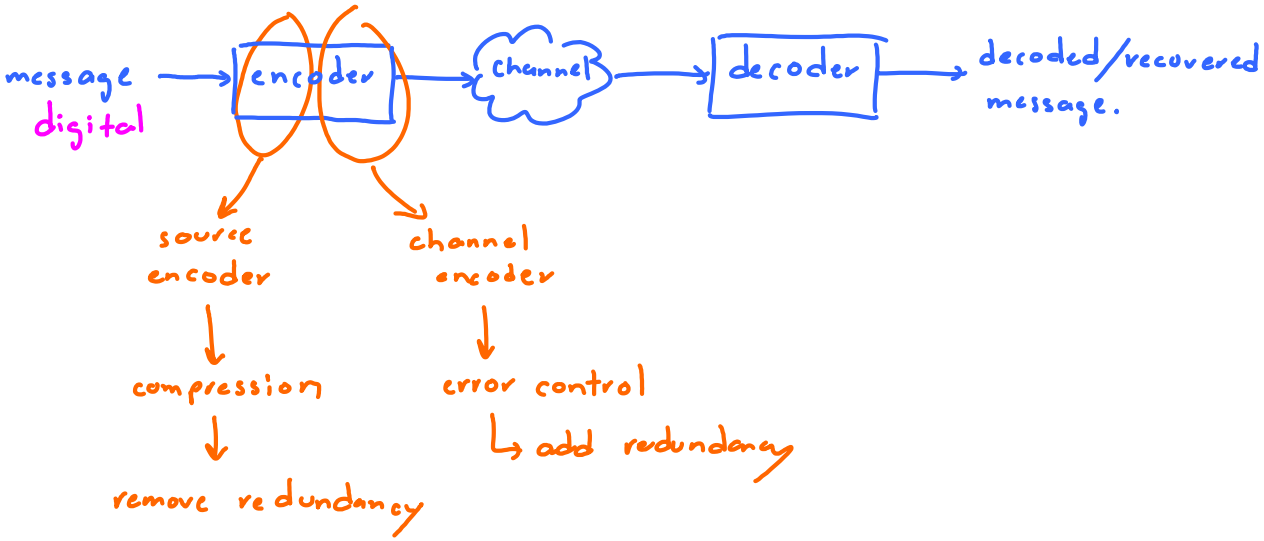
11.1 General Concepts for Source Coding

Friday, September 21, 2012
10:39 AM

11. Source Coding [C&C 16.1]

11.1 General Concepts

What can we do with message in digital format?



This section deals with source coding
in particular, binary source coding

Ex. English Text

Using ASCII, you going to need 7 bits/(source) symbol

Shannon / Cover : we need only ≈ 1.5 bits/symbol on average.

Ex. Suppose the possible messages are

	codewords	relative frequency of occurrence
1) Yes	→ 00	7% 001
2) No	→ 01	3% 01
3) Ok	→ 10	90% 1
4) Thank you	→ 11	3% 0001

$\mathbb{E}[\ell]=2$ (average) expected length of codeword

$\mathbb{E}[\ell]$ need pmf

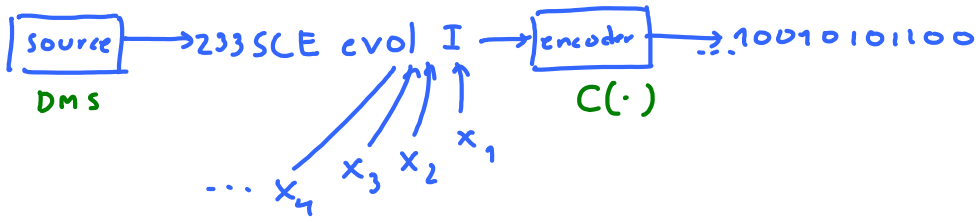
$\mathbb{E}[\ell] = 1\times 9 + 2\times 0.03 + 3\times 0.04 + 4\times 0.03$

$= 0.9 + 0.06 + 0.12 + 0.12$

$= 1.2 < 2$

Ex. Morse code [See slides]

Let's be more specific about information source



For us, we will consider discrete memoryless source (DMS) (digital)

$$p_{X_1 X_2, \ldots, X_n}(x_1, x_2, \ldots, x_n) = p_{X_1}(x_1)\, p_{X_2}(x_2) \cdots p_{X_n}(x_n)$$

Encoder : A function / mapping that takes source symbols (characters) into binary string.

Assumption : $X_1, X_2, \ldots$ are i.i.d. with shared/common pmf $p_X(x)$ support S_X

encoder $C : S_X \longrightarrow \{0,1\}^*$

$= \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, \ldots\}$

(ε : empty string)

Ex. code 1

x	$C(x)$	$\ell(x)$ (length of codeword)
1	00	2
2	01	2
3	10	2
4	11	2

source symbols; codewords; code symbols

If the source produce

1 2 2 1 3

↓ ↓ ↓ ↓ ↓

00 01 01 00 10 ← you have the following binary string to "transmit"

Let's assume $p_X(x) = \begin{cases} 1/2, & x = 1 \\ 1/4, & x = 2 \\ 1/8, & x = 3,4 \\ 0, & \text{otherwise} \end{cases}$

(without this, we can't calculate $\mathbb{E}[\ell(X)]$.)

Expected code length (average) $= \mathbb{E}[\ell(X)] = \sum_x \underbrace{\ell(x)}_{2}\, p_X(x) = 2 \underbrace{\sum_x p_X(x)}_{1} = 2.$

Ex. Code 2

Ex. Code 2

x	$c(x)$	$l(x)$
1	0	1
2	10	2
3	110	3
4	111	3

$$\mathbb{E}[l(X)] = 1\times\frac{1}{2} + 2\times\frac{1}{4} + 3\times\frac{1}{8} + 3\times\frac{1}{8}$$

$$= 1.75.$$

Ex. Code 3

x	$c(x)$	$l(x)$
1	0	1
2	1	1
3	0	1
4	1	1

$$\mathbb{E}[l(X)] = \sum_x l(x)\, p_X(x) = \sum_x 1\; p_X(x) = 1.$$

This code is bad because we have ambiguity at the receiver.

We will focus on "lossless" source coding. Therefore, this ambiguity is not allowed.

Def. A code is **nonsingular** if it is " 1 : 1 "

(if $x_1 \neq x_2$, then $C(x_1) \neq C(x_2)$.)

Obviously, workable code needs to be nonsingular.

This is enough for single transmission (one source symbol)

We usually wish to send a sequence (string) of source symbols.

Concatenation of codewords:

If want to transmit $x_1, x_2, x_3, \ldots$

then, transmit

$$C(x_1)\,C(x_2)\,C(x_3)\ldots$$

Ex Code 4 → Not prefix code because 010 (1 | 3)

x	$C(x)$
1	0
2	010
3	01
4	10

This code is nonsingular.

Suppose the receiver gets

010 ← 2
010 ← 1, 4
010 ← 3, 1

} ambiguity

Solution:

(1) Use fixed-length code.

(2) Use variable-length code.

(2.1) use special symbols btw any two codewords

0 ★ 11 ★ 001 ★ 0111 ★ → inefficient.

(2.2) design code(s) so that the receiver can decode the encoded string (string of codewords) w/o ambiguity.

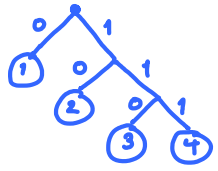
These codes are called uniquely decodable. (UD)

Defn. A code is UD if any encoded string has only one possible source string producing it.

Ex. Code 4 is not UD.

Ex. Code 2 is UD. and prefix(-free)

x	$C(x)$	$\ell(x)$
1	0	1
2	10	2
3	110	3
4	111	3



10 0110 110111100 10

2 1

Defn. A code is called a prefix code (prefix-free code) if no codeword is a prefix of another codeword.

↳ string s_1 is a prefix of string s_2 if $\exists$ string s_3, possibly empty, such that $s_2 = s_1 s_3$ (concatenation)

≡ you can put the codewords into a binary tree where all of them are leaves.

≡ instantaneous code ← because it can be decoded without looking at future codeword(s)

Ex. Code 5

x	$C(x)$

nonsingular: Yes

x	$C(x)$
1	1
2	10
3	100
4	1000

nonsingular : Yes
prefix-free : No
UD : Yes

100 1000 11 10 100 111
3

Problem: Need to look at one bit of the next codeword before you can determine the end of the current codeword.

Two famous examples
- Shannon-Fano Code
- Huffman code ←