

CAP5015-01 Multimedia Compression on the Internet

University of Central Florida
School of Electrical Engineering and Computer Science

Why Data Compression?

- Decrease storage requirements
- Strict bandwidth bound of communication channels
- Effective use of communication bandwidth; reduce amount of data without losing information
- Improves program execution speed by reducing disk access times
- Increased data security
- Multimedia data on information superhighway
- Growing number of “killer” applications.

(Digital Library, Massive text databases like TREC, web sites like google, medical imaging, satellite images, journals(IEEE,ACM), Newspapers etc. etc.)

Data Compression Metrics

- For data transmission application, the goal is speed. Speed of transmission depends upon the number of bits sent and obviously data compression helps the situation.
- Utilization of the available bandwidth is another goal of data compression.
- The time required for the encoder to generate the coded message, and the time required for the decoder to recover the original message are also important factors. In a data storage application, although the degree of compression is the primary concern, it is nonetheless necessary that the algorithm be efficient in order for the scheme to be practical.

Approximate Bit Rates for Uncompressed Sources (I)

Telephony 8000 samples/sec X 12 bit/sample
(Bandwidth=3.4kHz) = 96 kb/sec

Wideband Speech 16,000 s/s X 14 b/s = 224 kb/sec
(Tele.audio bw~7 kHz)

Compact Disc Audio 44,100 s/s X 16 b/s X 2 channels
(Bandwidth~ 20kHz) = 1.41 Mb/sec

Large Image Files

- low-resolution, TV quality, color video image: 512x512 pixels X 24 (8 bits/pixel, and 3 colors) = 6.3 Mb/image .
- 24 X 36 mm (35-mm) negative photograph scanned at 12 micron resolution 3000 X 2000 pixels/color, 8 bits/pixel, and 3 colors gives 144 Mbits.
- 14 X 17 inch radiograph scanned at 70 micron: 5000 X 6000 pixels, 12 bits/pixel gives 360 Mbits. Medium size hospital generates terrabytes each year.
- LANDSAT Thematic Mapper scene: 6000 X 6000 pixels/spectral band, 8 bits/pixel, and 6 nonthermal spectral bands : 1700 M bits
- Computer graphics: lightfield representation of Michelangelo's David: hundreds of megabytes

Approximate Bit Rates for Uncompressed Sources

Video

640x480 color pixel X 24 bits/pixel X 30 images/sec=221Mb/s

CIF (Videoconferencing: 360x288): 75Mb/sec

CCIR(TV: 720x576): 300 Mb/sec

HDTV (1260x720 pixels X 24 b/p X 80 images/sec): 1.3 Gb/sec

Gap between available bandwidth and the required bandwidth can be filled using compression!

Why Compression now?

- **Enabling technology: computers, digital communication, wireless Communication, and a solid foundation of 25-50 years of research and development of algorithms.**
- **VLSI technology, high speed microprocessors and DSPs.**
- **For images and video data, a better model of perceptually based distortion measures.**
- **Proliferation of Standards (Facsimile, Sound, Images and Video)**

Standards

Standards Bodies: ITU, US ANSI Committee, TIA, ETSI, TIC.
IEEE, CCITT

Fax: Group 3 and Group 4, JBIG

Still Images: JPEG(0.25-2 bit/pixel), JPEG-2000.

Video: MPEG1(1.5Mb/s), MPEG2(6-10 Mb/s),MPEG-2000

High quality audio for MPEG:84/128/192 kbs/s per channel

There are at least 12 audio standards including a few new standards used for cellular phones (GSM, I-95, UTMS).

(Practically, no standard for text data . The .pdf format is image based, html is used in most web based text interchanges. An emerging new format which is getting popular is XML. Various compression algorithms are used: Huffman, arithmetic, compress, Gzip, An emerging new algorithm is Bzip2)

Acronyms

ITU (formerly CCITT): International Telecommunication Union

TIA: Telecommunication Industries Association

ETSI: European Telecommunication Standards Institute

NIST: National Institute of Standards

TTC: Japanese Telecommunications Technology Committee

ISO: International Standards Organization

IEEE: Institute of Electrical and Electronic Engineering

ACM: Association of Computing Machinery

JBIG: Joint Bi-level Image Group

JPEG: Joint Photographic Expert Group

MPEG: Moving Pictures Expert Group

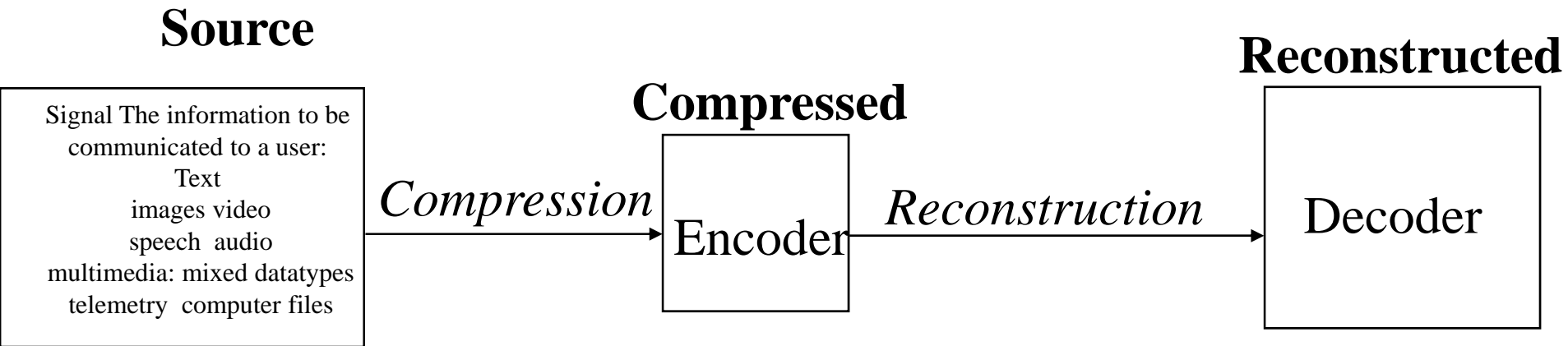
Synonyms and Related Areas

Source coding Bandwidth compression
Data compaction Redundancy removal
Signal Compression Coding Data Compression

Lossless and Lossy Data Compression

Data Encryption
Error Correcting Codes

Compression and Reconstruction



Lossless: **Reconstructed** is identical to **Source**.

Lossy: **Reconstructed** is different from **Source**.

Usually more compression with lossy, but give up exactness.

Lossless compression

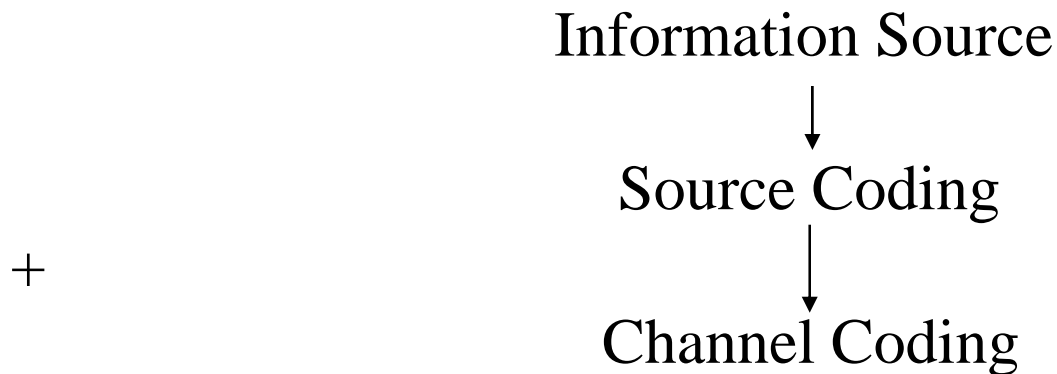
- Lossless compression can perfectly recover original data (if no storage or transmission bit errors, i.e., noiseless channel)
- Different names: noiseless coding, lossless coding, invertible coding, entropy coding, data compaction
(Example: variable length binary codewords (or no compression))
- Only works for digital sources.

Lossy Compression

- Reconstructed data is an approximation of the original data.
- Quantization and Compression
- Mostly used for image data
- Data could be digital or analog (usually replaced by sampled digital data)

Source and Channel Coding

Shannon model usually breaks encoder (& decoder) into two pieces:



Source coding reduces the number of bits in order to save on transmission time or storage space.

Channel Coding

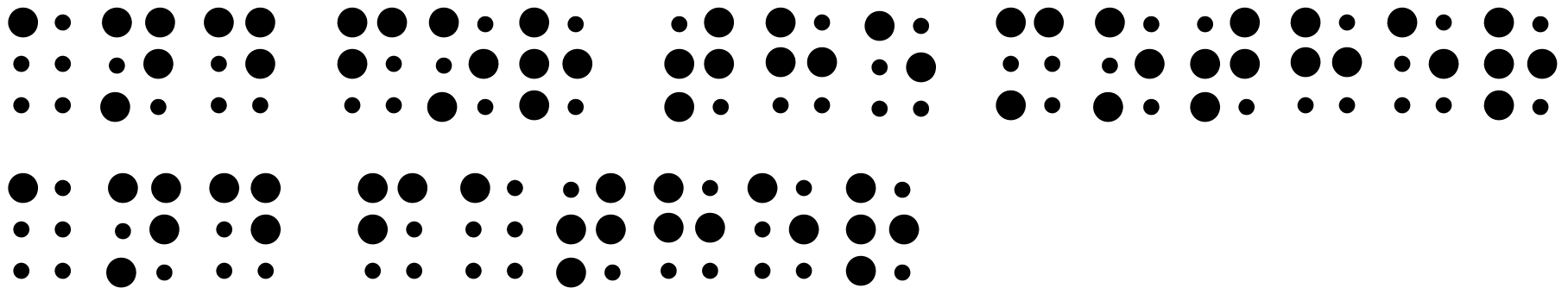
- Channel coding typically increases the number of bits or
- Inserts error correcting bits.

“Image coding” and “speech coding” mean source coding.

Encryption also increases number of bits The Objective is to make the message secure. Data Compression is a kind of encryption for ordinary users.

Grade 1 Braille (1820's)

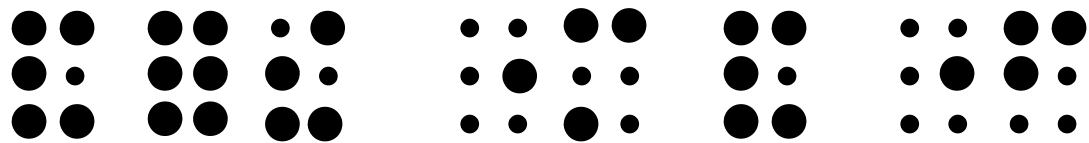
a	b	c	d	e	f	g	h	i	j	k	l	m
n	o	p	q	r	s	t	u	v	w	x	y	z



and for the mother and father
(24 letters; 24 symbols in encoding)

Grade 2 Braille Example

and for the mother and father

The image shows the Braille encoding of the sentence "and for the mother and father". Each word is represented by a 3x3 grid of dots. The words are: "and" (⠁⠗⠎), "for" (⠋⠕⠗), "the" (⠞⠓⠑), "mother" (⠍⠕⠞⠞⠑⠗), "and" (⠁⠗⠎), and "father" (⠋⠁⠞⠑⠗).

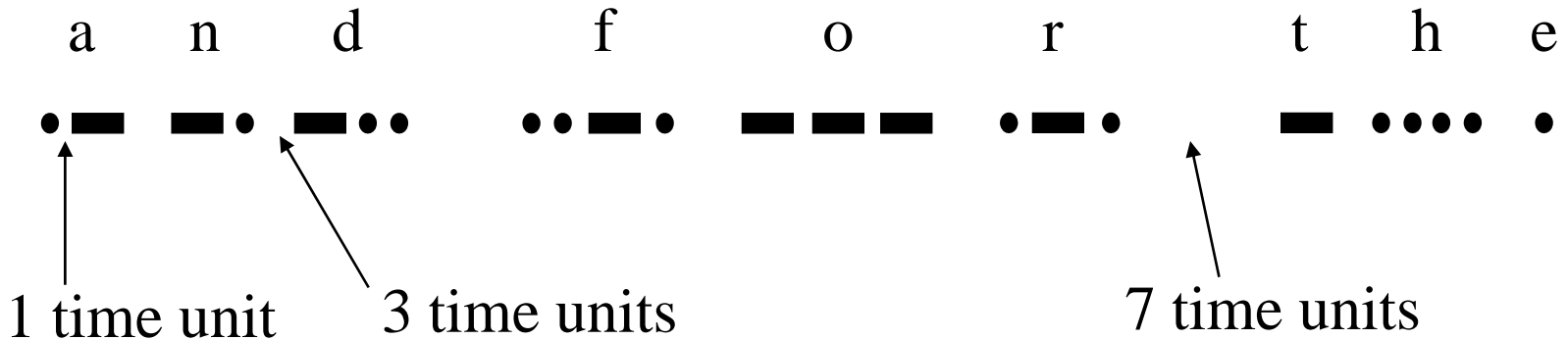
⠁⠗⠎ ⠋⠕⠗ ⠞⠓⠑ ⠍⠕⠞⠞⠑⠗ ⠁⠗⠎ ⠋⠁⠞⠑⠗

and for the mother and father
(24 letters; 8 symbols in encoding)

67% compression (20% is more typical)

Morse Code (1835)

a	• ■	h	••••	o	■ ■ ■	v	••• ■
b	■ •••	i	••	p	• ■ ■ •	w	• ■ ■ ■
c	■ • ■ •	j	• ■ ■ ■ ■	q	■ ■ • ■	x	■ • • ■
d	■ ••	k	■ • ■	r	• ■ •	y	■ • ■ ■
e	•	l	• ■ ••	s	•••	z	■ ■ ••
f	•• ■ •	m	■ ■	t	■		
g	■ ■ •	n	■ •	u	•• ■		



Morse Code, 2

0	■ ■ ■ ■ ■	5	● ● ● ● ●
1	● ■ ■ ■ ■	6	■ ● ● ● ●
2	● ● ■ ■ ■	7	■ ■ ● ● ●
3	● ● ● ■ ■	8	■ ■ ■ ● ●
4	● ● ● ● ■	9	■ ■ ■ ■ ●

Per character (on average)...

8.5 time units for English text

11 time units if characters are equally likely

17 time units for numbers

The Book with no ‘e’

E.V. Wright *‘Gadsby’ published in 1939*

The first sentence:

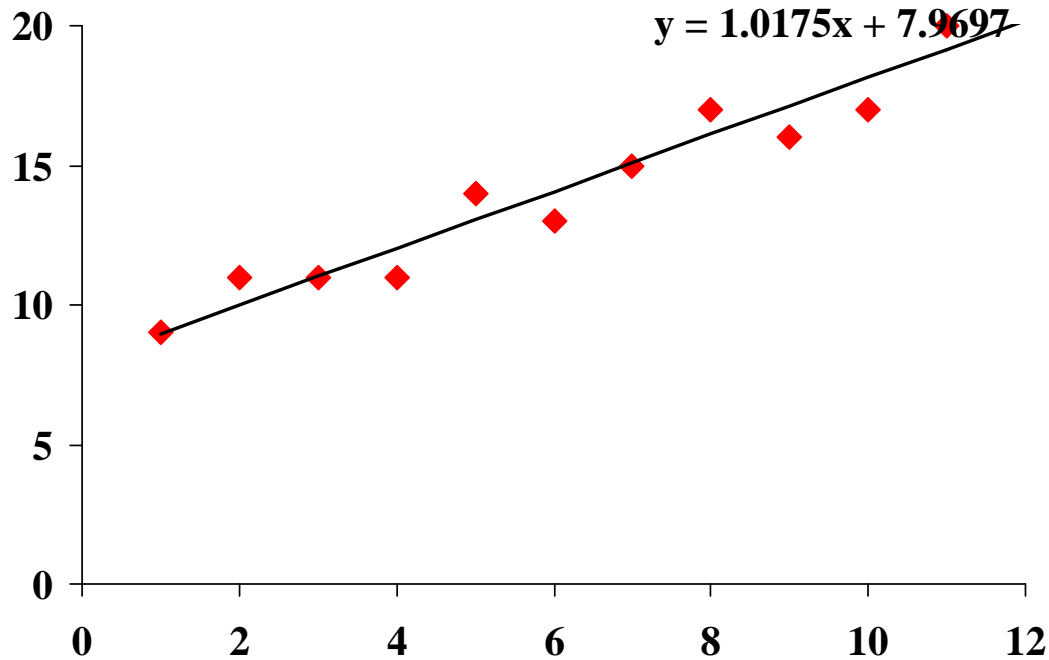
“If Youth, throughout all history, had had a champion to stand up for it; to show a doubting world that a child can think; and, possibly, do it practically; you wouldn’t constantly run across folks today who claim that ‘a child don’t know anything’ ...

A static model will perform poorly as far as data compression is concerned.

Example 1

9 11 11 11 14 13 15 17 16 17 20 21

We could represent each number using 5 bits (could use 4 bits).
⇒ Need $12 * 5 = 60$ bits (or $12 * 4 = 48$ bits) for entire message.



Example 1, cont.

Model: $x_n = n + 8$

Source: 9 11 11 11 14 13 15 17 16 17 20 21

Model: 9 10 11 12 13 14 15 16 17 18 19 20

Residual: 0 1 0 -1 1 -1 0 1 -1 -1 1 1

Only need to transmit the model parameters and residuals.

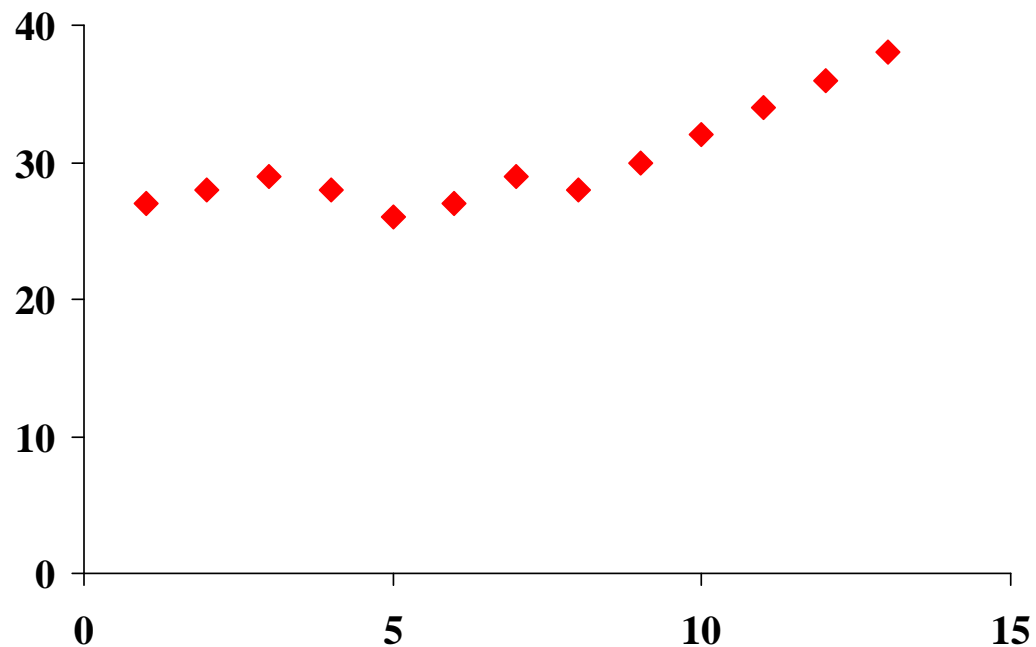
Residuals can be encoded using 2 bits each $\Rightarrow 12 * 2 = 24$ bits.

Savings as long as model parameters encoded in less than
 $60 - 24 = 36$ bits (or $48 - 24 = 24$ bits)

Example 2

27 28 29 28 26 27 29 28 30 32 34 36 38

We could represent each number using 6 bits (could use 4 bits).
⇒ Need $13 * 6 = 78$ bits (or $13 * 4 = 52$ bits) for entire message.



Example 2, cont.

Transmit first value, then successive differences.

Source: 27 28 29 28 26 27 29 28 30 32 34 36 38

Transmit: 27 1 1 -1 -2 1 2 -1 2 2 2 2 2

6 bits for first number, then 3 bits for each difference value.

$\Rightarrow 6 + 12 * 3 = 6 + 36 = 42$ bits (as compared to 78 bits) .

Encoder and decoder must also know the model being used!

Example 3

a_barayaran_array_ran_far_faar_faaar_away

(Note: _ is a single space character)

Sequence uses 8 symbols (a, _, b, r, y, n, f, w)

⇒ 3 bits per symbol are needed, $41 * 3 = 123$ bits total for sequence.

Symbol	Frequency	Encoding	Now require 106 bits, or 2.58 bits per symbol.
a	16	1	
r	8	000	
_	7	001	Compression ratio:
f	3	0100	123/106, or 1.16:1
y	3	0101	
n	2	0111	
b	1	01100	
w	1	01101	

Ad hoc Techniques

- Irreversible compression
- Run-length encoding
- Bit mapping
- Packing into a smaller number of bits
- Move to front
- Differential coding

Run-length Encoding

Example: two character alphabet {a,b}

aaaaaaaaaaaaabbbbbbbbaaaaaaaaaabbbbbbbbbbbbbbbbaaa

Uncompressed message requires $50 * 1 = 50$ bits

Run-length encoding (arbitrarily start with a)

13 8 11 15 3

Use 8 bits for each number $\Rightarrow 8 * 5 = 40$ bits.

Note: may increase number of bits required.

Bit Mapping

Eschew . . . ad . . . hoc . . . compression methods

40 characters * 8 bits = 320 bits

111111000110011100111111111110000011111111

Eschewadhoccompressionmethods

40 bits + 29 characters * 8 bits = 272 bits

Move-To-Front Encoding

The basic idea is to maintain the alphabet A of symbols as a list where frequently occurring symbols are located near the front. A symbol can be encoded as an index of the position of the symbol in the alphabet or as the number of symbols that precede it in the list. After the symbol is encoded it is moved to the front of the list which is the ‘move-to-front’ step. This step anticipates that it will be read many more times and will be a very common symbol for a while. This method is **locally adaptive** in the sense that it adapts itself to the frequencies of symbols in local areas of the input stream.

Move-To-Front Examples

Alphabet A= (a,b,c,d,) Message= abcddcbaaaab

Output	Alphabet
0	a,b,c,d
1	b,a,c,d
2	c,b,a,d
3	d,c,b,a
0	d,c,b,a
1	c,d,b,a
1	b,c,d,a
3	a,b,c,d
0	a,b,c,d
0	a,b,c,d
0	a,b,c,d
1	b,a,c,d

MTF..Second Strategy

Alphabet=(a,b,c,d,m,n,o,p)

Message=abcdcdbamnoppnm

Code with MTF= 0123012345670123 Average value=2.5

Code without MTF=0123321045677654 Average value=3.5

If the average value is small (it will be shown later in this course) that the output of the MTF can be compressed efficiently using Huffman or arithmetic coding. Use of MTF does not always guarantee better average. Try: abcdmnopabcdmnop.

Note the second strategy does not need $O(|A|)$ data movement at each step.

Differential Coding

Front Compression

a	a
aback	1back
abacus	4us
abalone	3lone
abandon	3ndon
abase	3se
abash	4h
abate	3te