Polar Codes

Achieving Optimal Channel Capacity in Erasure Channels

Anav Sood

March 2019

1 Probability Theory Background

We introduce some basic probability concepts. Consider U to be a discrete random variable that has support $\mathcal{U}$. Then, for some $u \in \mathcal{U}$, we notate $\mathbf{P}(U = u)$ as $p(u)$. Although it is ambiguous in general, it will be clear what random variable we are referring to in this document. With this convention, we define the *surprise function*, given by

$$S(u) = -\log p(u)$$

Lower values of $p(u)$ give to higher values of $S(u)$, corresponding to us being surprised to see u as an outcome. From the surprise function, we can define the *entropy* of a random variable

$$H(U) = \mathbf{E}[S(u)] = \sum_{u \in \mathcal{U}} -p(u) \log p(u)$$

and the *conditional entropy* of a random variable given another

$$H(U_1|U_2) = \sum_{u \in \mathcal{U}_2} p(u_2) H(U_1|U_2 = u_2)$$

For clarity, we also give the joint entropy of n random variables (although it follows from the initial definition of entropy)

$$H(U_1, \ldots, U_n) = \sum_{(u_1, \ldots, u_n \in \mathcal{U}_1 \times \cdots \times \mathcal{U}_n)} -p(u_1, \ldots, u_n) \log p(u_1, \ldots, u_n)$$

Using these definitions, we define our final quantity, the mutual information between two random variables

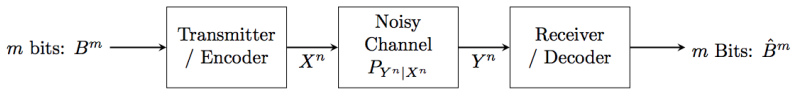
$$I(U_1; U_2) = H(U_1) + H(U_2) - H(U_1, U_2) = H(U_1) - H(U_1|U_2) = H(U_2) - H(U_2|U_1)$$

2 Channels

A channel is an abstract object helps model how we transmit bits from a source to a destination. In general, transmitting information involves the following steps:

1. The m bits that are desired to be sent through the channel as encoded into n bits by an encoder
2. The n encoded bits then travel through the channel, which alters their contents in some probabilistic way
3. The altered n bits are received by a decoder, which decodes them back into m bits

Should the encoding/decoding scheme works well, the m bits decoded by the decoder match the m bits encoded by the encoder. If this is not the case, we deem this round of encoding/decoding an error. This process is displayed below.



In this treatment, we will deal exclusively with *memoryless channels*, and refer to them simply as channels. A memoryless channel is one in which we can treat transmitting an input of size n through the channel as independently transmitting n individual independent inputs of size 1 through the channel.

For a particular channel, a *scheme* refers to the numbers m, n and a particular strategy for encoding and decoding. We call $R = m/n$ bits per channel use the *rate* of the scheme. We say that a particular rate R is achievable for the channel if there exists a sequence of schemes indexed by n all with rate greater than or equal to R, such that as $n \to \infty$ the probability of error goes to 0. We then define the *channel capacity* C to be

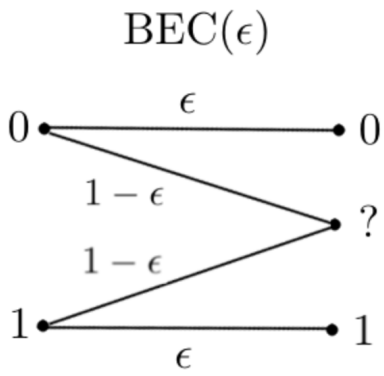
$$C := \sup\{R | R \text{ is achievable}\}$$

From the definitions, this seems like a tough value to compute, but the following theorem makes it quite simple to find in practice.

Theorem. $C = \sup_{p(X)} I(X;Y)$, where the sup is taken over probability distributions of X.

2.1 Binary Erasure Channel (BEC)

We consider a binary erasure channel, with erasure probability ϵ. The channel outputs exactly the input with probability $1 - \epsilon$ and outputs ? with probability ϵ, indicating that the input has been erased. This is displayed pictorally below



We prove the following theoretical result:

Claim. For an ϵ-BEC, the channel capacity C is given by $C = 1 - \epsilon$.

Proof. For a ϵ-BEC we see that

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(X) - H(X|Y = ?)P(Y = ?) - H(X|Y = 0)P(Y = 0) - H(X|Y = 1)P(Y = 1) \\ &= H(X) - H(X)\epsilon \\ &= (1 - \epsilon)H(X) \end{aligned}$$

where in the third step, we have noted that X and $X|Y = ?$ have the same distribution, and X is non-random given that $Y = 0, 1$. We leave it as an exercise to the reader to verify that if $X \in \{0, 1\}$, then $H(X)$ is maximized when $X \sim Ber(\frac{1}{2})$, and it achieves maximal value $H(X) = 1$. Thus we see that

$$C = \sup_{p(X)} (1 - \epsilon)H(X) = 1 - \epsilon$$

as desired.

3 Polar Codes for Erasure Channels

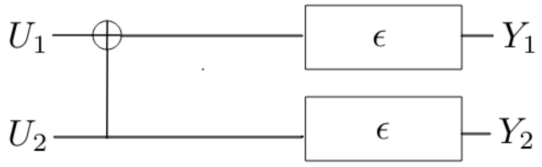
We've proved theoretically that an ϵ-BEC channel has channel capacity $1 - \epsilon$, but we have yet to see how to construct an encoding/decoding scheme that achieves this performance. In this section we introduce polar encoding/decoding, which is a methodology for asymptotically achieving this maximal $1 - \epsilon$ rate. We first give a pictorial depiction of the encoding/decoding scheme, and then analyze its performance.

To do this we first review how the X-OR operation, notated as $\oplus$, acts on bits. It is completely characterized by

$$1 \oplus 1 = 0 \oplus 0 = 0$$
$$1 \oplus 0 = 0 \oplus 1 = 1$$

3.1 The Building Block

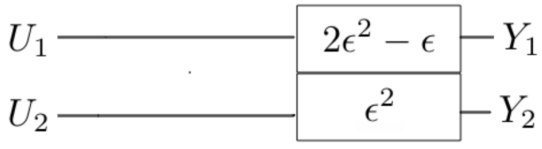
With the X-OR operation under our belt, suppose now that we are attempting to send 2 bits through a memoryless ϵ-BEC channel. Since our channel is memoryless, this is equivalent to sending each bit through an independent ϵ-BEC channel. Call these bits U_1 and U_2. The below diagram displays a methodology for doing so



The boxes represent ϵ-BECs, and our encoding scheme is to send $U_1 \oplus U_2$ into one of the erasure channels, through which we recover Y_1, and U_2 through the other erasure channel, through which we recover Y_2. We will refer to the above diagram as a building block with erasure probability ϵ. We offer the following decoding methodology, called *successive cancellation*, where we decode the bits one at a time.

First, we examine our outputs Y_1 and Y_2, and attempt to decode the value of U_1 without assuming any knowledge regarding U_2. Note that Y_1 or Y_2 (or both) could be missing, as they are the output of an erasure channel. If Y_1 is missing it is clear that we know nothing about U_1, as the value of Y_2 is independent of U_1. If Y_2 is missing, then even with the value of Y_1 we know nothing about U_1 as we need to know U_2 to determine the relationship between Y_1 and U_2. Thus it is clear that we only can recover U_1 in the case that we receive both Y_1 and Y_2. Thus we are effectively putting U_1 through an erasure channel with erasure probability $1 - (1 - \epsilon)^2 = \epsilon^2 - 2\epsilon$.

Now, we attempt to decode the value of U_2, assuming that we have decoded the value of U_1 correctly. In particular, we are assuming that the value of U_1 is known. From this it is obvious that the only way we cannot discern the value of U_2 is if we recieve neither the value of Y_1 nor the value of Y_2. Thus we are effectively putting U_2 through an erasure channel with erasure probability ϵ^2. We can represent this encoding/decoding scheme with the following reduced diagram:

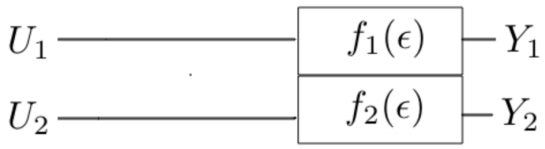


We should keep in mind the nuance that, when finding the value of U_2, we assume we have already correctly identified the value of U_1. Should the reader forget this fact, the above diagram can easily become misleading, but when viewed correctly the diagram is incredibly useful in constructing polar codes. We refer to the above diagram as a reduced building block with erasure probability ϵ. The bit U_1 is considered to be the first entry of the block, and U_2 the second.

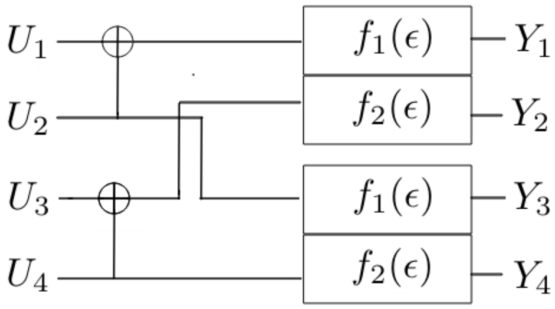
As a quick aside, we note that $2\epsilon - \epsilon \geq \epsilon$ and $\epsilon^2 \leq \epsilon$. We have equality if and only if $\epsilon = 0, 1$, in which case our channels are trivial (they either always erase, or never do). We will assume in general that $\epsilon \in (0, 1)$. We have taken two ϵ-BECs, each of which conveys a single bit with probability $1 - \epsilon$, and have created a scheme in which one bit is conveyed with decreased probability, and the other with increased probability given that the bit with higher probability of erasure is received. If anything, this seems like a counter-intuitive thing to do, but it will become clear why we do so as we build on this result.

3.2 Generalizing to n-Inputs

Now, suppose we wanted to build an encoding scheme generalized to n inputs. We will use the building block given in the previous sub-section to do this inductively for when n is a power of 2. First, we define the functions $f_1(x) = 2x - x^2$ and $f_2(x) = x^2$. Then the reduced building block from above can be written as



With this in mind, we begin outlining our generalized encoding strategy. As a first step, we explicitly describe its implementation for the case of $n = 4$ using the diagram below:

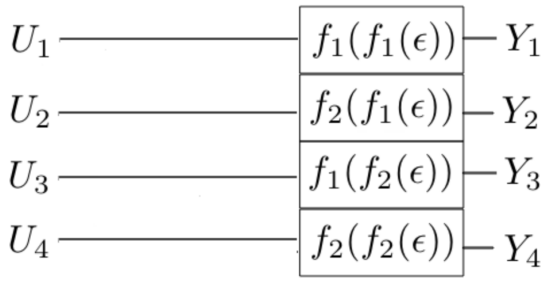


The diagram displays how we encode. The decoding strategy will again be successive cancellation. First, we decode U_1 using the output Y_1, Y_2, Y_3, Y_4 while assuming we know nothing about U_2, U_3, U_4. If examine the diagram above carefully, we see that we are effectively plugging in U_1 as the first entry of one of our building blocks with erasure probability $f(\epsilon)$ with U_2 as the second entry. Thus drawing from the results in the previous section, U_1 is effectively going through an erasure

channel with erasure probability $f_1(f_1(\epsilon)) = 2(2\epsilon - \epsilon^2) - (2\epsilon - \epsilon^2)^2$.

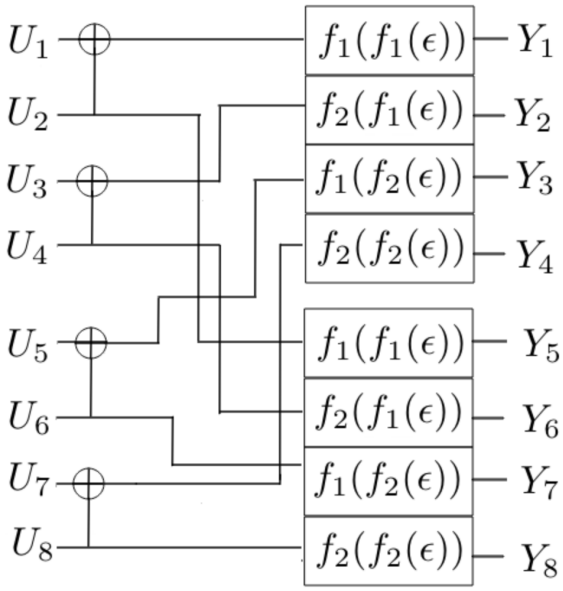
Now we decode U_2 with our output Y_1, Y_2, Y_3, Y_4 assuming we know nothing about U_3, U_4 and that our decoding for U_1 is correct. Again, viewing U_2 as the second entry of a building block with erasure probability $f_1(\epsilon)$ and noting our assumption that we have already correctly decoded U_1, the reasoning in the above sub-section tells us that U_2 is effectively going through an erasure channel with erasure probability $f_2(f_1(\epsilon)) = (2\epsilon - \epsilon^2)^2$.

Continuing this decoding scheme with the same reasoning will tell us that U_3 is effectively going through a erasure channel with erasure probability $f_1(f_2(\epsilon)) = 2\epsilon^2 - (\epsilon^2)^2$, and U_4 is effectively going through an erasure channel with erasure probability $f_2(f_2(\epsilon)) = (\epsilon^2)^2$. With this new understanding, we can visualize our encoding/decoding scheme for the larger $n = 4$ case using the following reduced diagram:



where again, the above diagram implicitly carries the assumption that when we are decoding U_i we have correctly decoded $U_1, \ldots, U_{i-1}$.

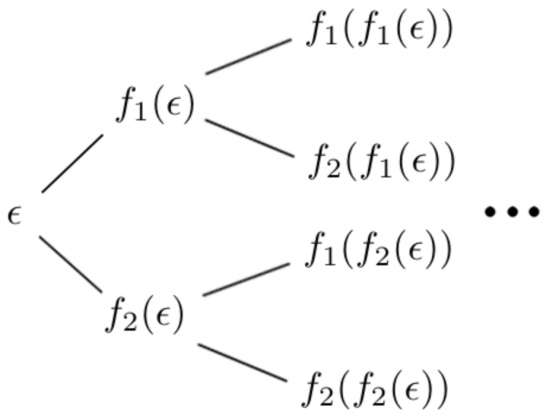
We can extend this methodology to inductively define the scheme for arbitrary n equal to powers of 2. Assume we have a reduced diagram for when $n = m$. Then the following algorithm shows how to construct a diagram for $n = 2m$ given the previous assumption. Take two of the specified reduced diagrams for $n = m$, and label the erasure channels in the first diagram $1, \ldots, m$ and in the second channel $m+1, \ldots, 2n$. For $i = 1, 3, \ldots 2n-1$, make a building block by sending U_i into channel i and U_{i+1} into channel $n + i$. We provide the diagram for $n = 8$, made by this iterative method, below.



The diagram displays how we encode, and we decode using successive cancellation as before.

3.3 How to Achieve Channel Capacity

From the diagrams above, it is clear that for n inputs, where $n = 2^m$ for some m, our encoding scheme is effectively represented by n erasure-channels, with erasure probabilities (loosely) given by the values at the mth depth level of this tree



Note, in actuality, the erasure probability for the ith channel is only correctly characterized by the above tree when we have successfully decoded the first $U_1, \ldots, U_{i-1}$ inputs. We seek to gain a better understanding of these values in the limit as the number of inputs tends to infinity. This

will require a decent background in probability theory.

Consider the following discrete step stochastic process

$$w_t = w_{t-1} + e_t(w_{t-1}(1 - w_{t-1}))$$

where $w_0 \in (0, 1)$ and e_t is a random variable that takes values 1 with probability $\frac{1}{2}$, and -1 with probability $\frac{1}{2}$. Namely

$$w_t = \begin{cases} 2w_{t-1} - w_{t-1}^2 & \text{if } e_t = 1 \\ w_{t-1}^2 & \text{if } e_t = -1 \end{cases}$$

We show that if $w_{t-1} \in (0, 1)$ then $w_t \in (0, 1)$. We can confirm this with case-work on the value of e_t. In the case that $e_t = -1$, the is obvious. Otherwise we note that

$$1 > w_{t-1} \implies 2 > w_{t-1} \implies 2w_{t-1} > w_{t-1}^2 \implies 2w_{t-1} - w_{t-1}^2 > 0$$
$$(w_{t-1} - 1)^2 > 0 \implies w_{t-1}^2 - 2w_{t-1} + 1 > 0 \implies 2w_{t-1} - w_{t-1}^2 < 1$$

From induction we then know that $w_t \in (0, 1)$ for all t. We can also see that

$$\mathbf{E}[w_{t+1}|w_t] = \frac{1}{2}\left(2w_{t-1} - w_{t-1}^2 + w_{t-1}^2\right) = w_{t-1}$$

which tells us that our stochastic process is a martingale (and thus a submartingale). In particular, since the w_t are bounded, we can apply Doob's martingale convergence theorem

Theorem. Let $(X_n, \mathcal{F}_n)$ be a submartingale, which satisfies

$$\sup_n \mathbf{E}X_n^+ < \infty$$

Then $\lim X_n = X_\infty$ exists almost surely.

to see that $\lim w_t = w_\infty$ exists almost surely. It is quite easy to contradict the possibility of having $\lim w_t$ be anything but 0 or 1, so the random variable w_∞ must have all its support on 0 and 1. Again because the w_t are bounded, we can apply dominated convergence theorem to see that

$$w_0 = \mathbf{E}(w_t) = \lim \mathbf{E}(w_t) = \mathbf{E}(\lim w_t) = \mathbf{E}(w_\infty)$$

This combined with the information regarding the support of w_∞ then tells us the distribution of w_∞ is exactly given by $\mathbf{P}(w_\infty = 1) = w_0$, $\mathbf{P}(w_\infty = 0) = 1 - w_0$.

Now, let's relate this back to our original question. Should we set $w_0 = \epsilon$, then w_t is equally likely to be the erasure probability (loosely) of any channel of our encoding/decoding scheme for when we have 2^t inputs. We then see, in the limit as the number of inputs tends to ∞, we end up with ϵ channels with erasure probability 1, and $1 - \epsilon$ channels with erasure probability 0. Namely exactly and ϵ proportion of the channels are purely noise, while $1 - \epsilon$ channels communicate the input bit perfectly, with no noise. This result of having all the channels become either entirely clear or entirely noise is reffered to as *polarization*

Then, rather than actually communicating over all the channels, we purposefully input a *frozen bit* (usually chosen to be 0) the proportion of ϵ channels which are noise, and communicate our actual message over proportion of $(1-\epsilon)$ channels which have no noise. This way we can perfectly decode everything sequentially, because either an element is passed through a no noise channel, or it is passed through an entirely noisy channel, but we are already aware of its value. In the limit, this is perfect communication, where it is clear that the ratio of inputted bits to channel use is $1-\epsilon$. Thus we have achieved channel capacity.

In practice, we are obviously working with a finite number of inputs and channels, so we just drop the ϵ proportion of channels that are most noisy. As our application scales to larger and larger inputs, the probability of error will tend to 0.