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

Lecture 30: Achieving the BSC_p capacity (II)

Tuesday, November 6, 2007

Lecturer: Atri Rudra *Scribe: Nathan Russell & Atri Rudra*

In the last lecture, we started with the description of our BSC_p capacity achieving code C^*, which is a concatenated code $C_{\text{out}} \circ C_{\text{in}}$, where C_{out} and C_{in} satisfying the following properties:

(i) C_{out}: The outer code with block length N and rate $1 - \frac{\varepsilon}{2}$ over F_{2^k}, with $k = O(\log N)$. Further, the outer code has a unique decoding algorithm D_{out} that can correct at most γ fraction of worst-case errors in time $T_{out}(N)$.

(ii) C_{in}: The inner code has dimension k, dimension n and a rate of $1 - H(p) - \varepsilon/2$. Further, there is a decoding algorithm D_{in} that runs in $T_{in}(k)$ time and has decoding error probability no more than $\frac{\gamma}{2}$ over BSC_p.

In today's lecture, we will analyze the properties of C^* and also see how to get our hands on C_{out} and C_{in} with the desired properties.

For the rest of the lecture, 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^*.

1 Decoding Error Probability

We begin this section by analyzing the natural decoding algorithm that we saw in the last lecture:

Step 1: Let $y'_i = D_{in}(y_i), 1 \leq i \leq N$.

Step 2: Run D_{out} on $\mathbf{y}' = (y'_1, \ldots, y'_N)$.

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 Chernoff bound, 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 decoding error probability. Expressed in asymptotic terms, the error probability is $2^{-\Omega(\frac{\gamma \mathcal{N}}{n})}$.

2 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. Recall that for such codes with rate $1 - H(p) - \frac{\varepsilon}{2}$, the MLD has a decoding error probability of $2^{-\Theta(\varepsilon^2 n)}$. Thus, if k is $\Omega\left(\frac{\log(\frac{1}{\gamma})}{\varepsilon^2}\right)$, Shannon's theorem implies the existence of a linear code with decoding error probability at most $\frac{\gamma}{2}$ (which is what we need).

Note, however, that since Shannon's proof uses MLD on the inner code, the decoding time for this code with dimension k is $2^{O(k)}$. 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 $2^{O(n)}$. (To see why the latter claim is true, note that there are 2^n possible messages and given any one of these messages, 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.) Thus, the overall construction time is $2^{O(n^2)}$.

3 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 $k = \Theta(\log N)$. Then the decoding algorithm for C_{out}, D_{out}, could be the Berlekamp-Welch algorithm. Note that for this D_{out} we can set $\gamma = \frac{\varepsilon}{4}$ and the decoding time is $T_{out}(N) = O(N^3)$.

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

3.1 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. (Recall that C' is the Reed-Solomon code concatenated with a code on the GV bound.) We have seen that such a code can be constructed in polynomial time.

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 $\mathbb{F}_{2^k}$.

We will pick C_{out} to be C' when considered over $\mathbb{F}_{2^k}$, where we choose k to be $\Theta\left(\frac{\log(\frac{1}{\gamma})}{\varepsilon^2}\right)$. Further, D_{out} is the GMD decoding algorithm for C'.

Now, to complete the specification of C^*, we relate γ to ε. The Zyablov bound gives $\delta_{out} = (1-R)H^{-1}(1-r)$, 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[1] $1 - O\left(\sqrt{\gamma}\log\frac{1}{\gamma}\right)$. 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 $R \cdot r = (1 - 2\sqrt{\gamma})(1 - O(\sqrt{\gamma}\log\frac{1}{\gamma}))$. This simplifies to $1 - O(\sqrt{\gamma}\log(\frac{1}{\gamma}))$. 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 $\Omega(\sqrt{\gamma}\log\frac{1}{\gamma})$, which would follow by setting $\gamma = \varepsilon^3$.

3.2 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 $2^{O(\frac{1}{\varepsilon^4}\log^2(\frac{1}{\varepsilon}))}$. The construction time for C_{out}, meanwhile, is only $\text{poly}(N)$. Thus, our overall, construction time is $\text{poly}(\mathcal{N}) + 2^{O(\frac{1}{\varepsilon^4}\log^2(\frac{1}{\varepsilon}))}$.

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

Theorem 3.1. *For every constant p and $0 < \varepsilon < 1 - H(p)$, there exists a 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.

Thus, we have answered in the affirmative 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 1. *Can we bring the high dependence on ε down to $\text{poly}\left(\frac{1}{\varepsilon}\right)$ in the decoding time complexity?*

For the binary erasure channel, the decoding time complexity can be brought down to $\mathcal{N} \cdot \text{poly}(\frac{1}{\varepsilon})$ using LDPC codes, specifically a class known as Tornado codes developed by Luby et al. [1]. The question for binary symmetric channels, however, is still open.

[1] 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 $r = 1 - O(\sqrt{\gamma}\log(1/\gamma))$.

3.3 Using Expander Codes

We begin with a theorem due to Spielman:

Theorem 3.2 ([2])**.** *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 deciding.*

Clearly, in terms of time complexity, this is superior to the previous option in Section 3.1. Such codes are called "Expander codes." One can essentially do the same calculations as in Section 3.1 with $\gamma = \Theta\left(\frac{\varepsilon^2}{\log^2(1/\varepsilon)}\right)$.[2] However, we obtain an encoding and decoding time of $\mathcal{N} \cdot 2^{\text{poly}(\frac{1}{\varepsilon})}$. Thus, even though we obtain an improvement in the time complexities as compared to Theorem 3.1, this does not answer Question 1.

References

[1] Michael Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, and Daniel A. Spielman. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*, 47(2):569–584, 2001.

[2] Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1731, 1996.

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