Information Theory

Mohamed Hamada

Software Engineering Lab
The University of Aizu

Email: hamada@u-aizu.ac.jp
URL: http://www.u-aizu.ac.jp/~hamada

Today's Topics

- **Entropy review**
- **Entropy and Data Compression**
- **Uniquely decodable codes**
- **Prefix Code**
- **Average Code Length**
- **Shannon's First Theorem**
- **Kraft-McMillan Inequality**
- **Code Efficiency**
- **Code Extension**

Entropy H(S)

- Entropy is the average information content of a source

$$\mathrm{H}(S) = \mathrm{E}\left[\mathrm{I}(s_k)\right]$$

$$\mathrm{H}(S) = \sum_{k=0}^{K-1} p_k \log_2\left(\frac{1}{p_k}\right)$$

Conditional Entropy H(Y|X)

Is the amount of information contained in Y such that X is given

$$\boxed{H(Y \mid X) = = \Sigma_j P(X=v_j)\, H(Y \mid X = v_j)}$$

Joint Entropy

Is the amount of information contained in both events X and Y

$$H(X, Y) = -\sum_{X,Y} p(x,y) \log p(x,y)$$

Chain Rule

Chain Rule

Relationship between conditional and joint entropy

$$\boxed{H(X,Y) = H(X) + H(Y|X)}$$

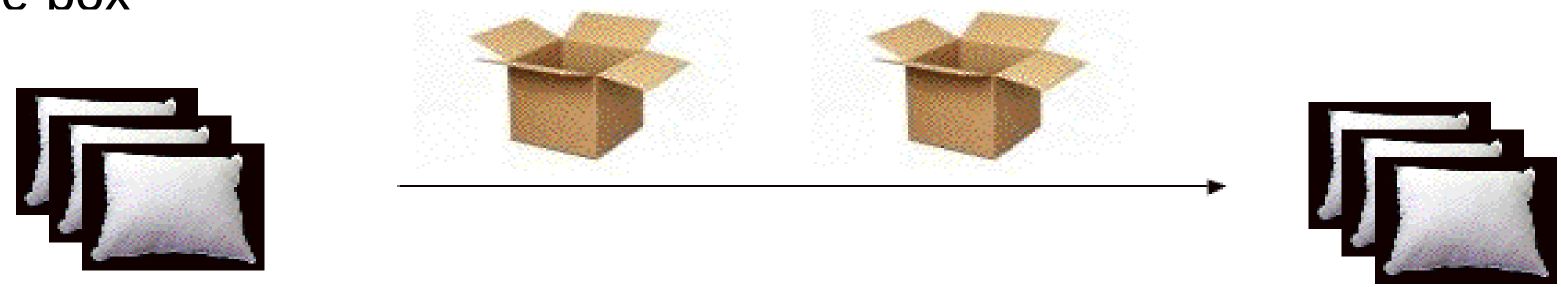
Entropy, Coding and Data Compression

Data vs. Information

- “yes,” “not,” “yes,” “yes,” “not” “not” ...
- In ASCII, each item is $3 \cdot 8 = 24$ bits ***of data***
- But if the only possible answers are “yes” and “not,” there is only one bit ***of information*** per item

Compression = Squeezing out the "Air"

- Suppose you want to ship pillows in boxes and are charged by the size of the box



- To use as few boxes as possible, squeeze out all the air, pack into boxes, fluff them up at the other end
- Lossless data compression = pillows are perfectly restored
- Lossy data compression = some damage to the pillows is OK (MP3 is a lossy compression standard for music)
- Loss may be OK if it is below human perceptual threshold
- Entropy is a measure of limit of **lossless** compression

Fixed length code

Example: Morse Code

A .08	B .01	C .03	D .04	E .12	F .02	G .02	H .06	I .07	J .00	K .01	L .04	M .02
.-	-...	-.-.	-..	.	..-.	--.	....	..	.---	-.-	.-..	--
N .07	O .08	P .02	Q .00	R .06	S .06	T .09	U .03	V .01	W .02	X .00	Y .02	Z .00
-.	---	.--.	--.-	.-.	...	-	..-	...-	.--	-..-	-.--	--..

Example: Telegraphy

Source English letters -> Morse Code

Sender: from Hokkaido

War → .--.-.-.

.--.-.-.

.--.-.-. → War

Receiver: in Tokyo

Coding Messages with Fixed Length Codes

- Example: 4 symbols, A, B, C, D
- A=00, B=01, C=10, D=11
- In general, with *n* symbols, codes need to be of length lg *n*, rounded up
- For English text, 26 letters + space = 27 symbols, length = 5 since $2^4 < 27 < 2^5$

  (replace all punctuation marks by space)

Uniquely decodable codes

- **If any encoded string has only one possible source string producing it then we have unique decodablity**

- **Example of uniquely decodable code is the *prefix code***

Prefix Coding (Instantaneous code)

- A **prefix code** is defined as a code in which ***no*** codeword is the prefix of some other code word.
- A prefix code is ***uniquely decodable.***

Example

Source Symbol	Code A	Code B (Prefix Code)	Code C
	Symbol Codeword	Symbol Codeword	Symbol Codeword
s_0	0	0	0
s_1	1	10	01
s_2	00	110	011
s_3	11	111	0111

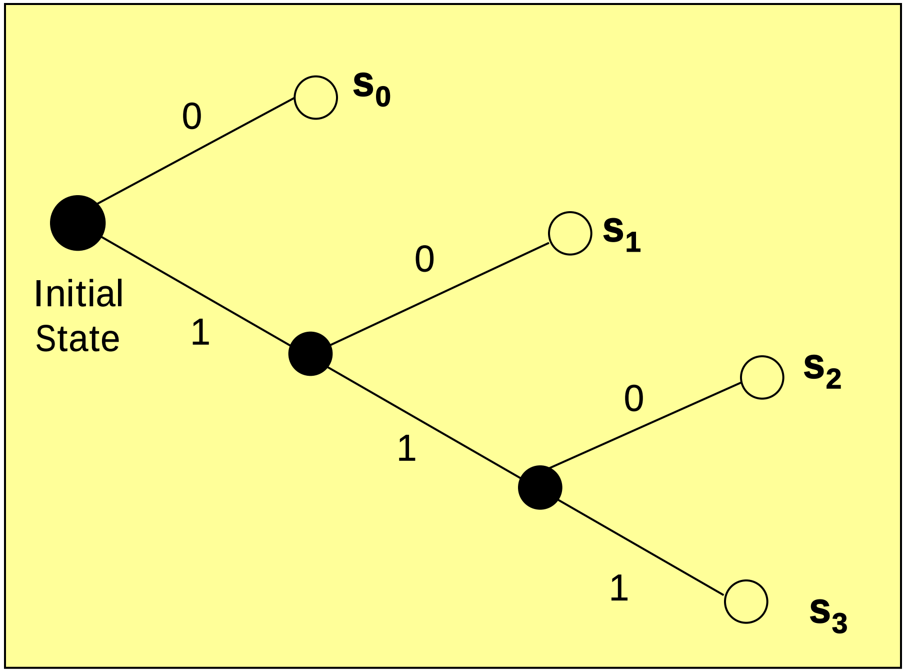
Uniquely Decodable Codes

Decoding of a Prefix Code

Example

Code B	
Source Symbol s_k	Symbol Codeword c_k
s_0	0
s_1	10
s_2	110
s_3	111

Decision Tree for Code B



- Example : Decode 1011111000
- Answer : $s_1 s_3 s_2 s_0 s_0$

Prefix Codes

Only one way to decode left to right when message received

Example 1

Symbol	A	B	C	D
Probability	.7	.1	.1	.1
Code	0	100	101	110

Received message:

$$\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{100}_{B}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{110}_{D}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{0}_{A}\underbrace{100}_{B}$$

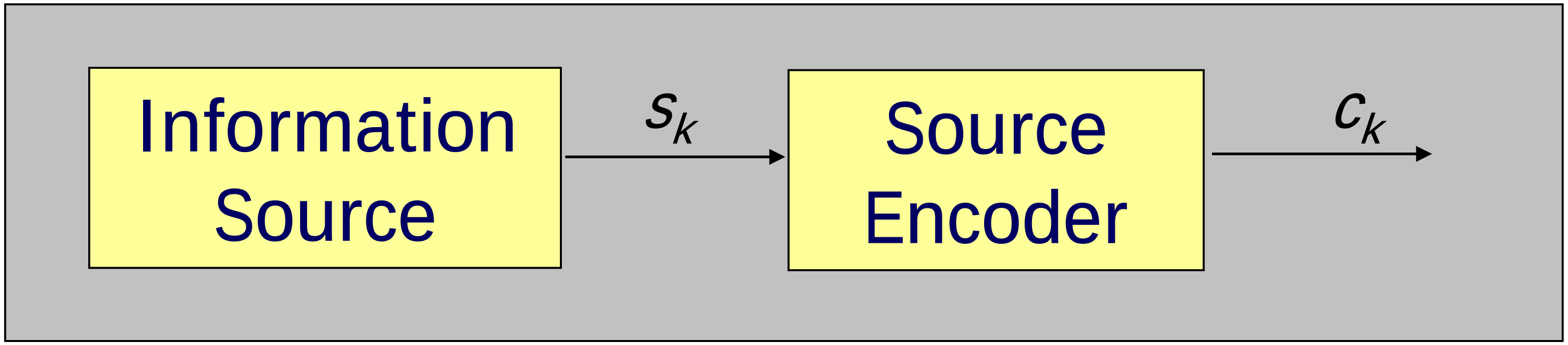
Prefix Codes

Example 2

Source Symbol s_k	Code E Symbol Codeword c_k
A	0
B	100
C	110
D	11

- IS CODE E A PREFIX CODE?
  - NO
  - WHY?
    - Code of D is a prefix to code of C

Average Code Length



- Source has K symbols
- Each symbol s_k has probability p_k
- Each symbol s_k is represented by a codeword c_k of length l_k bits
- **Average codeword length**

$$L = \sum_{k=0}^{K-1} p_k l_k$$

Example: Morse Code

A	B	C	D	E	F	G	H	I	J	K	L	M
.08	.01	.03	.04	.12	.02	.02	.06	.07	.00	.01	.04	.02
.-	-...	-.-.	-..	.	..-.	--.	....	..	.---	-.-	.-..	--
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
.07	.08	.02	.00	.06	.06	.09	.03	.01	.02	.00	.02	.00
-.	---	.--.	--.-	.-.	...	-	..-	...-	.--	-..-	-.--	--..

Average codeword length

$$L = \sum_{k=0}^{K-1} p_k l_k$$

= .08 * 2 + .01 * 4 + ... + .02 *4 + .00 * 4

Shannon's First Theorem: The Source Coding Theorem

$$\boxed{L \geq \mathrm{H}(S)}$$

- The outputs of an information source cannot be represented by a source code whose average length is less than the source entropy

Average Code Length

Example

Average bits per symbol:

$L = .7 \cdot 1 + .1 \cdot 3 + .1 \cdot 3 + .1 \cdot 3 = 1.6$

bits/symbol (down from 2)

A	B	C	D
.7	.1	.1	.1
0	100	101	110

Another prefix code that is better

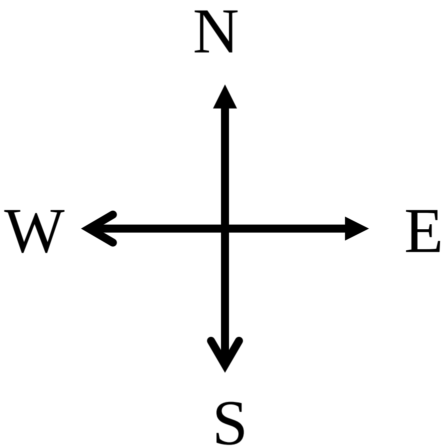
$L = .7 \cdot 1 + .1 \cdot 2 + .1 \cdot 3 + .1 \cdot 3 = 1.5$

A	B	C	D
.7	.1	.1	.1
0	10	110	111

Source Entropy Examples

Robot Example

- 4-way random walk



$$prob(x=S)=\frac{1}{2},\ prob(x=N)=\frac{1}{4}$$

$$prob(x=E)=prob(x=W)=\frac{1}{8}$$

$$H(X)=-(\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{4}\log_2\frac{1}{4}+\frac{1}{8}\log_2\frac{1}{8}+\frac{1}{8}\log_2\frac{1}{8})=1.75bps$$

Source Entropy Examples

Robot Example

Prefix Codes

symbol k	p_k	*fixed-length codeword*	*variable-length codeword*
S	*0.5*	*00*	*0*
N	*0.25*	*01*	*10*
E	*0.125*	*10*	*110*
W	*0.125*	*11*	*111*

symbol stream : S S N W S E N N N W S S S N E S S

fixed length: **00 00 01 11 00 10 01 01 11 00 00 00 01 10 00 00** 32bits

variable length: **0 0 10 111 0 110 10 10 111 0 0 0 10 110 0 0** 28bits

4 bits savings achieved by VLC (redundancy eliminated)

Entropy, Compressibility, Redundancy

- Lower entropy <=> More redundant <=> More compressible
- Higher entropy <=> Less redundant <=> Less compressible

Entropy and Compression

- First-order entropy is theoretical minimum on code length when only frequencies are taken into account
- L=.7·1+.1·2+.1·3+.1·3 = 1.5
- First-order Entropy = 1.353

A	B	C	D
.7	.1	.1	.1
0	10	110	111

- First-order Entropy of English is about 4 bits/character based on “typical” English texts

Bits

You are watching a set of independent random samples of X

You see that X has four possible values

P(X=A) = 1/4	P(X=B) = 1/4	P(X=C) = 1/4	P(X=D) = 1/4

So you might see output: BAACBADCDADDDA...

You transmit data over a binary serial link. You can encode each reading with two bits (e.g. A = 00, B = 01, C = 10, D = 11)

2 bits on average per symbol

0100001001001110110011111100...

Fewer Bits

Someone tells you that the probabilities are not equal

$P(X=A) = 1/2$	$P(X=B) = 1/4$	$P(X=C) = 1/8$	$P(X=D) = 1/8$

Is it possible...

...to invent a coding for your transmission that only uses 1.75 bits on average per symbol. How?

Fewer Bits

P(X=A) = 1/2	P(X=B) = 1/4	P(X=C) = 1/8	P(X=D) = 1/8

It's possible...

...to invent a coding for your transmission that only uses 1.75 bits on average per symbol.

A	**0**
B	**10**
C	**110**
D	**111**

(This is just one of several ways)

Fewer Bits

Suppose there are three equally likely values...

P(X=A) = 1/3	P(X=B) = 1/3	P(X=C) = 1/3

Here's a naïve coding, costing 2 bits per symbol

A	**00**
B	**01**
C	**10**

Can you think of a coding that would need only 1.6 bits per symbol on average?

In theory, it can in fact be done with 1.58496 bits per symbol.

Kraft-McMillan Inequality

$$\sum_{k=0}^{K-1} 2^{-l_k} \le 1$$

Example

Source Symbol s_k	Code D	
	Symbol Codeword c_k	Codeword Length l_k
s_0	0	1
s_1	10	2
s_2	110	3
s_3	11	2

- If codeword lengths of a code satisfy the Kraft McMillan's inequality, then a prefix code with these codeword lengths ***can be*** constructed.
- For code D
  - $2^{-1}+ 2^{-2}+ 2^{-3}+ 2^{-2}=9/8$
  - This means that Code D **IS NOT A PREFIX CODE**

Use of Kraft-McMillan Inequality

- We may use it if the number of symbols are large such that we cannot simply by inspection judge whether a given code is a prefix code or not
- **WHAT Kraft-McMillan Inequality Can Do:**
  - It can determine that a given code IS NOT A PREFIX CODE
  - It can identify that a prefix code could be constructed from a set of codeword lengths
- **WHAT Kraft-McMillan Inequality Cannot Do:**
  - It cannot guarantee that a given code is indeed a prefix code

Example

Source Symbol s_k	Code E	
	Symbol Codeword c_k	Codeword Length l_k
s_0	0	1
s_1	100	3
s_2	110	3
s_3	11	2

- For code E
  - $2^{-1} + 2^{-2} + 2^{-3} + 2^{-3} = 1$ and hence satisfy Kraft-Mcmillan inequality
- IS CODE E A PREFIX CODE?
  - NO
  - WHY?
    - s_3 is a prefix to s_2

Code Efficiency η

$$\boxed{\eta = \frac{\mathrm{H}(S)}{L}}$$

- An efficient code means η→1

Examples

Source Symbol s_k	Symbol Probability p_k	**Code I**		**Code II**	
		Symbol Codeword c_k	Codeword Length l_k	Symbol Codeword c_k	Codeword Length l_k
s_0	1/2	00	2	0	1
s_1	1/4	01	2	10	2
s_2	1/8	10	2	110	3
s_3	1/8	11	2	111	3

- Source Entropy
  - $H(S) = 1/2\log_2(2) + 1/4\log_2(4) + 1/8\log_2(8) + 1/\log_2(8)$
    $= 1\ ¾$ bits/symbol

Code I

$$L = 2 \times \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} \right) = 2$$

$$\eta = \frac{7/4}{2} = 0.875$$

Code II

$$L = \left(1 \times \frac{1}{2} + 2 \times \frac{1}{4} + 3 \times \frac{1}{8} + 3 \times \frac{1}{8} \right) = \frac{7}{4}$$

$$\eta = \frac{7/4}{7/4} = 1$$

For a Prefix Code

- Shannon's First Theorem $\boxed{\mathrm{H}(S) \leq L < \mathrm{H}(S) + 1}$

$$L = \mathrm{H}(S) \quad \textbf{if} \quad p_k = 2^{-l_k} \; \forall k$$

What is the Efficiency η?

η=1

if $p_k \neq 2^{-l_k}$ for some k $\Rightarrow$ η<1

However, we may increase efficiency by extending the source

Increasing Efficiency by Source Extension

- By extending the source we may potentially increase efficiency
- The drawback is
  - Increased decoding complexity

$$H(S^n) \le \bar{L}_n < H(S^n) + 1$$

$$nH(S) \le \bar{L}_n < nH(S) + 1$$

$$H(S) \le \frac{\bar{L}_n}{n} < H(S) + \frac{1}{n}$$

$$\eta = \frac{H(S)}{\bar{L}_n / n}$$

$\eta \to 1$ when

$n \to \infty$

Extension of a Discrete Memoryless Source

- Treats Blocks of *n* successive symbols

Information Source

$$S = \{s_0, s_1, \ldots, s_{K-1}\}$$

$$\Pr\{s_k\} = p_k, k = 0, 1, \ldots, K-1$$

$$\sum_{k=0}^{K-1} p_k = 1$$

Extended Information Source

$$S^n = \{\sigma_0, \sigma_1, \ldots, \sigma_{K^n-1}\}$$

$$\Pr\{\sigma_i\} = q_i, i = 0, 1, \ldots, K^n-1$$

$$\sum_{i=0}^{K^n-1} p_i = 1$$

Example

- $S=\{s_0,s_1,s_2\}$, $p_0=1/4$, $p_1=1/4$, $p_2=1/2$
- $H(S)=(1/4)\log_2(4)+ (1/4)\log_2(4)+ (1/2)\log_2(2)$

  $H(S)=3/2$ bits

Second-Order Extended Source

Symbols of S^2	σ_0	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	σ_8
Sequence of Symbols from S	$s_0\,s_0$	$s_0\,s_1$	$s_0\,s_2$	$s_1\,s_0$	$s_1\,s_1$	$s_1\,s_2$	$s_2\,s_0$	$s_2\,s_1$	$s_2\,s_2$
$P\{\sigma_i\}$, i=0,1,…,8	1/16	1/16	1/8	1/16	1/16	1/8	1/8	1/8	1/4

By Computing: $H(S^2)=3$ bits

Summery

- Source Encoding
  - ***Efficient*** representation of information sources
- Source Coding Requirements
  - Uniquely Decodable Codes
- Prefix Codes
  - No codeword is a prefix to some other code word

Code Efficiency

$$\boxed{\eta = \frac{\mathrm{H}(S)}{L}}$$

Kraft's Inequality

$$\boxed{\sum_{k=0}^{K-1} 2^{-l_k} \leq 1}$$

Source Coding Theorem

$$\boxed{\mathrm{H}(S) \leq L < \mathrm{H}(S) + 1}$$

End