CSC 5450 Randomness and Computation

Week 4 : Probabilistic Method I

Plan We will use the first moment method to construct good graphs and good codes.

It is a good opportunity to introduce the important concept of entropy.

① Good graphs : expander graphs, superconcentrators, high girth high chromatic number graphs

② Entropy : binomial coefficients, randomness extraction, compression

③ Good codes : Shannon's coding theorem

To Do : ① Please complete the questionnaire (it will become a compulsory question in HW 1).

② Homework 1 will be posted within this week.

Expander Graphs

Given an undirected graph $G=(V,E)$ and a subset $S \subseteq V$, we define $\delta(S)$ as the set of edges with one endpoint in S and another endpoint in $V-S$, and $N(S) := \{ u \in V-S \mid uv \in E \text{ for some } v \in S \}$ be the set of neighbors of S. Informally, we say a subset is <u>expanding</u> if $|\delta(S)|$ is large or $|N(S)|$ is large. There are at least two ways to define expansion:

- Edge expansion: A graph is a c-expander if $|\delta(S)| \geq c \cdot |S|$ for all $S \subseteq V$ with $|S| \leq |V|/2$.
- Vertex expansion: A graph is a c-expander if $|N(S)| \geq c \cdot |S|$ for all $S \subseteq V$ with $|S| \leq |V|/2$.

We want c to be as large as possible, while the graph is <u>sparse</u>. In other words, a graph is a "good" expander if it is a very sparse graph while it is very "well-connected" (i.e. it requires deleting many edges or many vertices to cut out a large subset of vertices). So, we don't consider a complete graph a good expander as it needs too many edges. One can think of expander graphs as very efficient connectors.

Can you think of some good expanders? A hypercube has $O(|V| \log |V|)$ edges and $c \geq 1$.

We are interested in constructing expander graphs with only $O(|V|)$ edges and constant c.

It is a very challenging task to give an explicit construction of such graphs, but it is easy to show that a random sparse graph will be an expander (see L4 of 2011 or MR 5.3).

Expander graphs are useful in many different areas, including derandomization, coding, random walk, constructing efficient networks, proving lower bounds, designing fast algorithms, etc.

We will see some examples later in this course.

Magical Graphs [1, Section 1.2]

In the following we construct a slightly different object, called a magical graph, which can be used to construct another interesting objected called a superconcentrator.

The proof is essentially the same as the existence proof of expanders.

Let $G=(L,R;E)$ be a bipartite graph. We say that G is an (n,m,d)-magical graph if

① $|L|=n$, ② $|R|=m$, ③ every left vertex (vertices in L) has d neighbors, and finally

④ $|N(S)| \geq |S|$ for every $S \subseteq L$ with $|S| \leq |L|/2 = n/2$.

Theorem For every large enough d and n and $m \geq 3n/4$, there exists an (n,m,d)-magical graph.

proof Let G be a random bipartite graph with n vertices on the left and m vertices on the right.

Each left vertex connects to a randomly chosen set of d vertices on the right.

We claim that G is a magical graph with high probability.

Let $S \subseteq L$ with $s := |S| \leq n/2$ and $T \subseteq R$ with $t := |T| < |S|$.

Let $X_{S,T}$ be the indicator variable that all edges from S go to T, and $X = \sum_{S,T} X_{S,T}$.

Then $E[X_{S,T}] = \Pr(X_{S,T}=1) = (t/m)^{sd}$.

Then
$$E[X] = E\left[\sum_{S,T} X_{S,T}\right] = \sum_{S,T} E[X_{S,T}] \leq \sum_{s \leq n/2} \binom{n}{s}\binom{m}{s}\left(\frac{s}{m}\right)^{sd} \quad (\text{as } t \leq s, \text{ and a union bound})$$
$$\leq \sum_{s \leq n/2} \left(\frac{ne}{s}\right)^s \left(\frac{me}{s}\right)^s \left(\frac{s}{m}\right)^{sd} \quad (\text{recall } \binom{n}{k} \leq \left(\frac{ne}{k}\right)^k \text{ from week 2})$$
$$= \sum_{s \leq n/2} \left[\left(\frac{ne}{s}\right)\left(\frac{me}{s}\right)\left(\frac{s}{m}\right)^d\right]^s$$
$$= \sum_{s \leq n/2} \left[O\left(\frac{s}{m}\right)^{d-2}\right]^s$$
$$\leq \sum_{s \leq n/2} \left[c_0 \cdot \left(\frac{2}{3}\right)^{d-2}\right]^s \quad (\text{as } s \leq n/2 \text{ and } m \geq 3n/4)$$
$$\leq \sum_{s \leq n/2} \left(\frac{1}{3}\right)^s \leq \frac{1}{2} \quad (\text{for } d \geq 20 \text{ say}).$$

Since the expected value is less than one and X is an integer value random variable, there exist some outcomes in the sample space that $X=0$, implying that magical graphs exist. □

In fact, if d is a large enough constant say $d \geq 30$, most graphs in the sample space are magical graphs.

Superconcentrator [1, Section 1.3.1]

Definition Let $G = (V, E)$ be a directed graph and let I and O be two subsets of V with n vertices, each called the input and output sets respectively. We say that G is a superconcentrator if for every k and every $S \subseteq I$ and $T \subseteq O$ with $|S| = |T| = k$, there exist k vertex disjoint paths in G from S to T.

We are interested in constructing a superconcentrator with as few edges as possible. A complete bipartite graph from I to O is a superconcentrator, but it requires n^2 edges.

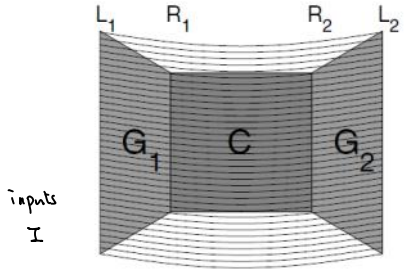
It was conjectured that a superconcentrator with $O(n)$ edges does not exist, but it turned out that one can use magical graphs to construct such a superconcentrator.

No randomness is needed in the following, instead a classical result called Hall's theorem from matching theory will be used.

The construction is recursive. We assume the existence of a superconcentrator C with $3n/4$ inputs, $3n/4$ outputs, and $O(3n/4)$ edges. The base case is when n is a constant, for which a complete bipartite graph would do.

We use two $(n, 3n/4, O(1))$-magical graphs, G_1 and G_2.

Now, a superconcentrator with n inputs and n outputs can be constructed by putting C, G_1, G_2 together:



(picture from [1])

Note that there is a perfect matching between the inputs and the outputs.

First, we show that it is a superconcentrator.

Let $I = O = \{1, 2, \ldots, n\}$ and the matching connects vertex j in I to vertex j in O.

Let $S \subseteq I$ and $T \subseteq O$ with $|S| = |T| = k$. We want to show that there are k vertex disjoint paths between S and T. (In the picture, $S \subseteq L_1$ and $T \subseteq L_2$.)

If $S \cap T \neq \phi$ when think of them as subsets of $\{1,2,\ldots,n\}$, then we can use the edges in the matching to connect those pairs in $S \cap T$.

So, we assume that $S \cap T = \phi$. In particular $|S| = |T| = k \leq n/2$.

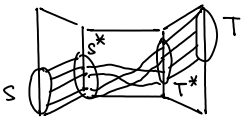
By the property of the magical graph, for any subset $S \subseteq L_1$ (see above picture) with $|S| \leq n/2$, we have $|N(S)| \geq |S|$.

By Hall's theorem, this implies that $S \subseteq L_1$ has a perfect matching to some subset S^* in R_1.

By the same argument, $T \subseteq L_2$ has a perfect matching to some subset T^* in R_2.

Since C is a superconcentrator, there are $|S^*| = |T^*|$ vertex disjoint paths between S^* and T^* in C.

Combining with the two matchings, these form $|S| = |T|$ vertex disjoint paths between S and T.

Pictorially, 

. So, it is a superconcentrator.

Finally, let $E(n)$ be the number of edges in this graph.

Then $E(n) = \underbrace{2 \cdot d \cdot n}_{\text{two magical graphs}} + \underbrace{n}_{\text{matching}} + \underbrace{E(3n/4)}_{\text{small superconcentrator}}$, and solving the recurrence gives $E(n) = O(n)$.

Superconcentrators are efficient communication networks (switching networks).

We will see an application of superconcentrators in network coding later.

Expander graphs can also be used to construct optimal sorting networks [Ajtai, Kolmós, Szemerédi].

High Girth High Chromatic Number Graphs [2]

Graph coloring: the objective is to use the minimum number of colors to color all vertices so that every pair of adjacent vertices receive different colors.

In general we would like to use as few colors as possible, and would like to understand that what graphs require large chromatic number.

One straightforward condition for a graph to have large chromatic number is to have a large clique, but there exist graphs with no triangles (clique of size 3) and yet with large chromatic number.

The following theorem is even more surprising. (see L4 of 2011, page 9)

Theorem For all k, l there exist graphs of chromatic number $> k$ and with no cycles of length $\leq l$.

Theorem For all k, ℓ, there exist graphs of chromatic number $> k$ and with no cycles of length $\leq \ell$.

Proof Consider a random graph where each edge is chosen with probability p.

Let $\chi(G)$ be the chromatic number of G and $\alpha(G)$ be the size of a maximum independent set in G.

Note that $n/\chi(G) \leq \alpha(G)$, as the graph is partitioned into $\chi(G)$ independent sets.

Thus $\chi(G) \geq n/\alpha(G)$. To lower bound $\chi(G)$, we will upper bound $\alpha(G)$ as in week 1.

$$\Pr(\alpha(G) \geq t) \leq \binom{n}{t}(1-p)^{\binom{t}{2}} < n^t e^{-p\binom{t}{2}} = \left(ne^{-p(t-1)/2}\right)^t$$

Set $t = \lceil 3\ln n/p \rceil$, this probability is at most $1/2$ (just a loose bound).

So we know that there are no independent sets of size $\Omega(\ln n/p)$ with good probability.

Next, we bound the number of cycles of length at most ℓ. Call this number X.

Then $E[X] = \sum_{i=3}^{\ell} \binom{n}{i} \frac{i!}{2i} p^i$ ($\binom{n}{i}$ subsets. for each subset, $i!$ permutations, each cycle of length i is counted $2i$ times)

$$\leq \sum_{i=3}^{\ell} \frac{n^i}{2i} p^i \quad \left(\text{recall } \binom{n}{i} \leq \frac{n^i}{i!}\right).$$

Ideally, we would like to choose p so that $E[X] < \frac{1}{2}$ say, and conclude that there are no such cycles. However, to do so, we need $p < \frac{1}{n}$. And then the bound on $\alpha(G)$ would become $O(n \ln n)$, and we could not say anything about $\chi(G)$.

The new idea is to set p larger and do some <u>deterministic modifications</u> later.

Set $p = n^{\varepsilon - 1}$ where $\varepsilon < 1/\ell$.

Then $E[X] \leq \sum_{i=3}^{\ell} \frac{n^{\varepsilon i}}{2i} = o(n)$ as $\varepsilon < 1/\ell$.

In particular, $\Pr(X \geq n/2) < \frac{1}{2}$ by Markov's inequality.

So, with positive probability, the graph has less than $n/2$ "short" (length $\leq \ell < 1/\varepsilon$) cycles and $\alpha(G) < 3n^{1-\varepsilon}\ln n$.

Now, the idea is to delete one vertex in each short cycle to obtain G^*.

Since we need to delete at most $n/2$ vertices, G^* has at least $n/2$ vertices.

Furthermore, it has no short cycles and no independent sets of size $3n^{1-\varepsilon}\ln n$.

Therefore, $\chi(G^*) \geq \frac{|V(G^*)|}{\alpha(G^*)} \geq \frac{n/2}{3n^{1-\varepsilon}\ln n} = \frac{n^{\varepsilon}}{6\ln n}$.

It is larger than any k for sufficiently large n. □

This "deletion method" (more generally "deterministic modification) is a useful technique in

probabilistic method.

Entropy [MU 9.1]

Entropy is an important definition that measures the amount of "information" in a random variable. We will see that it arises naturally in randomness extraction, compression, and error-correcting codes.

Definition $H(X) = -\sum_x \Pr(X=x) \log_2 \Pr(X=x)$.

Equivalently, $H(X) = E[\log_2(1/\Pr(X=x))]$.

For a binary random variable, $H(p) = -p\log_2 p - (1-p)\log_2(1-p)$.

Some basic facts about the entropy function:

- $H(X) \le \log_2 |S|$, where S is the set of outcomes of X.
- $H(X,Y) = H(X|Y) + H(Y) \le H(X) + H(Y)$, and equality achieves when X and Y are independent.

Entropy and Binomial Coefficients [MU 9.2]

The following is a basic relation between entropy and binomial coefficients that will be used later.

Lemma Suppose nq is an integer where $0 < q < 1$. Then $\frac{2^{nH(q)}}{n+1} \le \binom{n}{nq} \le 2^{nH(q)}$.

Proof Upper bound: $\binom{n}{nq} q^{qn} (1-q)^{(1-q)n} \le \sum_{k=0}^{n} \binom{n}{k} q^k (1-q)^{n-k} \le (q + (1-q))^n = 1$.

$$\Rightarrow \binom{n}{nq} \le q^{-qn}(1-q)^{-(1-q)n} = 2^{-q\log_2 q\, n - (1-q)\log_2(1-q)n} = 2^{nH(q)}.$$

Lower bound: We will argue that $\binom{n}{nq} q^{qn}(1-q)^{(1-q)n}$ is the largest term in the binomial expansion, and thus $\binom{n}{nq} q^{qn}(1-q)^{(1-q)n} \ge \frac{1}{n+1}$, and so $\binom{n}{nq} \ge \frac{2^{nH(q)}}{n+1}$.

To see that this is the largest term in the binomial expansion, we just need to compare two consecutive terms $\binom{n}{k} q^k (1-q)^{n-k}$ - $\binom{n}{k+1} q^{k+1} (1-q)^{n-k-1}$

$$= \binom{n}{k} q^k (1-q)^{n-k} \left(1 - \frac{n-k}{k+1} \cdot \frac{q}{1-q}\right)$$

This difference is nonnegative iff $1 - \frac{n-k}{k+1} \cdot \frac{q}{1-q} \ge 0$ iff $k \ge qn - 1 - q$.

Thus $k = qn$ gives the largest term in the expansion. □

Randomness Extraction [MU 9.3]

The goal here is to extract uniform random bits from a random variable.

Definition: Given X, an extraction function Ext should satisfy: $\Pr(\text{Ext}(X)=y \mid |y|=k) = 1/2^k$.

In other words, whenever its output is k bits, all k-bit strings are equally likely.

The proof will consist of two steps:

① If X is a random variable chosen uniformly from $\{0,\dots,m-1\}$, then we can extract $\sim \log_2 m$ bits.

② If we have n coin flips, then we can extract a random variable from $\{0,\dots,m-1\}$ for a large m.

The first step is easier.

Theorem If X is chosen uniformly randomly from $\{0,\dots,m-1\}$, then it is possible to extract $\lfloor \log_2 m \rfloor - 1$ independent and unbiased bits.

Proof (sketch) Let $\alpha = \lfloor \log_2 m \rfloor$.

The best case is when $m = 2^\alpha$, in which we just output the α-bit representation of X.

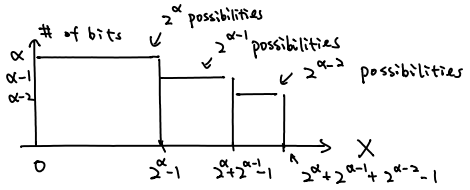
The worst case is when $m = 2^{\alpha+1}-1$. What we will do is the following.

If $X \in \{0,\dots,2^\alpha-1\}$, then output the α-bit representation of X.

Otherwise if $X \in \{2^\alpha,\dots,2^\alpha+(2^{\alpha-1}-1)\}$, then output the $(\alpha-1)$-bit representation of $(X-2^\alpha)$.

Otherwise if $X \in \{2^\alpha+2^{\alpha-1},\dots,2^\alpha+2^{\alpha-1}+(2^{\alpha-2}-1)\}$, then output the $(\alpha-2)$-bit representation of $(X-2^\alpha-2^{\alpha-1})$, and so on.

Pictorically,



It is clear that this will satisfy the requirement of the extraction function.

It can be shown that the expected number of bits output is at least $\lfloor \log_2 m \rfloor - 1$, by a simple induction (see L4 of 2011, or MU for details). □

Now, suppose we have a biased coin with probability $p > 1/2$ being head.

We cannot extract <1 bit from one coin flip.

So, instead, we flip the coins many times and hope to extract $H(p)$ bit per coin flip.

Theorem Given a coin which comes up head with probability $p > \frac{1}{2}$. For any δ and any sufficiently large n,

Theorem Given a coin which comes up head with probability $p > \frac{1}{2}$. For any δ and any sufficiently large n, there exists an extraction function, given n independent coin flips as input, outputs an average of at least $(1-\delta)nH(p)$ random bits.

proof: Algorithm: If there are j heads, map it to a number in $\{0, \ldots, \binom{n}{j}-1\}$.

Then use the previous algorithm to extract $\lfloor \log_2 \binom{n}{j} \rfloor - 1$ random bits.

Clearly, this algorithm satisfies the definition of an extraction function. We analyze its performance.

Let B be the number of bits output by the extraction function.

Let Z be the number of heads in the input.

Then $E[B] = \sum_{k=0}^{n} E[B|Z=k] \Pr(Z=k) \geq \sum_{k=\lfloor n(p-\varepsilon) \rfloor}^{\lceil n(p+\varepsilon) \rceil} E[B|Z=k] \Pr(Z=k)$ where ε is small s.t. $p-\varepsilon > \frac{1}{2}$

$\geq \sum_{k=\lfloor n(p-\varepsilon) \rfloor}^{\lceil n(p+\varepsilon) \rceil} (\lfloor \log_2 \binom{n}{k} \rfloor - 1) \Pr(Z=k)$ by previous theorem

$\geq \left(\log_2 \frac{2^{nH(p+\varepsilon)}}{n+1} - 2 \right) \sum_{k=\lfloor n(p-\varepsilon) \rfloor}^{\lceil n(p+\varepsilon) \rceil} \Pr(Z=k)$ by lemma

The sum is just $\Pr(|Z-np| \leq \varepsilon n)$, which is at least $1 - 2e^{-(np)(\varepsilon/p)^2/3} = 1 - 2e^{-n\varepsilon^2/3p}$ by Chernoff.

Hence, $E[B] \geq \left(nH(p+\varepsilon) - \log_2(n+1) - 2 \right)\left(1 - 2e^{-n\varepsilon^2/3p}\right)$.

By choosing ε small enough, we have $nH(p+\varepsilon) \geq (1-\delta/3)nH(p)$.

By setting n large enough, we have $\left(1 - 2e^{-n\varepsilon^2/3p}\right) \geq 1 - \delta/3$.

So, $E[B] \geq \left((1-\delta/3)nH(p) - \log_2(n+1) - 2 \right)(1-\delta/3)$.

By setting n large enough, $(\delta/3)nH(p) > \log_2(n+1) - 2$, and thus

$E[B] \geq \left(1 - (2\delta/3)nH(p)\right)(1-\delta/3) \geq (1-\delta)nH(p)$. □

We cannot do better than this.

Theorem Under the same setting, the average number of bits output by an extraction function Ext on an input sequence of n independent flips is at most $nH(p)$.

Proof The main observation is: If an input sequence x occurs with probability q, then the corresponding output sequence $\mathrm{Ext}(x)$ can have at most $|\mathrm{Ext}(x)| \leq \log_2(1/q)$ bits.

This is because all sequences with $|\mathrm{Ext}(x)|$ bits would have probability q by the requirement of an extraction function, and thus we must have $2^{|\mathrm{Ext}(x)|} \cdot q \leq 1 \Rightarrow |\mathrm{Ext}(x)| \leq \log_2(1/q)$.

Therefore, $E[B] = \sum_x \Pr(X=x)|\mathrm{Ext}(x)| \leq \sum_x \Pr(X=x) \log_2\left(1/\Pr(X=x)\right) = H(X) = nH(p)$. □

Compression [MU 9.4]

The goal here is to compress the output of a random variable.

We will show that n coin flips can be compressed as roughly $nH(p)$ bits.

Definition : A compression function takes as input a sequence of n coin flips, and outputs a sequence of bits such that each input yields a distinct output.

The following theorem is also known as the source coding theorem.

Theorem Consider a coin with probability $p > 1/2$ head. For any $\delta > 0$ and any sufficiently large n, there exists a compression function such that the expected number of bits is at most $(1+\delta)nH(p)$.

proof Let $\varepsilon > 0$ be a small enough constant with $p - \varepsilon > 1/2$.

Algorithm : When there are less than $n(p-\varepsilon)$ heads, then set the first bit to be one and use the naive scheme (i.e. one bit for each flip).

When there are at least $n(p-\varepsilon)$ heads, then there are at most

$$\sum_{j=\lceil n(p-\varepsilon)\rceil}^{n} \binom{n}{j} \leq \sum_{j=\lceil n(p-\varepsilon)\rceil}^{n} \binom{n}{\lceil n(p-\varepsilon)\rceil} \leq \frac{n}{2}\, 2^{nH(p-\varepsilon)}$$ such sequences (by Lemma).

For each such sequence, assign a unique sequence of at most $nH(p-\varepsilon) + \log_2 n$ bits to represent it, and add a zero to the first bit.

By Chernoff bound, $\Pr(np - X > n\varepsilon) < e^{-n\varepsilon^2/2p}$.

Therefore, the expected number of bits required is at most

$$e^{-n\varepsilon^2/2p}(n+1) + (1 - e^{-n\varepsilon^2/2p})(nH(p-\varepsilon) + \log_2 n + 1).$$

By setting ε small enough and n large enough, similar argument as in above theorem can be used to show that this is at most $(1+\delta)nH(p)$. □

There is an almost matching lower bound.

Theorem Under the same setting, the expected number of bits for any compression function $\geq (1-\delta)nH(p)$.

proof Observation : If a sequence S_1 is more likely than S_2, then an optimal compression function would assign fewer bits to S_1 than to S_2. Since $p > 1/2$, a sequence with more heads should get fewer bits than sequences with fewer heads.

Consider those sequences with $\lfloor n(p+\varepsilon)\rfloor$ heads. There are $\binom{n}{\lfloor n(p+\varepsilon)\rfloor} \geq 2^{nH(p+\varepsilon)}/(n+1)$ of them.

Therefore, one of them must need $\log_2(2^{nH(p+\varepsilon)}/(n+1)) - 1 = nH(p+\varepsilon) - \log_2(n+1) - 1$ bits, as there are not enough shorter strings.

By Chernoff bound, the number of heads is at most $\lfloor n(p+\varepsilon) \rfloor$ with probability $\geq 1 - e^{-n\varepsilon^2/12p}$.

Therefore, by the above observation, the expected number of bits output $\geq (1-e^{-n\varepsilon^2/12p})(nH(p+\varepsilon) - \log_2(n+1) - 1)$.

By setting ε small enough and n large enough, this is at least $(1-\delta)nH(p)$ bits. □

Error Correcting Code [MU 9.5]

Alice wants to send a message to Bob through a noisy channel, where each bit is flipped independently with probability p.

Question: Can they come up with a scheme so that Bob can figure out what Alice wanted to send?

Definition: A (k,n) encoding function maps k bits to n bits.

A (k,n) decoding function maps n bits to k bits.

Shannon solved this problem with matching upper and lower bounds.

This is the founding result of information theory.

Theorem For $p < 1/2$, for any $\delta, \gamma > 0$, for sufficiently large n, for any $k \leq n(1-H(p)-\delta)$, there are (k,n) encoding and decoding functions such that the receiver can obtain the correct message with probability at least $1-\gamma$.

proof First, we prove a weaker result that there exist appropriate coding functions when the input is chosen uniformly at random from all k-bit inputs. Then, we will modify to ensure that the error probability is at most γ on every possible input.

Encoding: For each k-bit string, assign a random n-bit string. Let the codewords be $X_0, X_1, \ldots X_{2^k-1}$.

Decoding: Assume the receiver know the codewords $X_0, \ldots, X_{2^k-1}$. Let the received string be R.

Check each X_i if the number of different bits between X_i and R is between $n(p-\varepsilon)$ and $n(p+\varepsilon)$. If there is a unique such X_i, then output the k-bit string corresponding to X_i as the message sent (i.e. output i).

Error possibilities: ① The original codeword differs from R in less than $n(p-\varepsilon)$ bits or more than $n(p+\varepsilon)$ bits.

② There are more than one codewords differ from R in between

$n(p-\varepsilon)$ and $n(p+\varepsilon)$ bits.

① By Chernoff bound, this is at most $2e^{-\varepsilon^2 n/3p} < \gamma/2$ when n is large enough.

② Suppose X_0 was sent and R is received. Since $X_1, \dots X_{2^k-1}$ are chosen randomly,

$$\begin{aligned}
&\Pr(R \text{ and some } X_i \text{ differ in between } n(p-\varepsilon) \text{ and } n(p+\varepsilon) \text{ bits}) \\
&\le 2^k \sum_{j=n(p-\varepsilon)}^{n(p+\varepsilon)} \binom{n}{j} / 2^n \\
&\le 2^k \cdot n \cdot \binom{n}{n(p+\varepsilon)} / 2^n \\
&\le 2^k \cdot n \cdot 2^{nH(p+\varepsilon)} / 2^n \\
&= n \cdot 2^{n(H(p+\varepsilon) - H(p) - \delta)} \qquad \text{since } k \le n(1-H(p)-\delta).
\end{aligned}$$

By setting ε small enough, $H(p+\varepsilon) - H(p) - \delta$ is negative.

By setting n large enough, this probability is at most $\gamma/2$.

Therefore, the error probability is at most γ when X_0 was sent.

Thus, the total error probability is at most $2^k\gamma$.

There exists a specific $X_0, \dots, X_{2^k-1}$ with total error probability at most $2^k\gamma$.

If each message was sent with probability $1/2^k$, then the average error probability is $\le \gamma$.

Worst case error probability: This is similar to the deletion method we have seen before.

By Markov's inequality, there are at most half the codewords with error probability $> 2\gamma$.

We just remove those codewords. It becomes a $(k-1, n)$ encoding and decoding scheme. □

The above result is best possible.

Theorem Under the same setting, for any $k \ge n(1-H(p)+\delta)$, there are no (k,n) encoding and decoding functions such that the receiver can obtain the correct message with probability at least γ.

proof (idea). The channel error is between $n(p-\varepsilon)$ and $n(p+\varepsilon)$ with high probability.

Consider the simpler setting where the decoding function is always correct in this range.

Then each codeword is associated with

$$\sum_{k=n(p-\varepsilon)}^{n(p+\varepsilon)} \binom{n}{k} \ge \binom{n}{np} \ge 2^{nH(p)}/n+1 \text{ bit sequences.}$$

But there are 2^k codewords.

So, $2^k \cdot \left(\frac{2^{nH(p)}}{n+1}\right) \geq 2^{n(1-H(p)+\delta)} \left(\frac{2^{nH(p)}}{n+1}\right) > 2^n$ for large enough n.

The associated sequences are more than 2^n, a contradiction.

The argument can be modified to allow for small decoding error in this range. □

We note that the above decoding algorithm is inefficient, with running time exponential on the the message size.

Finding a code with efficient encoding and decoding algorithms (ideally linear time algorithms) achieving the rate of the Shannon's coding theorem is still an active research area, fifty years after Shannon's result.

References

[1] Hoory, Linial, Wigderson. Expander graphs and their applications.

[2] Alon, Spencer. The probabilistic method. Second edition, page 38.