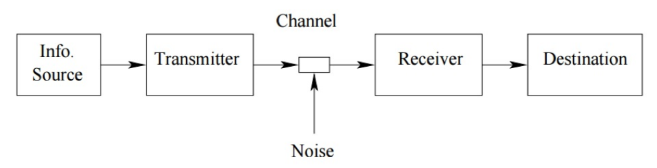
A Communication System



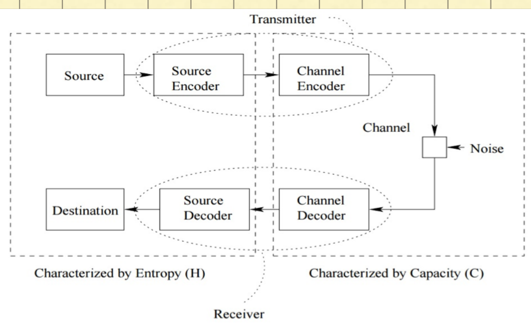
- Info. Source: any source of data we wish to transmit or store
- Transmitter: mapping data source to the channel alphabet in an efficient manner
- Receiver: mapping from channel to data to ensure "reliable" reception
- Destination: data sink

Question: Under what conditions can the output of the source be conveyed *reliably* to the destination? What is reliable? Low prob. of error? Low distortion?

channel 中会有噪音影响.

Receiver 会将编码数据解码.

An Expanded Communication System



What is the ultimate data compression (answer: the entropy H)? What is the ultimate transmission rate of communication (answer: channel capacity C)?

Source Encoder：压缩原始数据

Channel Encoder：增加纠错机制

Channel Decoder：纠错

Source Decoder：还原原始数据

压缩极限是熵(H)

传输率极限是信道容量(C)

Encoders

Source Encoder

- map from source to bits
- "matched" to the information source
- Goal: to get an *efficient* representation of the source (i.e., least number of bits per second, minimum distortion, etc.)

用尽可能少的bit来描述消息

Channel Encoder

- map from bits to channel
- depends on channel available (channel model, bandwidth, noise, distortion, etc.) In communication theory, we work with hypothetical channels which in some way capture the essential features of the physical world.
- Goal: to get *reliable* communication

尽量达到可靠传输

Source Encoders

- Goal: To get an efficient representation (i.e., small number of bits) of the source on average.

Example 1: An urn contains 8 numbered balls. One ball is selected. How many binary symbols are required to represent the outcome?

Outcome	1	2	3	4	5	6	7	8
Representation	000	001	010	011	100	101	110	111

Answer: Require 3 bits to represent any given outcome.

Example 2: Consider a horse race with 8 horses. It was determined that the probability of horse i winning is

$$\Pr[\text{horse } i \text{ wins}] = \left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}\right)$$

Answer 1: Let's try the code of the previous example.

Outcome	Probability	Representation 1
0	$\frac{1}{2}$	000
1	$\frac{1}{4}$	001
2	$\frac{1}{8}$	010
3	$\frac{1}{16}$	011
4	$\frac{1}{64}$	100
5	$\frac{1}{64}$	101
6	$\frac{1}{64}$	110
7	$\frac{1}{64}$	111

To represent a given outcome, the average number of bits is $\bar{\ell} = 3$.

Answer 2: What if we allow the length of each representation to vary amongst the outcomes, e.g., a Huffman code:

若知道概率分布，使用的编码长度可减少.

Outcome	Probability	Representation 2
0	$\frac{1}{2}$	0
1	$\frac{1}{4}$	10
2	$\frac{1}{8}$	110
3	$\frac{1}{16}$	1110
4	$\frac{1}{64}$	111100
5	$\frac{1}{64}$	111101
6	$\frac{1}{64}$	111110
7	$\frac{1}{64}$	111111

The average number of bits is

$$\bar{\ell} = \frac{1}{2} + \frac{1}{4}\cdot 2 + \frac{1}{8}\cdot 3 + \frac{1}{16}\cdot 4 + \frac{4}{64}\cdot 6 = 2$$

Entropy

Definition: The source **entropy**, $H(X)$ of a random variable X with a probability mass function $p(x)$, is defined as

$$H(X) = \sum_x p(x)\log_2 \frac{1}{p(x)}$$

As we will show later in the course, the most effcient representation has average codeword length $\bar{\ell}$ as

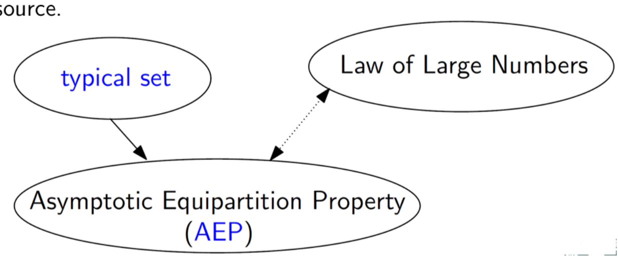
$$H(X) \le \bar{\ell} < H(X) + 1$$

$$\Pr[\text{horse } i \text{ wins}] = \left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}\right)$$

$$H(X) = \frac{1}{2}\log 2 + \frac{1}{4}\log 4 + \frac{1}{8}\log 8 + \frac{1}{16}\log 16 + \frac{4}{64}\log 64 = 2$$

The Huffman code is optimal!

- Information theory and coding deal with the "typical" or expected behavior of the source.
- Entropy is a measure of the *average* uncertainty associated with the source.

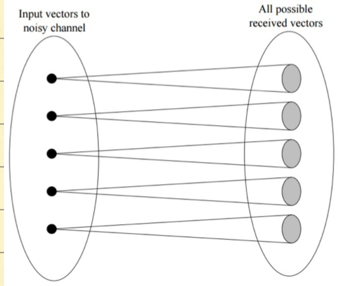


Channel Encoder

- Goal: To achieve an economical (high rate) and reliable (low probability of error) transmission of bits over a channel.

With a channel code we add *redundancy* to the transmitted data sequence which allows for the correction of errors that are introduced by the channel.

增加冗余信息以降低错误率.



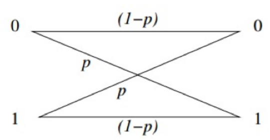
Each transmitted codeword is corrupted by the channel. Each codeword corresponds to a set of possible received vectors.

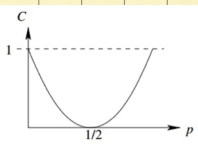
Specify a set of codewords so that at the receiver it is possible to distinguish which element was sent with high-probability.

The *channel coding theorem* tells us the maximum number of such codewords we can define and still maintain completely distinguishable outputs.

Shannon's Channel Coding Theorem There is a quantity called the *capacity*, C, of a channel such that for every rate $R < C$ there exists a sequence of $(\underbrace{2^{nR}}_{\#\text{codewords}}, \underbrace{n}_{\#\text{chan. uses}})$ codes such that $\Pr[error] \to 0$ as $n \to \infty$. Conversely, for any code, if $\Pr[error] \to 0$ as $n \to 1$ then $R \leq C$.

Binary Symmetric Channel





- Input channel alphabet $=$ Output channel alphabet $= \{0, 1\}$
- Assume *independent* channel uses (i.e., memoryless)
- Channel randomly flips the bit *with probability p*
- For $p = 0$ or $p = 1$, $C = 1$ bits/channel use (*noiseless channel* or *inversion channel*)
- Worst case: $p = 1/2$, in which case the input and the output are *statistically independent* ($C = 0$)
- *Question*: How do we *devise codes* which perform well on this channel?

Repetition

- In this code, we repeat one bit *odd times*. The code consists of two possible codewords:

$$\mathcal{C} = \{000\cdots0, 111\cdots1\}$$

- Decoding by a *majority voting* scheme: if there are more 0's than 1's then declare 0, otherwise 1.
- Suppose that $R = 1/3$, i.e., the source output can be encoded before transmission by repeating each bit *three times*.

Example:

10100 ⟶ 111000111000000

↓

11100 ⟵ 101011111001100

The *bit error probability* $\Pr_e$ is:

$$\begin{aligned}\Pr_e &= \Pr[2\text{channel errors}] + \Pr[3\text{channel errors}] \\ &= 3p^2(1-p) + p^3 \\ &= 3p^2 - 2p^3\end{aligned}$$

If $p \leq 1/2$, $\Pr_e$ is less than p. So, the repetition code **improves** the channel's reliability. And for **small** p, the improvement is dramatic.

重复次数是奇数次的.避免奇偶投票出现平局.

For $R = 1/3$, the *bit error probability* Pr_e is:

$$\text{Pr}_e = 3p^2 - 2p^3.$$

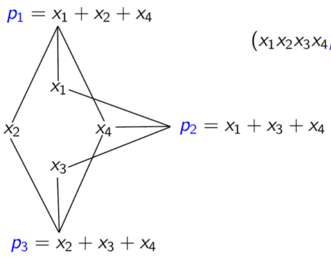
For $R = 1/(2m+1)$, the *bit error probability* Pr_e is:

$$\begin{aligned}
\text{Pr}_e &= \sum_{k=m+1}^{2m+1} \text{Pr}[k \text{ errors out of } 2m+1 \text{ transmitted bits}] \\
&= \sum_{k=m+1}^{2m+1} \binom{2m+1}{k} p^k (1-p)^{2m+1-k} \\
&= \binom{2m+1}{k} p^{m+1} + \text{terms of higher degree in } p.
\end{aligned}$$

Thus, $\text{Pr}_e \to 0$ as $m \to 1$. However, $R \to 0$! Repetition code is NOT efficient! Shannon demonstrated that there exist codes which are *capacity achieving* at non-zero rates.

要想将错误率降为0，则必须让传输率趋于0。

Hamming Code



$(x_1 x_2 x_3 x_4 p_1 p_2 p_3)$

The $(7, 4)$ Hamming code can correct 1 bit error with Rate $R = 4/7$. This code is much better than repetition code.

Hamming codes can be computed in *linear algebra* through *matrices*. This will be explained later in this course.