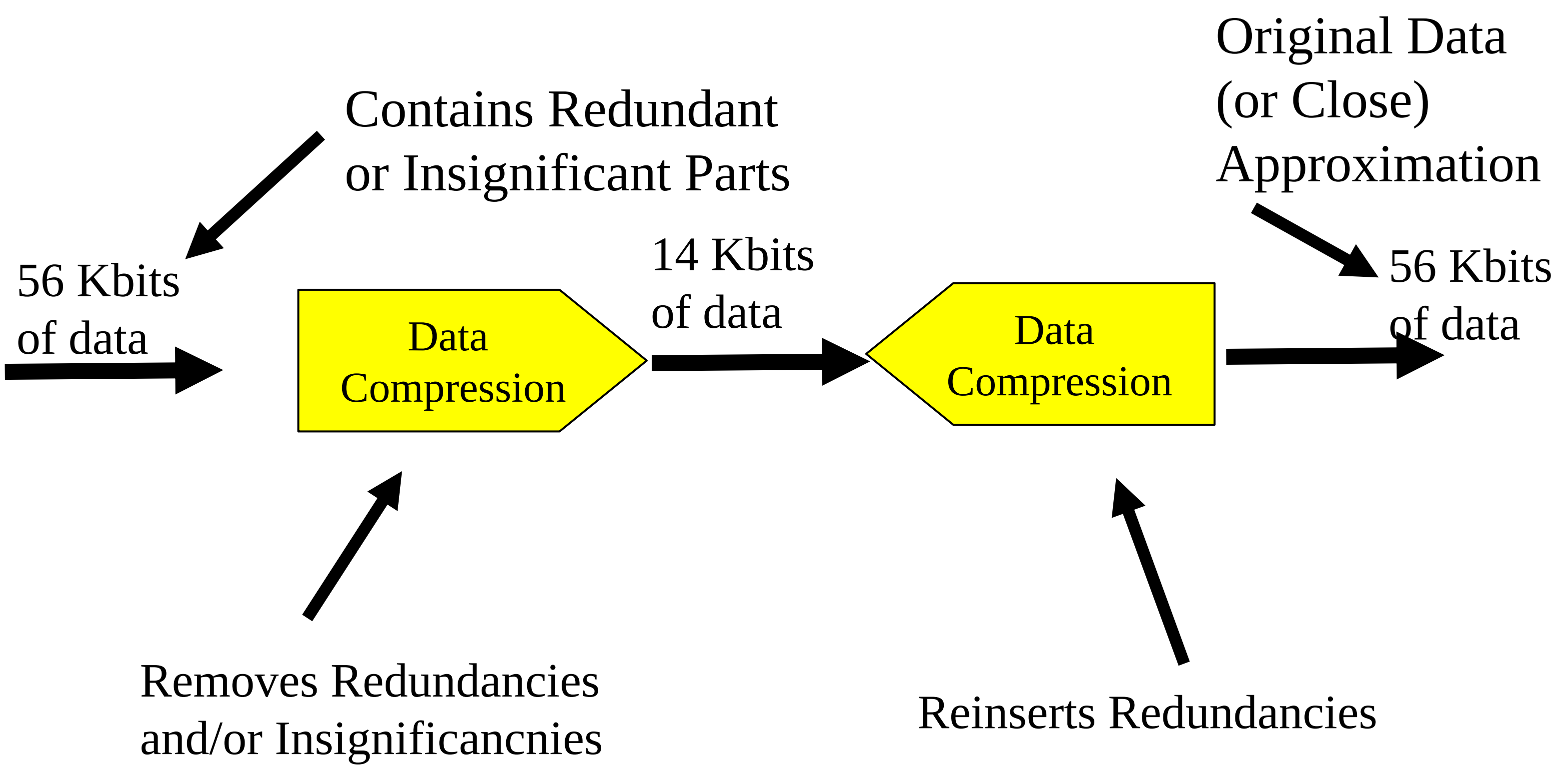
EECE 523

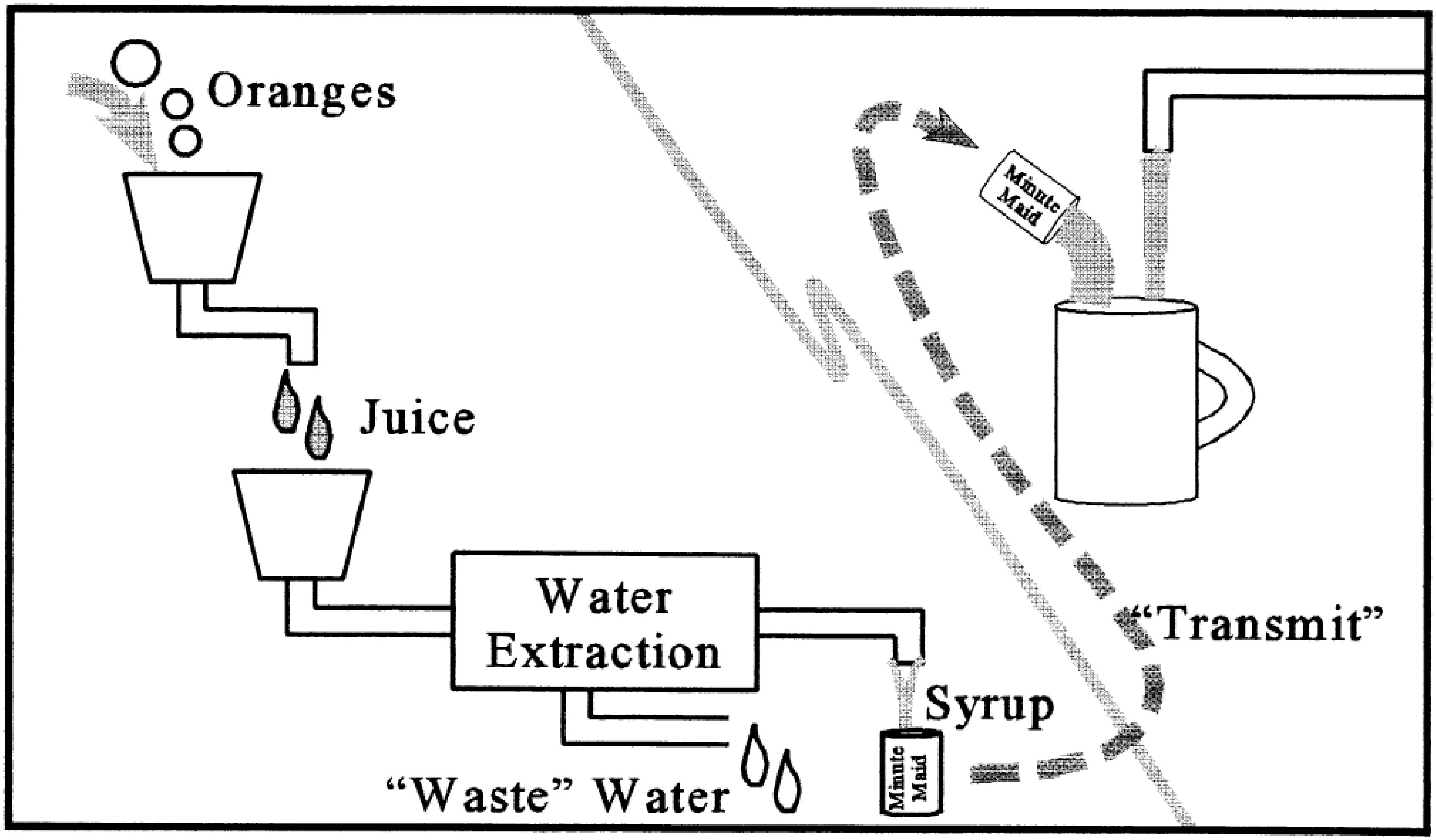
Ch. 1
Introduction
&
Lossless Example

I. Introduction

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



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 compresssed 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 of content of file

- Examples:
  - UNIX "compress" command
  - PKZIP
  - Ram Doublers, Stacker, etc.

<u>Lossy Compression</u>: Original digital data can only be <u>approximately</u> 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 to data

- Examples:
  - JPEG (for digitized photos)
  - MPEG (for digitized video)
  - Speech compression used in digital cell phones
  - Digital answering machines
  - MP3 (for music)

Lossless Example to Motivate Needed Theory

Consider a source with alphabet $\{A, B, C, D\}$

Suppose this is a typical source sequence:

DAABCAAABADBAAABBAAACDBC

The # of occurances of symbols in this sequence are:

$N_A = 12 \quad N_B = 6 \quad N_C = 3 \quad N_D = 3$

$$\underbrace{Total\ \#\ of\ symbols\ in\ sequence}_{\triangleq N} = 24$$

Say we have a binary code for the symbols such that

length in bits of code word A = ℓ_A

length in bits of code word B = ℓ_B

length in bits of code word D = ℓ_D

$$\underbrace{Total\ \#\ of\ bits\ to\ code\ sequence}_{\triangleq B_T} = N_A \ell_A + N_B \ell_B + N_C \ell_C + N_D \ell_D$$

$$= 12\ell_A + 6\ell_B + 3\ell_C + 3\ell_D$$

A useful measure of lossless compression = “Avg” # Bits/symbols

If we “avg over the data”: ⟵ “Data Analysis Average”

$$\text{Avg \# Bits/symbol} = \frac{B_T}{N}$$

$$= \frac{N_A}{N}\ell_A + \frac{N_B}{N}\ell_B + \frac{N_C}{N}\ell_C \; \frac{N_D}{N}\ell_D$$

ℓ_A “Freq. of Occurance”

$$= 0.5\ell_A + 0.25\ell_B$$

⇒Want ℓ_A to be smallest code length
then ℓ_B
then ℓ_C *and* ℓ_D

⇒Most likely symbols get fewer bits

Prob. Theory Says: $$\lim_{N\to\infty} \frac{N_i}{N} = P_i$$

Prob. of ith event

⇒Base on our theory on probability

⇒Avg. # Bits/Symbol uses probability average (not data average)

So, to study lossless <u>theory</u> we need to "specify" a probability <u>model</u> for the source:

P_A, P_B, P_c, P_D

⇒Avg Bits/symbol $= E\{\ell\}$

RV representing
code word 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 ℓ_A smallest, then ℓ_B, then $\ell_C = \ell_D$

So try: A = 0   B = 1   C = 01   D = 10

Code #1

Does that work?   NO!

Can't decode sequences uniquely:

Original: D A A B C …
Coded: 1 0 0 0 | 0 |
Decoded:

$$\left.\begin{cases} BAAABA\ldots \\ DACAB\ldots \\ \quad etc. \end{cases}\right\} \textit{No unique decoding}$$

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

Code #2

No, because

CA → 010
AB → 010

CB → 0110
ADA → 0110

How about: A = 0 B = 10 C = 11 D = xyz

Code # 3

Can't be 00
or 01
Seen above

D Must use 3 bits

xyz	
000	AAA
001	
010	AB
011	AC
100	BA
101	
110	CA
111	

possibilities

But…. No!

If $D = 001 \Rightarrow$ DA 0010
AAB

If $D = 101 \Rightarrow$ DA 1010
BB

If $D = 111 \Rightarrow$ DA 1110
CB

How about: A = 0 B = 10 C = 110 D = 111

Code #4

not a
Code word

Note this property: No other codeword is a <u>prefix</u>

Yes, this works! <Verify!>

So optimal lossless coding is a <u>constrained optimization</u> problem

minimize Avg. Bits/symbol

subject to decodability

Code #	Avg. bits/symbol
1	1.25 bits
2	1.5 bits
3	1.625 bits
4	1.75 bits

Not Decodable

Decodable

Compare to 2 bits/symbol if we used std. binary coding

Original stream using std. binary coding uses 2 x 24 = 48 bits

Coded stream used 1.75 x 24 = 42 bits

Compression!

To study this ➔ Need some theory

Prob. Theory + Decodability	Information Theory (at least <u>part</u> 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