

Data Compression



Mark Matthews PhD

Data compression

Compression reduces the size of a file:

- To save **space** when storing it.
- To save **time** when transmitting it.
- Most files have lots of redundancy.

Who needs compression?

- Moore's law: # transistors on a chip doubles every 18–24 months.
- Parkinson's law: data expands to fill space available.
- Text, images, sound, video, ...

“ Everyday, we create 2.5 quintillion bytes of data—so much that 90% of the data in the world today has been created in the last two years alone. ” — IBM report on big data (2011)

Basic concepts ancient (1950s), best technology recently developed.

Applications

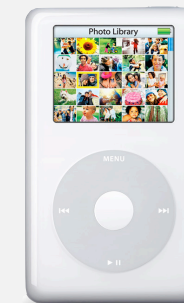
Generic file compression.

- Files: GZIP, BZIP, 7z.
- Archivers: PKZIP.
- File systems: NTFS, ZFS, HFS+, ReFS, GFS.



Multimedia.

- Images: GIF, JPEG.
- Sound: MP3.
- Video: MPEG, DivX™, HDTV.



Communication.

- ITU-T T4 Group 3 Fax.
- V.42bis modem.
- Skype, Google hangout.



Databases. Google, Facebook,

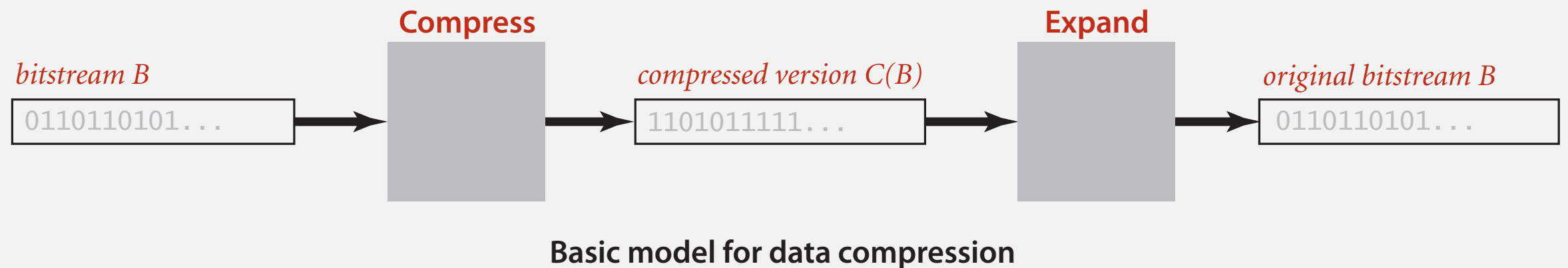


Lossless compression and expansion

Message. Binary data B we want to compress.

Compress. Generates a "compressed" representation $C(B)$.

Expand. Reconstructs original bitstream B .



Compression ratio. Bits in $C(B)$ / bits in B .

Ex. 50–75% or better compression ratio for natural language.

Food for thought

Data compression has been omnipresent since antiquity:

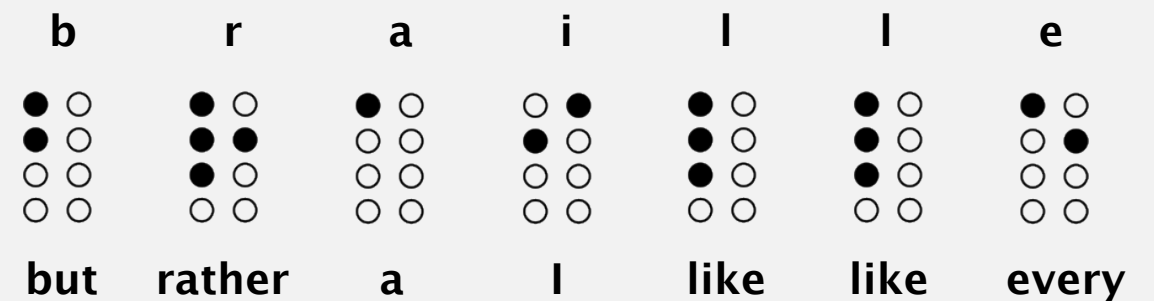
- Number systems.
- Natural languages.
- Mathematical notation.



$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$$

has played a central role in communications technology,

- Grade 2 Braille.
- Morse code.
- Telephone system.



and is part of modern life.

- MP3.
- MPEG.



Q. What role will it play in the future?

Morse Code

Transmit text by series of tones, lights or clicks.



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.

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	■ ■ ● ●

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

Morse Code

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.



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	■ ■ ● ●

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

Claude Shannon - Information Theory



The Bell System Technical Journal

Vol. XXVII

July, 1948

No. 3

A Mathematical Theory of Communication

By C. E. SHANNON

INTRODUCTION

THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist¹ and Hartley² on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message

Information entropy is the average rate at which information is produced by a stochastic source of data.

Data representation: genomic code

Genome. String over the alphabet $\{A, C, T, G\}$.

Goal. Encode an N -character genome: `ATAGATGCATAG...`

Standard ASCII encoding.

- 8 bits per char.
- $8N$ bits.

char	hex	binary
A	41	01000001
C	43	01000011
T	54	01010100
G	47	01000111

Two-bit encoding.

- 2 bits per char.
- $2N$ bits.

char	binary
A	00
C	01
T	10
G	11

Fixed-length code. k -bit code supports alphabet of size 2^k .

Amazing but true. Some genomic databases in 1990s used ASCII.

Universal data compression

[US Patent 5,533,051](#) on "Methods for Data Compression", which is capable of compression **all** files.

[Slashdot](#) reports of the Zero Space Tuner™ and BinaryAccelerator™.

*“ ZeoSync has announced a breakthrough in data compression that allows for 100:1 lossless compression of **random** data. If this is true, our bandwidth problems just got a lot smaller.... ”*

Universal data compression

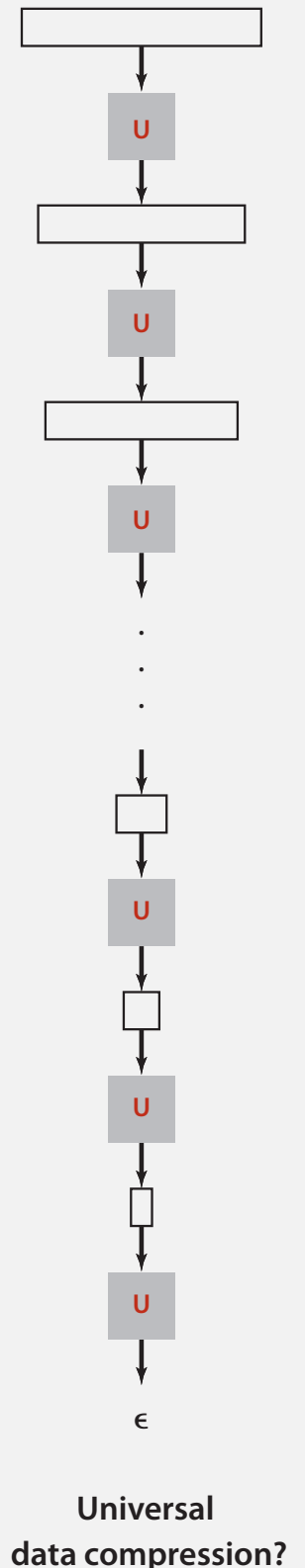
Proposition. No algorithm can compress every bitstring.

Pf 1. [by contradiction]

- Suppose you have a universal data compression algorithm U that can compress every bitstream.
- Given bitstring B_0 , compress it to get smaller bitstring B_1 .
- Compress B_1 to get a smaller bitstring B_2 .
- Continue until reaching bitstring of size 0.
- Implication: all bitstrings can be compressed to 0 bits!

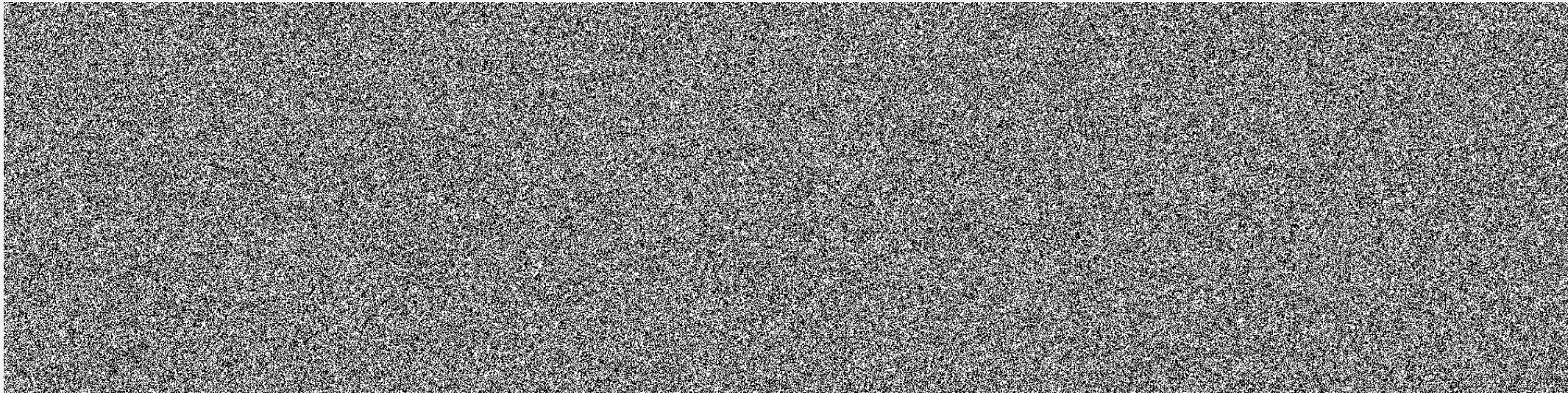
Pf 2. [by counting]

- Suppose your algorithm that can compress all 1,000-bit strings.
- 2^{1000} possible bitstrings with 1,000 bits.
- Only $1 + 2 + 4 + \dots + 2^{998} + 2^{999}$ can be encoded with ≤ 999 bits.
- Similarly, only 1 in 2^{499} bitstrings can be encoded with ≤ 500 bits!



Undecidability

```
% java RandomBits | java PictureDump 2000 500
```



1000000 bits

A difficult file to compress: one million (pseudo-) random bits

```
public class RandomBits
{
    public static void main(String[] args)
    {
        int x = 11111;
        for (int i = 0; i < 1000000; i++)
        {
            x = x * 314159 + 218281;
            BinaryStdOut.write(x > 0);
        }
        BinaryStdOut.close();
    }
}
```


Rdenudcany in Enlgsih Inagugae

Q. How mcuh rdenudcany is in the Enlgsih Inagugae?

“ ... randomising letters in the middle of words [has] little or no effect on the ability of skilled readers to understand the text. This is easy to denmtrasote. In a pubiltacion of New Scnieitst you could ramdinose all the letetrs, keipeng the first two and last two the same, and reibadailty would hadrly be aftcfeed. My ansaylis did not come to much beucase the thoery at the time was for shape and sengeuce retigcionon. Saberi's work sugsegts we may have some pofrweul palrlael prsooscers at work. The resaon for this is suerly that idnetiyfing coentnt by paarllel prseocsing speeds up regnicoiton. We only need the first and last two letetrs to spot chganes in menieng. ” — Graham Rawlinson

A. Quite a bit.

Reading and writing binary data

Binary standard input. Read **bits** from standard input.

```
public class BinaryStdIn
```

<code>boolean readBoolean()</code>	<i>read 1 bit of data and return as a boolean value</i>
<code>char readChar()</code>	<i>read 8 bits of data and return as a char value</i>
<code>char readChar(int r)</code>	<i>read r bits of data and return as a char value</i>
<i>[similar methods for byte (8 bits); short (16 bits); int (32 bits); long and double (64 bits)]</i>	
<code>boolean isEmpty()</code>	<i>is the bitstream empty?</i>
<code>void close()</code>	<i>close the bitstream</i>

Binary standard output. Write **bits** to standard output

```
public class BinaryStdOut
```

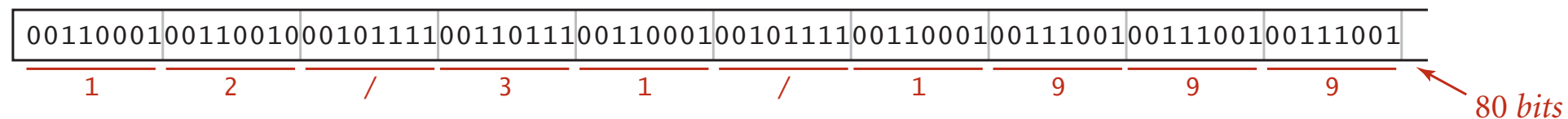
<code>void write(boolean b)</code>	<i>write the specified bit</i>
<code>void write(char c)</code>	<i>write the specified 8-bit char</i>
<code>void write(char c, int r)</code>	<i>write the r least significant bits of the specified char</i>
<i>[similar methods for byte (8 bits); short (16 bits); int (32 bits); long and double (64 bits)]</i>	
<code>void close()</code>	<i>close the bitstream</i>

Writing binary data

Date representation. Three different ways to represent 12/31/1999.

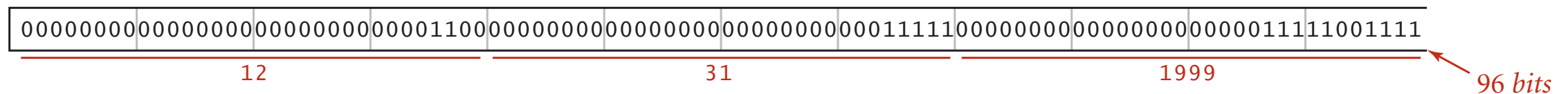
A character stream (StdOut)

```
StdOut.print(month + "/" + day + "/" + year);
```



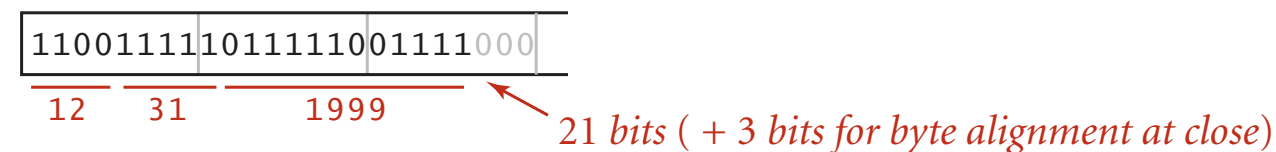
Three ints (BinaryStdOut)

```
BinaryStdOut.write(month);  
BinaryStdOut.write(day);  
BinaryStdOut.write(year);
```



A 4-bit field, a 5-bit field, and a 12-bit field (BinaryStdOut)

```
BinaryStdOut.write(month, 4);  
BinaryStdOut.write(day, 5);  
BinaryStdOut.write(year, 12);
```



Binary dumps

Q. How to examine the contents of a bitstream?

Standard character stream

```
% more abra.txt  
ABRACADABRA!
```

Bitstream represented as 0 and 1 characters

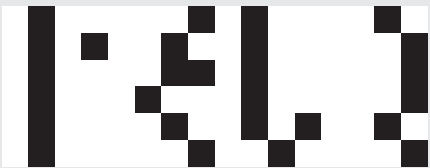
```
% java BinaryDump 16 < abra.txt  
0100000101000010  
0101001001000001  
0100001101000001  
0100010001000001  
0100001001010010  
0100000100100001  
96 bits
```

Bitstream represented with hex digits

```
% java HexDump 4 < abra.txt  
41 42 52 41  
43 41 44 41  
42 52 41 21  
12 bytes
```

Bitstream represented as pixels in a Picture

```
% java PictureDump 16 6 < abra.txt
```



← 16-by-6 pixel window, magnified

96 bits

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SP	!	“	#	\$	%	&	‘	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

Hexadecimal to ASCII conversion table