Lecture 2

Prefix-free codes and Kraft's inequality

Announcements

- OH - Pulkit 6-7 pm today (near classroom)
  - Noah 1:30-2:30 pm tomorrow (Packard 318)
- Pre-course survey
- Quiz

Quiz

$8 \rightarrow \log_2 8 = 3$

$16 \rightarrow \log_2 16 = 4$

$\log_2 9 \rightarrow$

1. We want to design fixed length codes for an alphabet of size 9.

   1.1 What is the length in bits/symbol for such a code?

$\lceil \log_2 |\mathcal{X}| \rceil$

$8 < 9 \leq 16$

$\log_2 16 = 4$

Quiz

$X = \{A, B, C, D, \ldots, H, I\}$

Block = AA
Symbol = A

1.2 You observe that there is an overhead above due to 9 not being a power of 2. To partially circumvent this, you decide to encode pairs of symbols together, hence effectively working with an alphabet of size 81. What is the length in bits/symbol for a fixed length code applied on blocks of 2 symbols?

A → 0000
B → 0001
⋮
I →

00000000 ← AA
AB
AC
AD
⋮
II

$64 < 81 \leq 128$
2^6 2^7

$\frac{\lceil \log_2 81 \rceil}{2} = \frac{7}{2} = 3.5$

Quiz

1.3 Now the alphabet size is 81 because we will be encoding pairs of symbols together, so the length in ~~bits/symbol~~ bits/block is $\lceil \log_2(81) \rceil = 7$. However, we are encoding blocks of 2 symbols, so the length in bits/symbol is $\lceil \log_2(81) \rceil / 2 = 3.5$. Did working in blocks of 2 give a better compression ratio?

Yes

$\log_2 9 = 3.17$ bits/symbol

Quiz

2. Given a random sequence $X = x_1, x_2, \ldots, x_n$ sampled from probability distribution $P(A) = 0.5, P(B) = 0.3, P(C) = 0.2$, and the code below, compute the expected code length $E[l(X)]$ in bits/symbol.

Symbol	Codeword
A	`0`
B	`01`
C	`11`

$l(x)$

1

2

2

$E[l(X)]$

$= P(A)\,l(A) + P(B)\,l(B) + P(C)\,l(C)$

$= 0.5 \times 1 + 0.3 \times 2 + 0.2 \times 2$

$= 0.5 + 0.6 + 0.4 = 1.5$ bits/sym.

Let's decode!

Symbol	Probability	Code	$l(x)$
A	0.49	0	1
B	0.49	10	2
C	0.01	110	3
D	0.01	111	3

11111010000111110 -> DCBA- .. - - -

Outline

- **Losslessness: uniquely decodable codes and prefix-free codes**
- Decoding prefix-free codes with trees
- Good prefix-free codes
- Shannon code - a good prefix-free code
- Kraft's inequality - a necessary condition for the existence of a prefix-free code

A	0
B	0

— Not lossless

A	0
B	00

—

AA → 00

B → 00

Not lossless

Uniquely decodable code:

A code is uniquely decodable if no two input sequences (say x^n, y^m where $m, n \geq 1$) are encoded to the same output.

A	0
B	00

→ not uniquely decodable

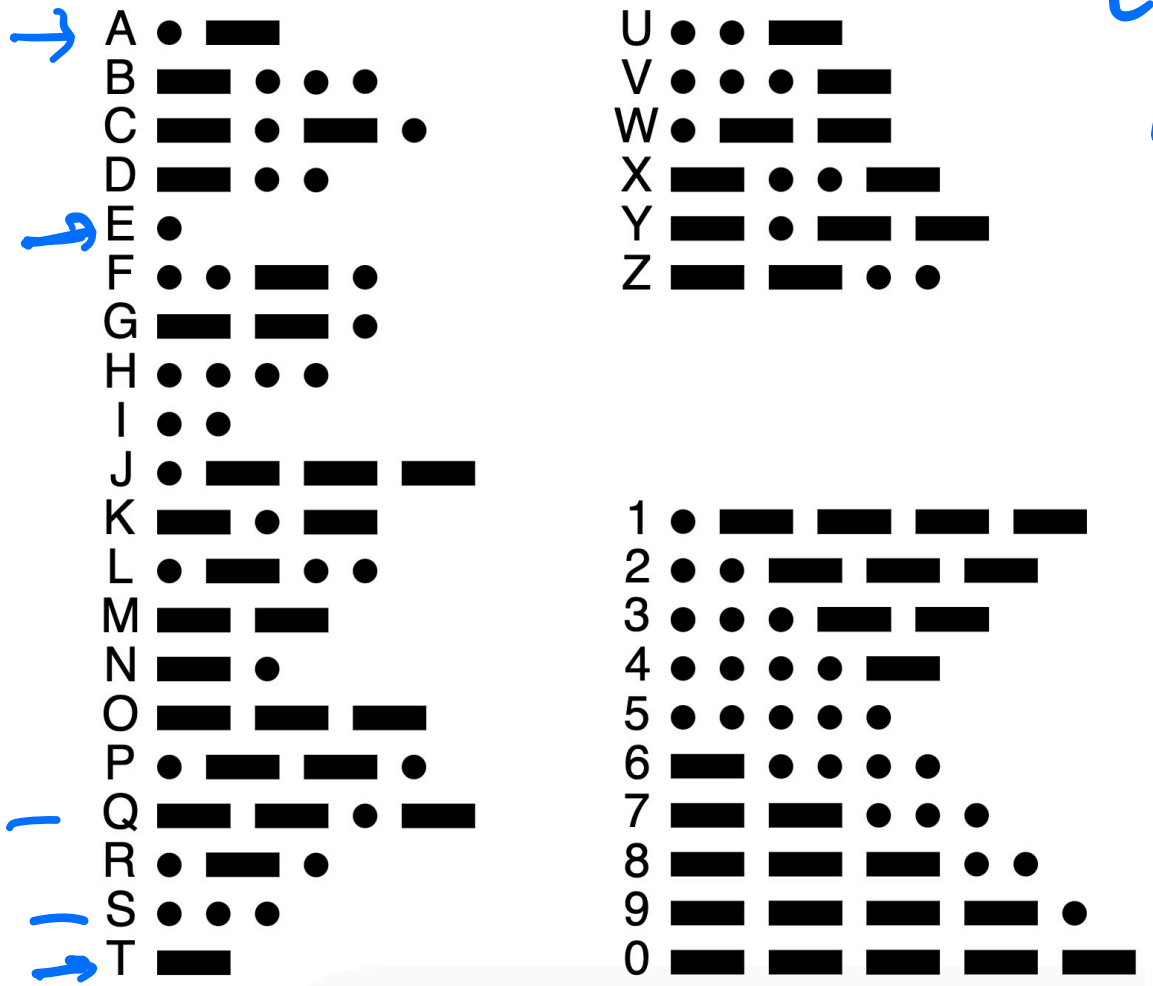
$x^2 = (x_1, x_2) = (A, A)$

$y^1 = (y_1) = (B)$

$Enc(x^2) = Enc(y^1) = 00$

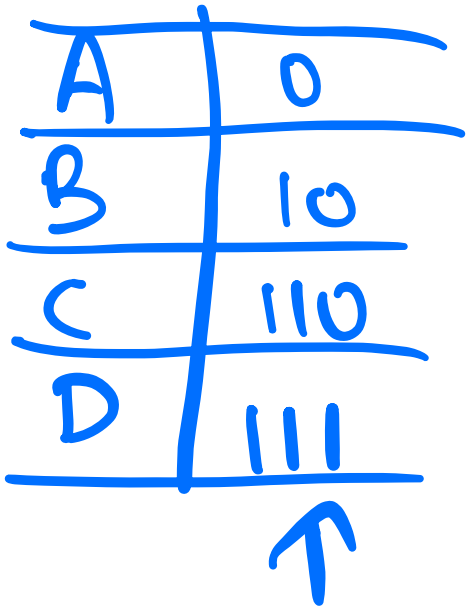
International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.



ET • —

A • —

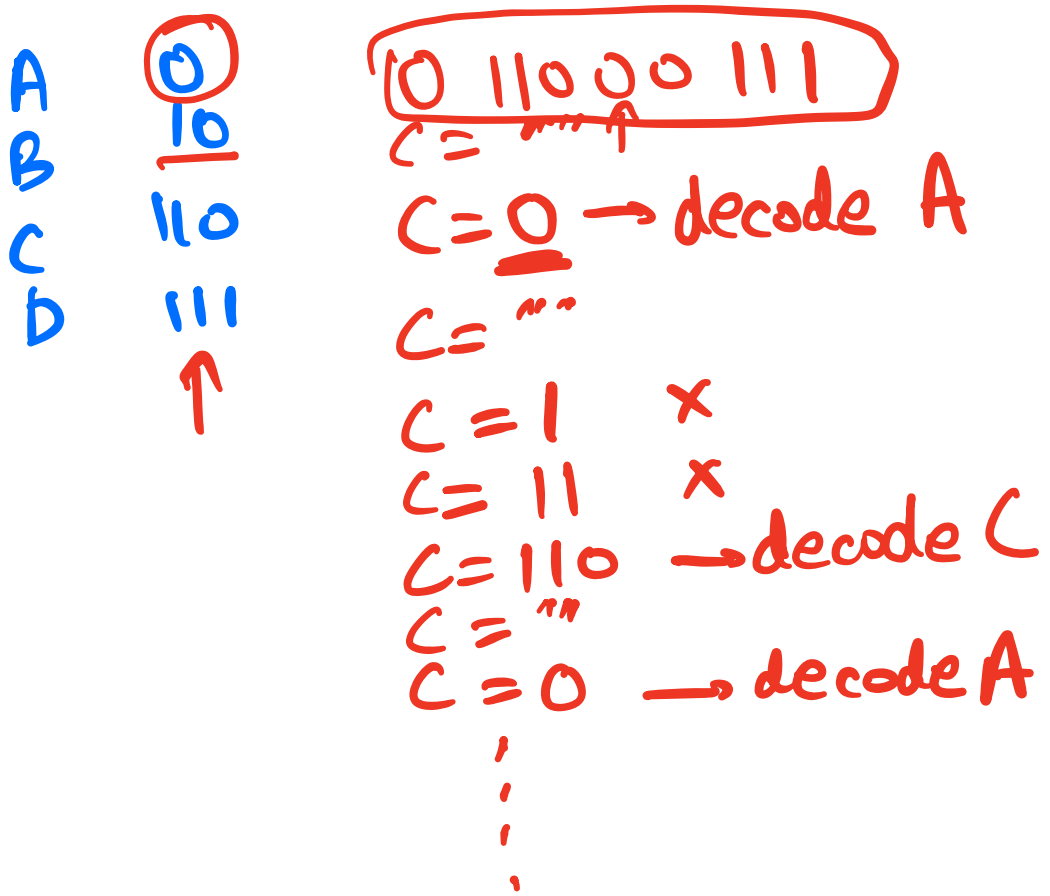


Decoding algorithm

Start with C = ""

Loop:

- add next bit to C
- check for C in table
- If found
  - decode symbol
  - C = ""
- Else:
  - continue



Prefix codes / Prefix-free codes / Instantaneous codes :

No codeword is a prefix of another codeword.

Property

1. Prefix free codes are uniquely decodable.
2. Uniquely decodable $\not\Rightarrow$ Prefix-free

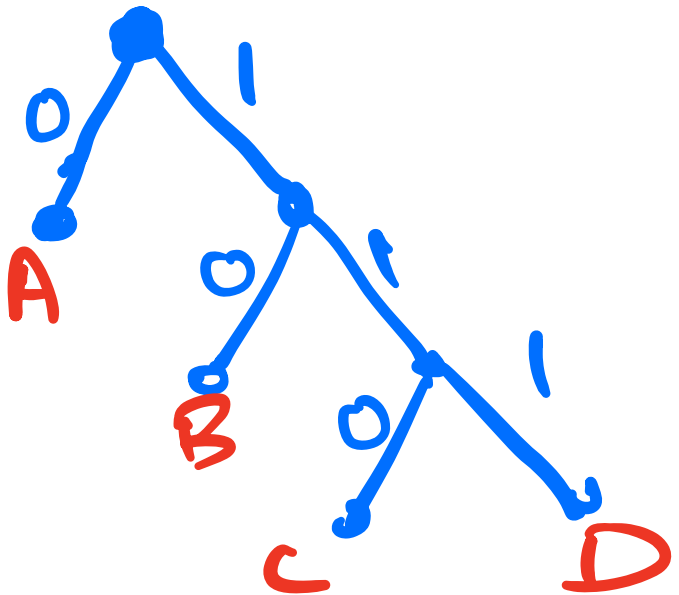
Outline

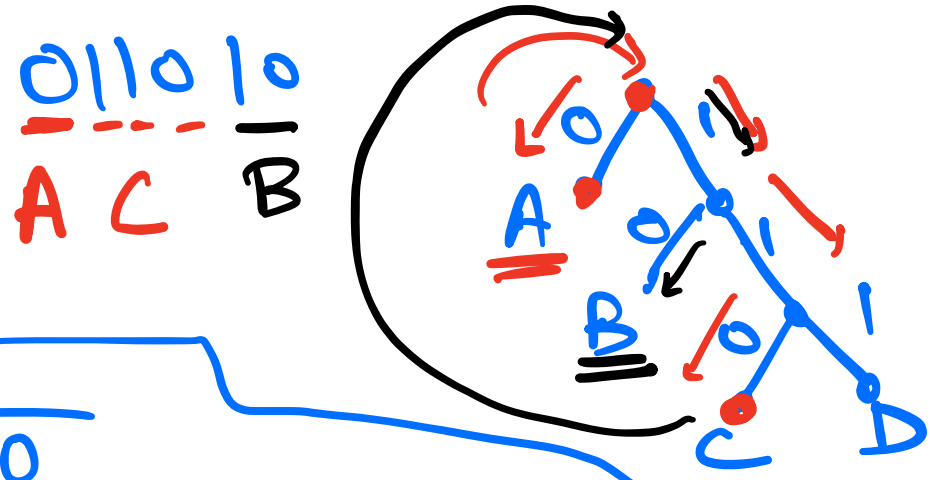
- Losslessness: uniquely decodable codes and prefix-free codes
- **Decoding prefix-free codes with trees**
- Good prefix-free codes
- Shannon code - a good prefix-free code
- Kraft's inequality - a necessary condition for the existence of a prefix-free code

A	0
B	10
C	110
D	111

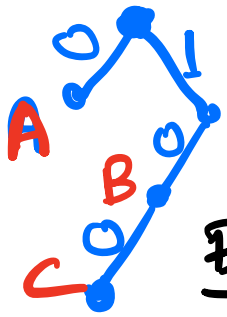
Codewords ⇓ Leaves of a binary tree

Path from root → leaf = codeword





A	0
B	10
C	100



B is not a leaf

B is prefix of C

Prefix code $\Leftrightarrow$ Binary tree where all codewords are leaves

Outline

- Losslessness: uniquely decodable codes and prefix-free codes
- Decoding prefix-free codes with trees
- **Good prefix-free codes**
- Shannon code - a good prefix-free code
- Kraft's inequality - a necessary condition for the existence of a prefix-free code

$\min. \mathbb{E}[\ell(x)]$
$s.t.$ prefix code

Good prefix-free code

1. For symbols s_1, s_2
   if $P(s_1) > P(s_2) \Rightarrow \ell(s_1) \leq \ell(s_2)$

If I have $\ell(s_1) > \ell(s_2)$

→ swap codewords for s_1 & s_2
& minimize $\mathbb{E}[\ell(X)]$

2. [Trust us]

$$\ell(x) \approx \log_2 \frac{1}{P(x)}$$

Example:

1. Uniform distribution

A ½
B ½

$\ell(x) = \log_2 \frac{1}{1/2} = \log_2 2 = 1$

A → 0
B → 1

In general: $x \sim \text{Unif}\{1, \ldots k\}$

Good code: $\ell(x) \approx \log_2 k$

(fixed length code)

$\ell(x) = \lceil \log_2 k \rceil$

2.

	C(x)	P(x)	$\log_2 1/P(x)$
A	0	½	1
B	10	¼	2
C	110	⅛	3
D	111	⅛	3

Outline

- Losslessness: uniquely decodable codes and prefix-free codes
- Decoding prefix-free codes with trees
- Good prefix-free codes
- **Shannon code - a good prefix-free code**
- Kraft's inequality - a necessary condition for the existence of a prefix-free code

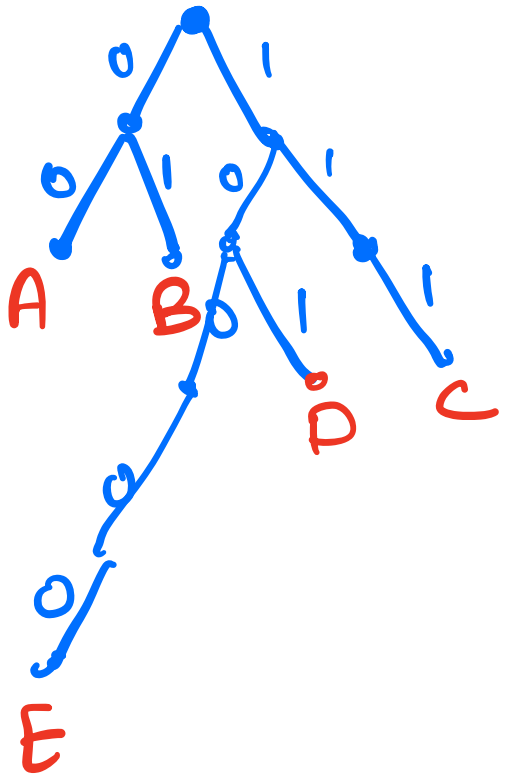
Shannon Code :→

1. Prefix code
2. For symbol x, $l(x) = \lceil \log_2 \frac{1}{p(x)} \rceil$

Construction:

1. Compute $l(x) = \lceil \log_2 \frac{1}{p(x)} \rceil$ for each x
2. Sort symbols in $l(x_1) \leq l(x_2) \ldots$
3. For each symbol x (in the above order)
   - assign leaf of tree at depth $l(x)$ without violating ~~prefix~~ condition.

x	p(x)	l(x)	c(x)
A	0.3	2	00
B	0.3	2	01
C	0.2	3	111
D	0.15	3	101
E	0.05	5	10000



To Prove: A leaf at $\ell(x)$ depth is always available without violating prefix condition.

Proof:

$$\sum_{x \in X} 2^{-\ell(x)} = \sum_{x \in X} 2^{-\lceil \log_2 \frac{1}{P(x)} \rceil}$$

$$\leq \sum_{x \in X} 2^{-\log_2 1/P(x)}$$

$$= \sum_{x \in X} P(x) = 1$$

$$\sum_{x \in X} 2^{-\ell(x)} \leq 1 \quad \text{— (1)}$$

Say we assigned $x_1, \ldots x_m$

Next x_{m+1}

Then $\sum_{i=1}^{m} 2^{-\ell(x_i)} + 2^{-\ell(x_{m+1})} \leq 1$

(based on (1))

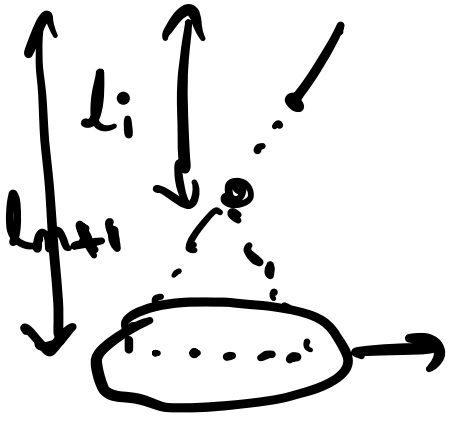
Or: $\sum_{i=1}^{m} 2^{\ell(x_{m+1}) - \ell(x_i)} \leq 2^{\ell(x_{m+1})} - 1$

$$\sum_{i=1}^{m} 2^{\ell(x_{m+1}) - \ell(x_i)} \leq 2^{\ell(x_{m+1})} - 1$$

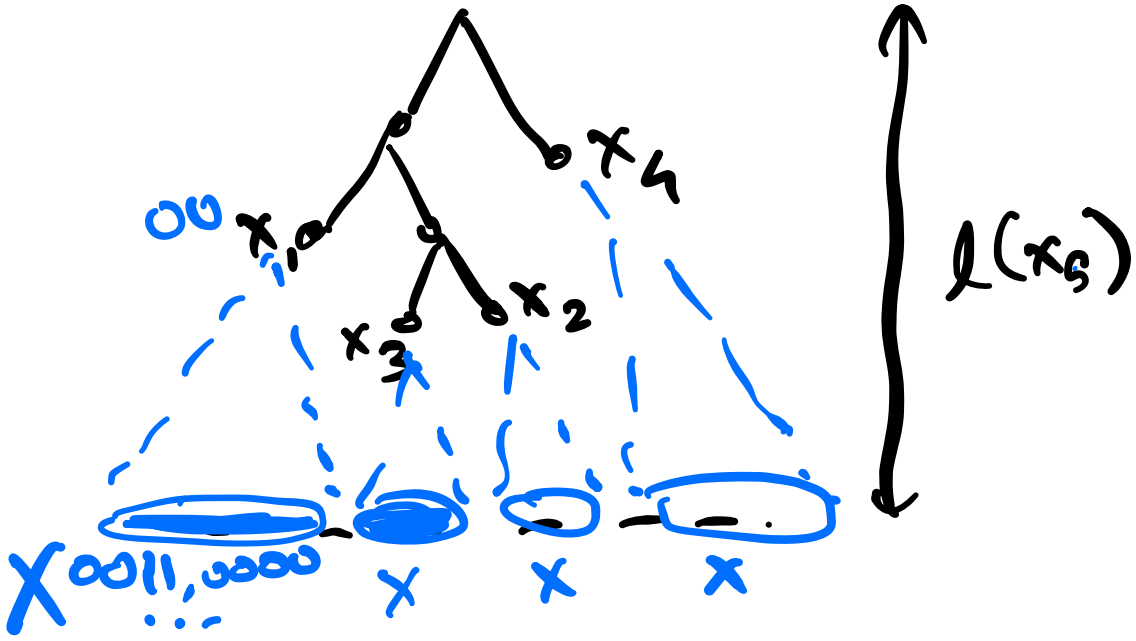
$2^{\ell(x_{m+1}) - \ell(x_i)}$: no. of desc. of $c(x_i)$ at depth $\ell(x_{m+1})$

$2^{\ell(x_{m+1})}$: total no. of leaves at $\ell(x_{m+1})$

-1: ②



No. of leaves at l_{m+1} that use desc. of depth l_i node: $2^{l_{m+1} - l_i}$



$$2^{l(x_{m+1})} - \sum_{i=1}^{m} 2^{l(x_{m+1}) - l(x_i)} \geq 1$$

Total no. of leaves at $l(x_{m+1})$ depth

no. of leaves NOT allowed because they are descended from existing codeword

Outline

- Losslessness: uniquely decodable codes and prefix-free codes
- Decoding prefix-free codes with trees
- Good prefix-free codes
- Shannon code - a good prefix-free code
- **Kraft's inequality - a necessary condition for the existence of a prefix-free code**

Thank you!