I. Introduction

**Wht is Dta Cmprsn?** The systematic removal and, later, reintroduction of redundancy from data, and possibly also the removal of insignificant data.

Contains Redundant or Insignificant Parts

Original Data (Or Close Approximation)

56 Kbits of data

Data Compression

14 Kbits of data

Data Decompression

Removes Redundancies and/or Insignificancies

Reinserts Redundancies

Similar to Orange Juice Concentrate:
II. Lossless Vs. Lossy

There are two broad categories of data compression:
- Lossless Compression
- Lossy Compression

Lossless Compression: Original digital data can be exactly reconstructed from the compressed data

- Generally applied to:
  - text files (documents, source code, etc.)
  - program files

- Can achieve modest compression ratios:
  - 2:1 to 8:1 depending on type and content of file

- Examples:
  - UNIX “compress” command
  - PKZIP
  - Ram Doublers, Stacker, etc.
Lossy Compression: Original digital data can only be approximately reconstructed from the compressed data.

- Generally applied to:
  - signals (speech, music, sensor signals, etc.)
  - images (digitized photos, digitized video)

- Can achieve large compression ratios:
  - up to (many tens):1 depending on type & content of data

- Examples:
  - JPEG (for digitized photos)
  - MPEG (for digitized video)
  - speech compression used in digital cell phones
  - digital answering machines
Lossless Example to Motivate Needed Theory

Consider a source w/ alphabet E, A, B, C, B, S

Suppose this is a typical source sequence:

DAABC AAA BADBA AAA BB AAAAC DBC

The # of occurrences of symbols in this sequence are: \( N_A = 12 \quad N_B = 6 \quad N_C = 3 \quad N_S = 3 \)

Total # symbols in sequence = 24

\( N \)

Say we have a binary code for the symbols such that

- Length in bits of codeword \( A = e_A \)
- \( \ldots \)
- Length in bits of codeword \( D = e_D \)

\( B = e_B \)

Total # bits to code sequence = \( N_A e_A + N_B e_B + N_C e_C \)

\( B_r \)

\( = 12 e_A + 6 e_B + 3 e_C + 3 e_D \)
A useful measure of lossless compression = "Avg. #Bits/symbol"

If we "avg over the data" = "Data Analysis Average"

\[ \text{Avg. #Bits/symbol} = \frac{B}{N} \]

\[ = \frac{N}{N} l_a + \frac{N}{N} l_b + \frac{N}{N} l_c + \frac{N}{N} l_d \]

"Freq. of Occurrence"

\[ = 0.5 l_a + 0.25 l_b + 0.125 l_c + 0.125 l_d \]

\[ \Rightarrow \begin{cases} \text{Want } l_a \text{ to be smallest code length} \\ \text{then } l_b \\ \text{then } l_c & l_d \end{cases} \]

\[ \Rightarrow \text{Most likely symbols get fewer bits} \]

Prob. Theory Says: \( \lim_{N \to 00} \frac{N_i}{N} = P_i \)

\( P_i \) \( \text{Prob. of Event} \)

\[ \Rightarrow \text{Base our theory on Probability} \]

\[ \Rightarrow \text{Avg. # Bits/symbol uses Probability Average} \] (Not Data Average)
So, to study lossless theory we need to "specify" a probability model for the source:

\[ P_A, P_B, P_C, P_D \]

\[ \Rightarrow \text{Avg Bits/symbol} = E[\ell] \]

\[ \uparrow \text{RV representing codeword length of coded random symbols} \]

\[ = P_A \ell_A + P_B \ell_B + P_C \ell_C + P_D \ell_D \]

So, for this example we want \( \ell_A \) smallest, then \( \ell_B \), then \( \ell_C = \ell_D \)

So try: \( A = 0 \), \( B = 1 \), \( C = 01 \), \( D = 10 \)

Does that work? NO!

Can't decode sequences uniquely:

Original: DAABCC...

Coded: 1000101

Decoded: \{BA4 ABA B ... \} \( N \)s unique decoding

\{ DAC AB ... \}

\{ etc \}
Code #2

How about: A = 0, B = 10, C = 01, D = 11

No, because CA \rightarrow 010, CB \rightarrow 0110, A0A \rightarrow 010

Code #3

How about: A = 0, B = 10, C = 11, D = X92

D must use 3 bits

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Possibilities

But... No!

If D = 001 \Rightarrow DA \rightarrow 0010, AAB \rightarrow 0010

If D = 101 \Rightarrow DA \rightarrow 1010, BB \rightarrow 1010

If D = 111 \Rightarrow DA \rightarrow 1110, CB \rightarrow 1110
How about: \( A = 0 \) \( B = 10 \) \( C = 110 \) \( D = 111 \)

Code #4

Note this property: No other codeword is a prefix.

Yes, this works! <Verify!>

---

Optimal

So, lossless coding is a constrained optimization problem.

\[
\begin{align*}
\text{minimize} & \quad \text{Avg. Bits/symbol} \\
\text{subject to} & \quad \text{Decodability}
\end{align*}
\]

<table>
<thead>
<tr>
<th>Code #</th>
<th>Avg. bit/symbol</th>
<th>Not Decodable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.25 bits</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1.5 bits</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1.625 bits</td>
<td>Not Decodable</td>
</tr>
<tr>
<td>4</td>
<td>1.75 bits</td>
<td>Decodable</td>
</tr>
</tbody>
</table>

\[ \text{Compare to 2 bits/symbol if we used std. binary coding.} \]

- Original using std. Binary coding uses \( 2 \times 24 = 48 \) bits.
- Coded stream uses \( 1.75 \times 24 = 42 \) bits.
To study this \rightarrow Need Some Theory

\[
\text{Prob. Theory} \quad + \quad \text{Information Theory}
\]

\[
\text{Decodability} \quad \{ \text{at least part of it} \}
\]

Role of Info Theory Here:

- Determine conditions need for decodability

- Determine lower bounds for Avg. bits/symbol

- Provide understanding of how structure of the Source Prob. Model impacts lower bounds

- Provide basis/structure on which practical codes can be built