with the worst-case error decoder for C_{out}. See Figure 12.1 for an illustration of the main ideas. Next, we present the details.

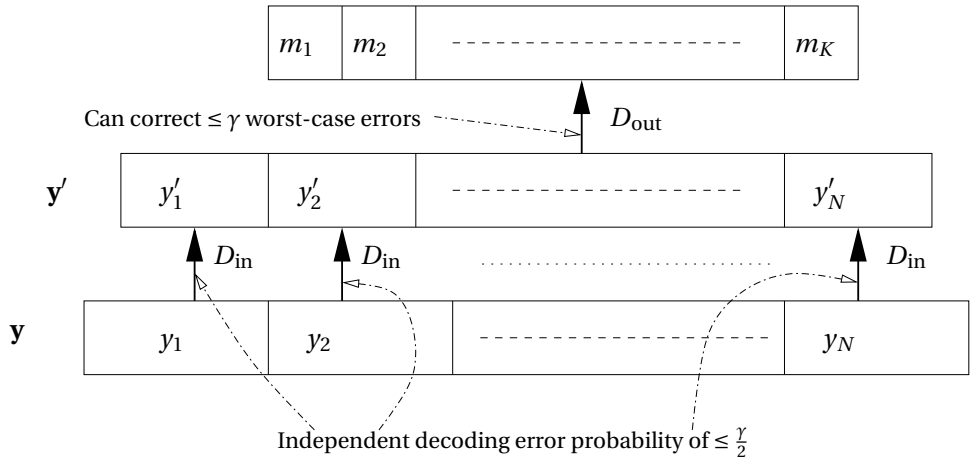

Figure 12.1: Efficiently achieving capacity of BSC_p.

We answer Question 6.3.1 in the affirmative by using a concatenated code $C_{\text{out}} \circ C_{\text{in}}$ with the following properties (where $\gamma > 0$ is a parameter that depends only on ε and will be fixed later on):

(i) C_{out}: The outer code is a linear $[N,K]_{2^k}$ code with rate $R \geq 1 - \frac{\varepsilon}{2}$, where $k = O(\log N)$. Further, the outer code has a unique decoding algorithm D_{out} that can correct at most γ fraction of worst-case errors in time $T_{\text{out}}(N)$.

(ii) C_{in}: The inner code is a linear binary $[n,k]_2$ code with a rate of $r \geq 1 - H(p) - \varepsilon/2$. Further,

there is a decoding algorithm D_{in} (which returns the transmitted codeword) that runs in time $T_{\text{in}}(k)$ and has decoding error probability no more than $\frac{\gamma}{2}$ over BSC_p.

Table 12.2 summarizes the different parameters of C_{out} and C_{in}.

	Dimension	Block length	q	Rate	Decoder	Decoding time	Decoding guarantee
C_{out}	K	N	2^k	$1-\frac{\varepsilon}{2}$	D_{out}	$T_{\text{out}}(N)$	$\leq \gamma$ fraction of worst-case errors
C_{in}	$k \leq O(\log N)$	n	2	$1-H(p)-\frac{\varepsilon}{2}$	D_{in}	$T_{\text{in}}(k)$	$\leq \frac{\gamma}{2}$ decoding error probability over BSC_p

Table 12.2: Summary of properties of C_{out} and C_{in}

Suppose $C^* = C_{\text{out}} \circ C_{\text{in}}$. Then, it is easy to check that

$$R(C^*) = R \cdot r \geq \left(1-\frac{\varepsilon}{2}\right) \cdot \left(1-H(p)-\frac{\varepsilon}{2}\right) \geq 1-H(p)-\varepsilon,$$

as desired.

For the rest of the chapter, we will assume that p is an absolute constant. Note that this implies that $k = \Theta(n)$ and thus, we will use k and n interchangeably in our asymptotic bounds. Finally, we will use $\mathcal{N} = nN$ to denote the block length of C^*.

The decoding algorithm for C^* that we will use is Algorithm 7, which for concreteness we reproduce as Algorithm 11.

Algorithm 11 Decoder for efficiently achieving BSC_p capacity

INPUT: Received word $\mathbf{y} = (y_1, \cdots, y_N) \in [q^n]^N$
OUTPUT: Message $\mathbf{m}' \in [q^k]^K$

1: $\mathbf{y}' \leftarrow (y'_1, \cdots, y'_N) \in [q^k]^N$ where

$$C_{\text{in}}(y'_i) = D_{\text{in}}(y_i) \quad 1 \leq i \leq N.$$

2: $\mathbf{m}' \leftarrow D_{\text{out}}(\mathbf{y}')$
3: RETURN $\mathbf{m}'$

Note that encoding C^* takes time

$$O(N^2k^2) + O(Nkn) \leq O(N^2n^2) = O(\mathcal{N}^2),$$

as both the outer and inner codes are linear[1]. Further, the decoding by Algorithm 11 takes time

$$N \cdot T_{\text{in}}(k) + T_{\text{out}}(N) \leq \text{poly}(N),$$

[1]Note that encoding the outer code takes $O(N^2)$ operations over $\mathbb{F}_{q^k}$. The term $O(N^2k^2)$ then follows from the fact that each operation over $\mathbb{F}_{q^k}$ can be implemented with $O(k^2)$ operations over $\mathbb{F}_q$.

where the inequality is true as long as

$$T_{\text{out}}(N) = N^{O(1)} \text{ and } T_{\text{in}}(k) = 2^{O(k)}. \tag{12.1}$$

Next, we will show that decoding via Algorithm 11 leads to an exponentially small decoding error probability over BSC_p. Further, we will use constructions that we have already seen in this book to instantiate C_{out} and C_{in} with the required properties.

12.2 Decoding Error Probability

We begin by analyzing Algorithm 11.

By the properties of D_{in}, for any fixed i, there is an error at y'_i with probability $\leq \frac{\gamma}{2}$. Each such error is independent, since errors in BSC_p itself are independent by definition. Because of this, and by linearity of expectation, the expected number of errors in $\mathbf{y}'$ is $\leq \frac{\gamma N}{2}$.

Taken together, those two facts allow us to conclude that, by the (multiplicative) Chernoff bound (Theorem 3.1.6), the probability that the total number of errors will be more than γN is at most $e^{-\frac{\gamma N}{6}}$. Since the decoder D_{out} fails only when there are more than γN errors, this is also the final decoding error probability. Expressed in asymptotic terms, the error probability is $2^{-\Omega(\frac{\gamma \cdot N}{n})}$.

12.3 The Inner Code

We find C_{in} with the required properties by an exhaustive search among linear codes of dimension k with block length n that achieve the BSC_p capacity by Shannon's theorem (Theorem 6.3.1). Recall that for such codes with rate $1 - H(p) - \frac{\varepsilon}{2}$, the MLD has a decoding error probability of $2^{-\Theta(\varepsilon^2 n)}$ (Exercise 6.5.1). Thus, if k is at least $\Omega\left(\frac{\log(\frac{1}{\gamma})}{\varepsilon^2}\right)$, Exercise 6.5.1 implies the existence of a linear code with decoding error probability at most $\frac{\gamma}{2}$ (which is what we need). Thus, with the restriction on k from the outer code, we have the following restriction on k:

$$\Omega\left(\frac{\log(\frac{1}{\gamma})}{\varepsilon^2}\right) \leq k \leq O\big(\log N\big).$$

Note, however, that since the proof of Theorem 6.3.1 uses MLD on the inner code and Algorithm 1 is the only known implementation of MLD, we have $T_{\text{in}} = 2^{O(k)}$ (which is what we needed in (12.1). The construction time is even worse. There are $2^{O(kn)}$ generator matrices; for each of these, we must check the error rate for each of 2^k possible transmitted codewords, and for each codeword, computing the decoding error probability requires time $2^{O(n)}$.[2] Thus, the construction time for C_{in} is $2^{O(n^2)}$.

[2] To see why the latter claim is true, note that there are 2^n possible received words and given any one of these received words, one can determine (i) if the MLD produces a decoding error in time $2^{O(k)}$ and (ii) the probability that the received word can be realized, given the transmitted codeword in polynomial time.

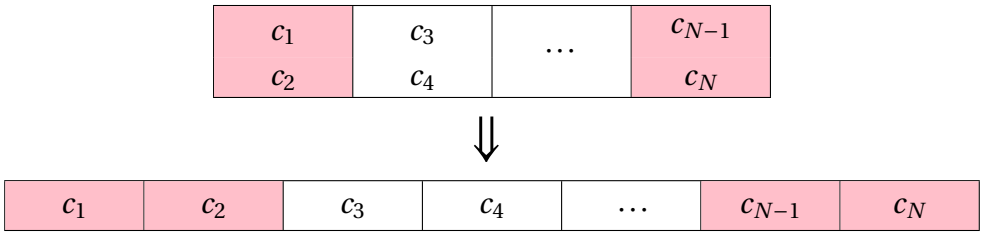

Figure 12.2: Error Correction cannot decrease during "folding." The example has $k = 2$ and a pink cell implies an error.

12.4 The Outer Code

We need an outer code with the required properties. There are several ways to do this.

One option is to set C_{out} to be a Reed-Solomon code over $\mathbb{F}_{2^k}$ with $k = \Theta(\log N)$ and rate $1-\frac{\varepsilon}{2}$. Then the decoding algorithm D_{out}, could be the error decoding algorithm from Theorem 11.2.2. Note that for this D_{out} we can set $\gamma = \frac{\varepsilon}{4}$ and the decoding time is $T_{\text{out}}(N) = O(N^3)$.

Till now everything looks on track. However, the problem is the construction time for C_{in}, which as we saw earlier is $2^{O(n^2)}$. Our choice of n implies that the construction time is $2^{O(\log^2 N)} \leq N^{O(\log N)}$, which of course is not polynomial time. Thus, the trick is to find a C_{out} defined over a smaller alphabet (certainly no larger than $2^{O(\sqrt{\log N})}$). This is what we do next.

12.4.1 Using a binary code as the outer code

The main observation is that we can also use an outer code which is some explicit binary linear code (call it C') that lies on the Zyablov bound and can be corrected from errors up to half its design distance. We have seen that such a code can be constructed in polynomial time (Theorem 11.3.3).

Note that even though C' is a binary code, we can think of C' as a code over $\mathbb{F}_{2^k}$ in the obvious way: every k consecutive bits are considered to be an element in $\mathbb{F}_{2^k}$ (say via a linear map). Note that the rate of the code does not change. Further, any decoder for C' that corrects bit errors can be used to correct errors over $\mathbb{F}_{2^k}$. In particular, if the algorithm can correct β fraction of bit errors, then it can correct that same fraction of errors over $\mathbb{F}_{2^k}$. To see this, think of the received word as $\mathbf{y} \in (\mathbb{F}_{2^k})^{N'/k}$, where N' is the block length of C' (as a binary code), which is at a fractional Hamming distance at most ρ away from $\mathbf{c} \in (\mathbb{F}_{2^k})^{N'/k}$. Here $\mathbf{c}$ is what once gets by "folding" consecutive k bits into one symbol in some codeword $\mathbf{c}' \in C'$. Now consider $\mathbf{y}' \in \mathbb{F}_2^{N'}$, which is just "unfolded" version of $\mathbf{y}$. Now note that each symbol in $\mathbf{y}$ that is in error (w.r.t. $\mathbf{c}$) leads to at most k bit errors in $\mathbf{y}'$ (w.r.t. $\mathbf{c}'$). Thus, in the unfolded version, the total number of errors is at most

$$k \cdot \rho \cdot \frac{N'}{k} = \rho \cdot N'.$$

(See Figure 12.2 for an illustration for $k = 2$.) Thus to decode $\mathbf{y}$, one can just "unfold" $\mathbf{y}$ to $\mathbf{y}'$ and use the decoding algorithm for C' (which can handle ρ fraction of errors) on $\mathbf{y}'$.

We will pick C_{out} to be C' when considered over $\mathbb{F}_{2^k}$, where we choose

$$k = \Theta\left(\frac{\log(\frac{1}{\gamma})}{\varepsilon^2}\right).$$

Further, D_{out} is the GMD decoding algorithm (Algorithm 10) for C'.

Now, to complete the specification of C^*, we relate γ to ε. The Zyablov bound gives $\delta_{\text{out}} = (1-R)H^{-1}(1-r)$, where R and r are the rates of the outer and inners codes for C'. Now we can set $1-R = 2\sqrt{\gamma}$ (which implies that $R = 1-2\sqrt{\gamma}$) and $H^{-1}(1-r) = \sqrt{\gamma}$, which implies that r is[3] $1-O\left(\sqrt{\gamma}\log\frac{1}{\gamma}\right)$. Since we picked D_{out} to be the GMD decoding algorithm, it can correct $\frac{\delta_{\text{out}}}{2} = \gamma$ fraction of errors in polynomial time, as desired.

The overall rate of C_{out} is simply $R \cdot r = \left(1-2\sqrt{\gamma}\right)\cdot\left(1-O\left(\sqrt{\gamma}\log\frac{1}{\gamma}\right)\right)$. This simplifies to $1-O\left(\sqrt{\gamma}\log\left(\frac{1}{\gamma}\right)\right)$. Recall that we need this to be at least $1-\frac{\varepsilon}{2}$. Thus, we would be done here if we could show that ε is $\Omega\left(\sqrt{\gamma}\log\frac{1}{\gamma}\right)$, which would follow by setting

$$\gamma = \varepsilon^3.$$

12.4.2 Wrapping Up

We now recall the construction, encoding and decoding time complexity for our construction of C^*. The construction time for C_{in} is $2^{O(n^2)}$, which substituting for n, is $2^{O\left(\frac{1}{\varepsilon^4}\log^2\left(\frac{1}{\varepsilon}\right)\right)}$. The construction time for C_{out}, meanwhile, is only poly(N). Thus, our overall, construction time is $\text{poly}(\mathcal{N}) + 2^{O\left(\frac{1}{\varepsilon^4}\log^2\left(\frac{1}{\varepsilon}\right)\right)}$.

As we have seen in Section 12.1, the encoding time for this code is $O(\mathcal{N}^2)$, and the decoding time is $N^{O(1)} + N\cdot 2^{O(n)} = \text{poly}(\mathcal{N}) + \mathcal{N}\cdot 2^{O\left(\frac{1}{\varepsilon^2}\log\left(\frac{1}{\varepsilon}\right)\right)}$. We also have shown that the decoding error probability is exponentially small: $2^{-\Omega(\frac{\gamma\cdot\mathcal{N}}{n})} = 2^{-\Omega(\varepsilon^6\mathcal{N})}$. Thus, we have proved the following result:

Theorem 12.4.1. *For every constant p and $0 < \varepsilon < 1-H(p)$, there exists a linear code C^* of block length $\mathcal{N}$ and rate at least $1-H(p)-\varepsilon$, such that*

(a) C^* can be constructed in time $\text{poly}(\mathcal{N}) + 2^{O(\varepsilon^{-5})}$;

(b) C^* can be encoded in time $O(\mathcal{N}^2)$; and

(c) There exists a $\text{poly}(\mathcal{N}) + \mathcal{N}\cdot 2^{O(\varepsilon^{-5})}$ time decoding algorithm that has an error probability of at most $2^{-\Omega(\varepsilon^6\mathcal{N})}$ over the BSC_p.

[3]Note that $r = 1-H(\sqrt{\gamma}) = 1+\sqrt{\gamma}\log\sqrt{\gamma} + (1-\sqrt{\gamma})\log(1-\sqrt{\gamma})$. Noting that $\log(1-\sqrt{\gamma}) = -\sqrt{\gamma}-\Theta(\gamma)$, we can deduce that $r = 1-O(\sqrt{\gamma}\log(1/\gamma))$.

Thus, we have answered in the affirmative Question 6.3.1, which was the central open question from Shannon's work. However, there is a still somewhat unsatisfactory aspect of the result above. In particular, the exponential dependence on $1/\varepsilon$ in the decoding time complexity is not nice. This leads to the following question:

Question 12.4.1. *Can we bring the high dependence on ε down to $\mathrm{poly}\left(\frac{1}{\varepsilon}\right)$ in the decoding time complexity?*

12.5 Discussion and Bibliographic Notes

Forney answered Question 6.3.1 in the affirmative by using concatenated codes. (As was mentioned earlier, this was Forney's motivation for inventing code concatenation: the implication for the rate vs. distance question was studied by Zyablov later on.)

We now discuss Question 12.4.1. For the binary erasure channel, the decoding time complexity can be brought down to $\mathcal{N} \cdot \mathrm{poly}(\frac{1}{\varepsilon})$ using LDPC codes, specifically a class known as Tornado codes developed by Luby et al. [40]. The question for binary symmetric channels, however, is still open. Recently there have been some exciting progress on this front by the construction of the so-called Polar codes.

We conclude by noting an improvement to Theorem 12.4.1. We begin with a theorem due to Spielman:

Theorem 12.5.1 ([51])**.** *For every small enough $\beta > 0$, there exists an explicit C_{out} of rate $\frac{1}{1+\beta}$ and block length N, which can correct $\Omega\left(\frac{\beta^2}{(\log \frac{1}{\beta})^2}\right)$ errors, and has $O(N)$ encoding and decoding.*

Clearly, in terms of time complexity, this is superior to the previous option in Section 12.4.1. Such codes are called "Expander codes." One can essentially do the same calculations as in Section 12.4.1 with $\gamma = \Theta\left(\frac{\varepsilon^2}{\log^2(1/\varepsilon)}\right)$.[4] However, we obtain an encoding and decoding time of $\mathcal{N} \cdot 2^{\mathrm{poly}(\frac{1}{\varepsilon})}$. Thus, even though we obtain an improvement in the time complexities as compared to Theorem 12.4.1, this does not answer Question 12.4.1.

[4] This is because we need $1/(1+\beta) = 1 - \varepsilon/2$, which implies that $\beta = \Theta(\varepsilon)$.