Math 5251 Math of Coding: Information, compression, Error-Correction & Finite fields

INTRO Day 1

- Go over syllabus items, text by Garrett (Chapters 2-6, 8-17)
  arrange office hours

What's it about?

Math 5248
Cryptology & number theory

- Cryptography = making codes
- Cryptanalysis = breaking codes

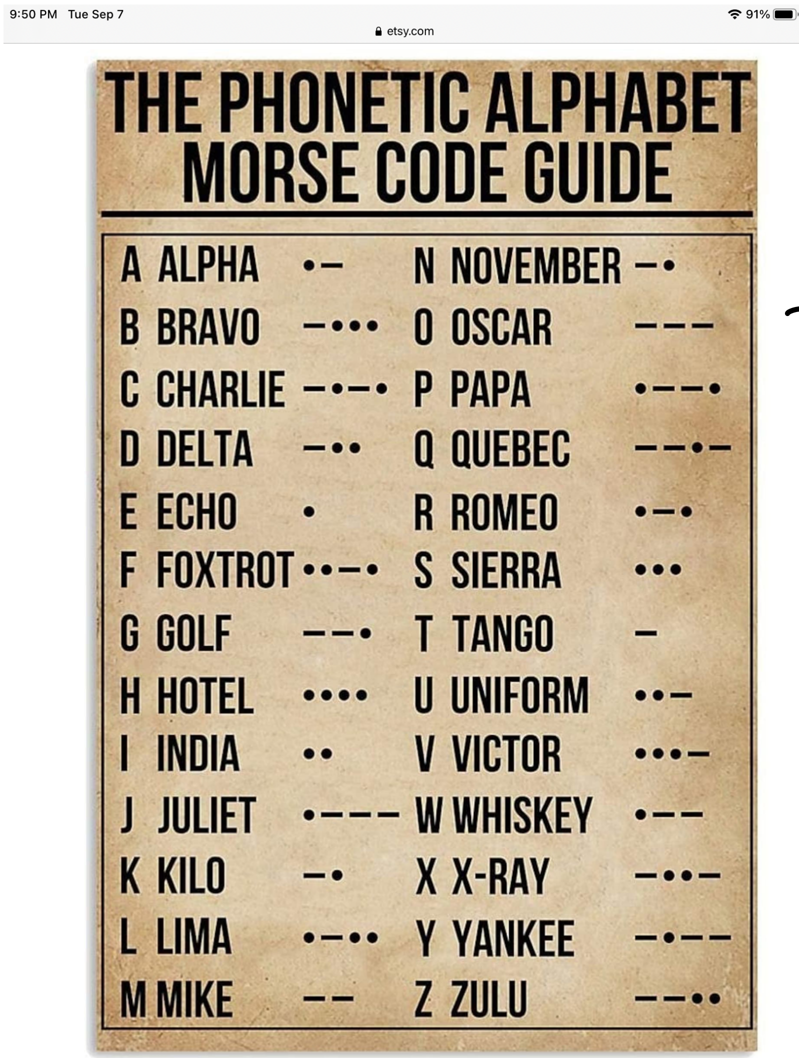
... for secrecy

Math 5251
Coding theory

- Noiseless coding = choosing codewords efficiently, to achieve compression for limited bandwith, avoiding noise
- Noisy coding = choosing codewords with redundancy, to allow error detection and/or error correction

due to noise

EXAMPLE of noiseless coding:



THE PHONETIC ALPHABET
MORSE CODE GUIDE

A	ALPHA	•–	N	NOVEMBER	–•
B	BRAVO	–•••	O	OSCAR	–––
C	CHARLIE	–•–•	P	PAPA	•––•
D	DELTA	–••	Q	QUEBEC	––•–
E	ECHO	•	R	ROMEO	•–•
F	FOXTROT	••–•	S	SIERRA	•••
G	GOLF	––•	T	TANGO	–
H	HOTEL	••••	U	UNIFORM	••–
I	INDIA	••	V	VICTOR	•••–
J	JULIET	•–––	W	WHISKEY	•––
K	KILO	–•–	X	X-RAY	–••–
L	LIMA	•–••	Y	YANKEE	–•––
M	MIKE	––	Z	ZULU	––••

Morse code

Note how letter frequencies affect code word length

e.g. E = "•"

T = "–"

versus

Q = "– – • –"

Z = "– – • •"

We'll see how to optimally (!) design it with the 3 symbols {•, –, space}, introducing the concept of entropy, and Huffman coding (§3.4).

EXAMPLES of noisy coding

(1) (International Radio) Phonetic alphabet

e.g. C = CHARLIE

P = PAPA

T = TANGO ← not short!

achieves error correction (with redundancy) inefficiently

(2) Book ISBN-10 numbers

e.g. Garrett's book is

ISBN-10:	0	1	3	1	0	1	9	6	7	8
multiply by	10	9	8	7	6	5	4	3	2	

sum $0 + 9 + 24 + 7 + 0 + 5 + 36 + 18 + 14 +$

$= 121$ ← always divisible by 11

$\equiv 0 \mod 11$

Detects some errors (but doesn't correct)

(3) QR-codes achieve both
some error-detection and
correction

They use Reed-Solomon codes (§17.3)
(see WSJ article by Eugenia Cheng)

(4) R. Ehrenborg's parlor trick
"Decoding the Hamming code" (see link on syllabus)
uses the binary Hamming [7,4,3] code from § 12.4

On the math & abstraction level:

Like Math 5248,

- early part (noiseless coding) only uses elementary counting, probability, calculus; not so hard
- later part (noisy coding) uses modular arithmetic, particularly $\mathbb{Z}/p\mathbb{Z}$ for p prime as finite fields, constructs all finite fields using polynomials with $\mathbb{Z}/p\mathbb{Z}$ coefficients. Does linear algebra, matrices over finite fields. A bit harder than 5248!

I occasionally ask for proofs on HW & exams, but all easier than ones from lecture or book.

§3.1 Noiseless coding

Start with a finite alphabet of symbols Σ

e.g. $\Sigma = \{ \bullet, -, \text{space} \}$ in Morse code

$\Sigma = \{ A, B, C, \dots, Y, Z \}$ in English

$\Sigma = \{0, 1\}$ for computer applications

binary alphabet

and can form the collection Σ^* of all words in the alphabet Σ

e.g. $\Sigma = \{0,1\}$

has $\Sigma^* = \{0,1\}^*$

$= \{ \phi, 0, 1, 00, 01, 10, 11, 000, 001, \dots \}$

the empty word

Given a finite set W of source words or letters a map $f : W \to \Sigma^*$ is called a **coding** or **encoding** of W using alphabet Σ.

The image of f is a subset $\mathcal{C}$ called the set of **code words**.

EXAMPLES

(1) $W = \{\text{spoken English words}\} \xrightarrow{f = \text{spelling}} \{A, B, C, \ldots, Y, Z\}^* = \Sigma^*$

and $\mathcal{C} = \text{image}(f) = \{\text{written English words}\}$

(2) $W = \{A, B, C, \ldots, Z, 0, 1, \ldots, 9\} \xrightarrow{f = \text{Morse code}} \{\bullet, -, \text{space}\}^*$

Messages come from

$$W^* = \{\text{sequences } (w_1, w_2, \ldots, w_n) \text{ of source words } w_i \in W\}$$

and a message is encoded by concatenating the images under f of each word w_i:

$$W^* \xrightarrow{f^*} \Sigma^*$$

$$f^*(w_1, \ldots, w_n) = f(w_1) f(w_2) \cdots f(w_n)$$

EXAMPLE The map $W = \{A, B, C, D, E\}$ with $\Sigma = \{0,1,2\}$

given by $f \downarrow$

$\Sigma^* = \{0, 1, 20, 21, 22\}$

would encode the source message

$(w_1, w_2, w_3, w_4, w_5, w_6, w_7)$

A C E D E A D

by $\downarrow f^*$

0 20 22 21 22 0 21

DEF'N:

Say the code f is uniquely decipherable if no two distinct messages $(w_1, \ldots, w_n)$ $(w_1', \ldots, w_m')$ get encoded by the same image under f^*, that is $W^* \xrightarrow{f^*} \Sigma^*$ is an injective function.

(Requires $W \xrightarrow{f} \Sigma^*$ injective, but that's not enough)

EXAMPLE

Morse code with a final space at the end of each word is uniquely decipherable, but without the final space it would not be

e.g. $T =$ "–"
$M =$ "– – –" $\Rightarrow$
$O =$ "– –"

$$f^*(\text{TOMTOM}) = f^*(\text{MMMM}) = f^*(\text{TOTTOTTOT})$$

$$= 12 \text{ dashes in a row}$$

Here's one way to avoid the problem...

DEF'N:

Say $f: W \to \Sigma^*$ is a prefix or instantaneous code if no two code words $w \neq w'$ have $f(w)$ a prefix of $f(w')$ of the other.

initial segment, e.g. $f(w) =$ CARD
$f(w') =$ CARDIO

EXAMPLES

(1) Morse code with space at end is prefix (dedicating a new letter to mark a "space" between words always achieves this)

(2) Any code with $f(w)$ all of same length (and $f: W \to \Sigma^*$ injective) is prefix.

(3) The code

$$\begin{aligned} A &\xrightarrow{f} 0 \\ B &\longrightarrow 1 \\ C &\longrightarrow 20 \\ D &\longrightarrow 21 \\ E &\longrightarrow 22 \end{aligned}$$

is prefix.

PROPOSITION
Prefix codes are always uniquely decipherable, instantaneously, that is, without lookahead (or memory requirements)

EXAMPLE Decode/decipher

020222122021 with f as above

$$\Downarrow$$

0|20|22|21|22|0|21
A | C | E | D | E | A | D

work from left

NON-EXAMPLE

If $W = \{A, B, C\} \xrightarrow{f} \Sigma^* \supseteq \{0, 01, 11\}$ then it is uniquely decipherable, but not prefix;

one way to decipher is after given the whole message, one can work backward from the end to decipher it

e.g. 0 0 0 0 1 1 1 0 1 0 0 1

0 | 0 | 0 | 0 1 | 1 1 | 0 1 | 0 | 0 1

A A A B C B A B

work from right

Not instantaneous!

We'll insist on uniquely decipherable codes in this course. It will turn out there is no reason to sacrifice it, unless storage is an issue

– see "lossless" vs. "lossy" compression in Wikipedia