INFORMATION THEORY & CODING

Week 5 : Source Coding 1

Dr. Rui Wang

Department of Electrical and Electronic Engineering
Southern Univ. of Science and Technology (SUSTech)

Email: wang.r@sustech.edu.cn

October 17, 2023

Review Summary

Theorem (AEP)

"Almost all events are almost equally surprising." Specifically, if $X_1, X_2, \ldots$ are i.i.d. $\sim p(x)$, then

$$-\frac{1}{n} \log p(X_1, X_2, \ldots, X_n) \to H(X) \text{in probability}.$$

Definition

The *typical set* $A_\epsilon^{(n)}$ is the set of sequences $x_1, x_2, \ldots, x_n$ satisfying

$$2^{-n(H(X)+\epsilon)} \leq p(x_1, x_2, \ldots, x_n) \leq 2^{-n(H(X)-\epsilon)}.$$

Review Summary

Properties of the typical set

1. If $(x_1, x_2, \ldots, x_n) \in A_\epsilon^{(n)}$, then $H(X) - \epsilon \leq -\frac{1}{n} \log p(x_1, x_2, \ldots, x_n) \leq H(X) + \epsilon$.
2. $\Pr[A_\epsilon^{(n)}] > 1 - \epsilon$ for n sufficiently large.
3. $|A_\epsilon^{(n)}| \leq 2^{n(H(X)+\epsilon)}$, where $|A|$ denotes the cardinality of the set A.
4. $|A_\epsilon^{(n)}| \geq (1-\epsilon)2^{n(H(X)-\epsilon)}$ for n sufficiently large.

Theorem

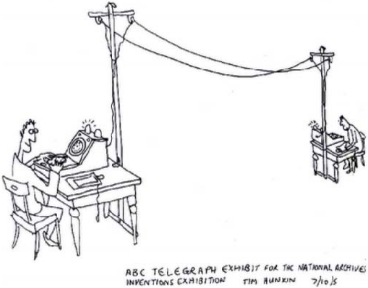
Let X^n be i.i.d. $\sim p(x)$. There exists a code that one-to-one maps sequences x^n of length n into binary strings with

$$E[\frac{1}{n}\ell(X^n)] \leq H(X) + \epsilon$$

for n sufficiently large.

Source Coding

Which horse won in the horse racing?



Which horse won in the horse racing?

X	Pr	Code I	Code II
0	1/2	000	0
1	1/4	001	10
2	1/8	010	110
3	1/16	011	1110
4	1/64	100	111100
5	1/64	101	111101
6	1/64	110	111110
7	1/64	111	111111

$$H(X) = -\sum p_i \log p_i = 2\text{bits}$$

Which code is better?

Source Coding (Data Compression)

- We interpret that $H(X)$ is the best achievable data compression.

- We want to develop practical lossless coding algorithms that approach, or achieve the entropy limit $H(X)$.

Terminology

X	Pr	Code I	Code II
0	$1/2$	000	0
1	$1/4$	001	10
2	$1/8$	010	110
3	$1/16$	011	1110
4	$1/64$	100	111100
5	$1/64$	101	111101
6	$1/64$	110	111110
7	$1/64$	111	111111

- Source alphabet $\mathcal{X} = \{0, 1, 2, 3, 4, 5, 6, 7\}$.
- Code alphabet $\mathcal{D} = \{0, 1\}$.
- Codeword, e.g., 010 for $X = 2$ in Code 1.
- Codeword length, e.g., codeword length for Code 1 is 3.
- Codebook: all the codewords.

Source Coding

Notation (Alphabet Extension)

The set of all possible sequences based on a finite alphabet $\mathcal{D}$ is denoted by $\mathcal{D}^*$. E.g.,
$\mathcal{D} = \{0, 1\} \rightarrowtail \mathcal{D}^* = \{0, 1, 00, 01, 10, 11, 000, \ldots\}$.

Definition (Source Code)

Let $\mathcal{X}$ be the alphabet of a random variable X, and $\mathcal{D}$ be the alphabet of code. A *source code* C for the random variable X is a map

$$
\begin{aligned}
C: \quad & \mathcal{X} \to \mathcal{D}^* \\
& x \mapsto C(x)
\end{aligned}
$$

where $C(x)$ is the codeword associated with x. Let $\ell(x)$ denote the length of C(x).

Source Coding

Definition

The *expected length* $L(X)$ of a source code C for a random variable X with probability mass function $p(x)$ is

$$L(X) = E\ell(X) = \sum_{x \in \mathcal{X}} p(x)\ell(x).$$

X	Pr	Code I	Code II
0	1/2	000	0
1	1/4	001	10
2	1/8	010	110
3	1/16	011	1110
4	1/64	100	111100
5	1/64	101	111101
6	1/64	110	111110
7	1/64	111	111111

$L_1(X) = 3$

$L_2(X) = 2$

Source Coding Applications

- Magnetic recording: cassette, hard drive ...
- Speech compression
- Compact disk (CD)
- Image compression: JPEG

Source Coding: Set of codes

For $\mathcal{X} = \{1, 2, 3, 4\}$ and $\mathcal{D} = \{0, 1\}$, consider

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

- Code efficiency $= H(X)/E[\ell(X)]$
- Which code is best? Would we prefer C_I or C_{II}?

  Consider C_I and decode string: 00001. It would come from $1, 2, 1, 2, 3$ or $2, 1, 2, 1, 3$ or $1, 1, 1, 1, 3$, or etc.

Source Coding: Set of codes

For $\mathcal{X} = \{1, 2, 3, 4\}$ and $\mathcal{D} = \{0, 1\}$, consider

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

- Code efficiency $= H(X)/E[\ell(X)]$
- Which code is best? Would we prefer C_I or C_{II}?

  Consider C_{II} and decode string: 0011. It could be either $1, 1, 2, 2$ or $3, 4$.

Source Coding: Set of codes

For $\mathcal{X} = \{1, 2, 3, 4\}$ and $\mathcal{D} = \{0, 1\}$, consider

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

- Consider C_{III}. Can we decode 1100000000?

  Yes. But if we only see a prefix, such as 11, we don't know until we see more bits to the end.

  $1100000000 = 3, 2, 2, 2, 2$

  $11000000000 = 4, 2, 2, 2, 2$

For $\mathcal{X} = \{1, 2, 3, 4\}$ and $\mathcal{D} = \{0, 1\}$, consider

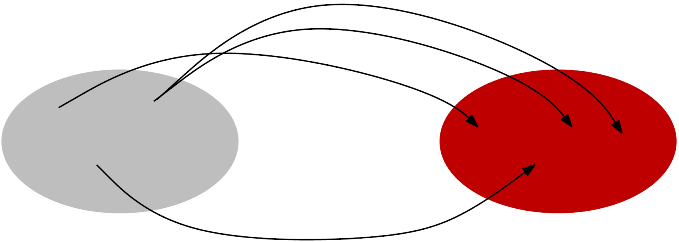
x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

- Consider C_{IV}. This code seems at least feasible (since $E[\ell] \geq H$). Decoding seems easy: (e.g., $111110100 = 111, 110, 10, 0 = 4, 3, 2, 1$).

Definition (Nonsingular Code)

A code C is called *nonsingular* if every realization of $\mathcal{X}$ maps onto a difference codeword in $\mathcal{D}^*$, i.e.,

$$x \neq x' \Rightarrow C(x) \neq C(x').$$



Definition (Nonsingular Code)

A code C is called *nonsingular* if every element of $\mathcal{X}$ maps onto a difference string in $\mathcal{D}^*$, i.e.,

$$x \neq x' \Rightarrow C(x) \neq C(x').$$

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

C_I is singular.

Source Coding: Code types

Definition (Code Extension)

The *extension* of a code C: $\mathcal{X} \rightarrow \mathcal{D}^*$ is defined by

$$C(x_1 x_2 \cdots x_n) = C(x_1)C(x_2)\cdots C(x_n).$$

Definition (Unique Decodable Code)

A code is called *uniquely decodable* if its extension is nonsingular.

$$x_1 x_2 \ldots x_m \neq x_1' x_2' \ldots x_n' \Rightarrow C\left(x_1 x_2 \ldots x_m\right) \neq C\left(x_1' x_2' \ldots x_n'\right)$$

Definition (Unique Decodable Code)

A code is called *uniquely decodable* if its extension is nonsingular.

C_{II}^* is singular. $(C(1,1) = C(3) = 00)$

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

C_I is singular.
C_{II} is NOT u.d..

Source Coding: Code types

Definition (Unique Decodable Code)

A code is called *uniquely decodable* if its extension is nonsingular.

C_{III} is uniquely decodable.

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

C_I is singular.
C_{II} is NOT u.d..

Source Coding: Code types

Definition (Unique Decodable Code)

A code is called *uniquely decodable* if its extension is nonsingular.

$$1100000000 = 3, 2, 2, 2, 2$$
$$11000000000 = 4, 2, 2, 2, 2$$

To know the source, we have to wait until the end!

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

C_I is singular.
C_{II} is NOT u.d..

Source Coding: Code types

Definition (Prefix Code)

A code C is called a *prefix code* (a.k.a. *instantaneous*) iff no codeword of C is a prefix of any other codeword of C.

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

C_I is singular.
C_{II} is NOT u.d..
C_{III} is NOT prefix.
C_{IV} is prefix.

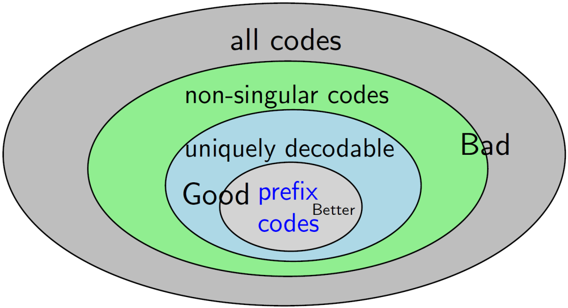
For $\mathcal{X} = \{1, 2, 3, 4\}$ and binary code, consider

x	$p(x)$	C_I	C_{II}	C_{III}	C_{IV}
1	1/2	0	0	10	0
2	1/4	0	1	00	10
3	1/8	1	00	11	110
4	1/8	10	11	110	111
$H(X)$	1.75	–	–	–	–
$E\ell(X)$	–	1.125	1.25	2.125	1.75

- C_I is singular.
- C_{II} is non-singular, but not uniquely decodable.
- C_{III} is non-singular, uniquely decodable, but NOT prefix.
- C_{IV} is non-singular, uniquely decodable, and prefix.

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Source Coding: Classes of codes



- Goal: to find a prefix code with minimum expected length.

Kraft Inequality

Theorem 5.2.1 (Kraft Inequality)

For any prefix code over an alphabet of size D, the codeword lengths $\ell_1, \ell_2, \ldots, \ell_m$ *must satisfy the inequality*

$$\sum_i D^{-\ell_i} \leq 1.$$

Conversely, given a set of codeword lengths that satisfy this inequality, there exists a prefix code with these codeword lengths.

Kraft Inequality

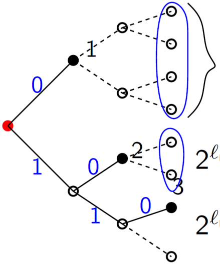
Proof Idea. (A small example) To prove: A prefix code with lengths $\ell_1, \ell_2, \ldots, \ell_m$, the inequality

$$\sum_i D^{-\ell_i} \leq 1 \qquad \text{holds.}$$

Depth: 0 1 2 3

x	$c(x)$
1	0
2	10
3	110

$\ell_{\max} = 3$



$2^{\ell_{\max}-\ell_i} = 2^{3-1} = 4$

$2^{\ell_{\max}-\ell_j} = 2^{3-2} = 2$

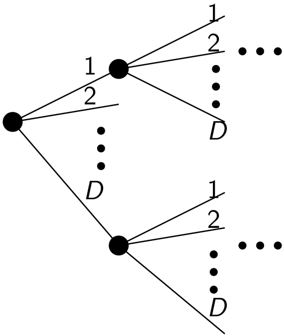
$2^{\ell_{\max}-\ell_{\max}} = 2^{3-3} = 1$

$$\sum_i 2^{-\ell_i} \leq 1 \Leftarrow \sum_i 2^{\ell_{\max}-\ell_i} \leq 2^{\ell_{\max}}$$

Kraft Inequality

Proof. (in general)

- Represent the set of prefix codes on a D-ary tree:

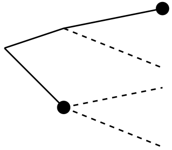


- Codewords correspond to leaves
- Path from root to each leaf determines a codeword
- Prefix condition: won't get to a codeword until we get to a leaf (no descendants of codewords are codewords)

Kraft Inequality

Proof. (in general)

- $\ell_{\max} = \max_i(\ell_i)$ is the length of the longest codeword.
- We can expand the full-tree down to depth $\ell_{\max}$:



The nodes at the level $\ell_{\max}$ are either

1. codewords
2. descendants of codewords
3. neither

- Consider a codeword i at depth ℓ_i in tree
- There are $D^{\ell_{\max}-\ell_i}$ descendants in the tree at depth $\ell_{\max}$
- Descendants of code i are disjoint from decedents of code j (prefix free condition)

Proof. (in general)

- All the above implies:

$$\sum_i D^{\ell_{\max}-\ell_i} \leq D^{\ell_{\max}} \quad \Rightarrow \sum_i D^{-\ell_i} \leq 1$$

- Conversely: given codewords lengths $\ell_1, \ell_2, \ldots, \ell_m$ satisfying Kraft inequality, try to construct a prefix code.

Proof. (in general)

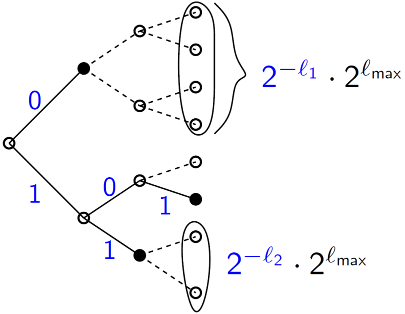
- Conversely: given codewords lengths $\ell_1, \ell_2, \ldots, \ell_m$ satisfying Kraft inequality, try to construct a prefix code.

$$\{\ell_1, \ell_2, \ell_3\} = \{1, 2, 3\}$$

$$2^{-1} + 2^{-2} + 2^{-3} \leq 1$$

x	$c(x)$
1	0
2	11
3	101

C is prefix.



Proof. (in general)

- Conversely: given codewords lengths $\ell_1, \ell_2, \ldots, \ell_m$ satisfying Kraft inequality, try to construct a prefix code.

Left as an Exercise.

Reading & Homework

Reading : 5.1, 5.2

Homework : Problems 5.1, 5.3