

02/04 [Section 4.8 of the book]

Huffman Codes and Data Compression

Consider storing or communicating a large text file.

How many bits do we need to store or communicate?

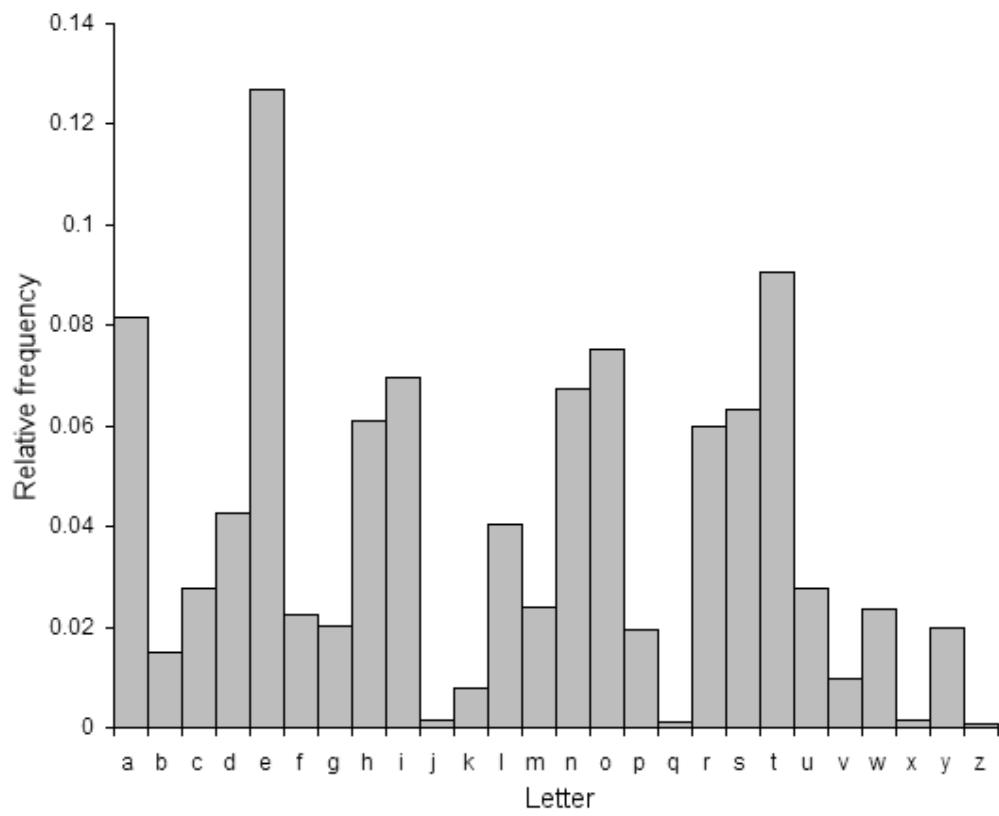
26 alphabets, punctuation symbols,
blank space
 ≤ 32 symbols.

Encode each symbol using 5 bits.

Text file has n symbols
 \Rightarrow Use $5n$ bits. ✓

Question: Can we do better? How?

✓ Observation: Some letters in the alphabet are used more frequently.



→ This idea was already used
in 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	• -	U	• • -
B	- - - .	V	• • • -
C	- - . .	W	• - -
D	- - . .	X	- . . -
E	•	Y	- - . -
F	• . . -	Z	- - - .
G	- - -		
H	• • . .		
I	• •		
J	• - - -		
K	- - . -	1	• - - - -
L	• - - .	2	• - - -
M	- - -	3	• • -
N	- - .	4	• • • -
O	- - -	5	• • • .
P	• - - - .	6	- - . .
Q	- - - . -	7	- - - . .
R	• - - .	8	- - - - .
S	• • -	9	- - - - - .
T	-	0	- - - - - - .

Source: Wiki

•  Pause  Pause .

(i)	a a	✓
(ii)	e t e t	
(iii)	e t a	
(iv)	a e t	
(v)	e k	

Morse: adds a 'pause' after each symbol.

- A Ternary encoding .
- Prefix code : general way to solve this problem .

02/06

Prefix codes

Let S be a set of symbols

Encoding function \equiv Prefix code

$\gamma: S \rightarrow \{0,1\}^*$
 $\text{---} \uparrow \text{binary strings of}$

arbitrary length:

Property of γ :

For any distinct $x, y \in S$,

$\gamma(x)$ is not a prefix of $\gamma(y)$.

How to encode a string using γ :

$x_1 x_2 \dots x_i \dots$ ($x_i \in S$)

$\gamma(x_1) \gamma(x_2) \dots \gamma(x_i) \dots$

Decoding procedure:

(i) Scan from left to right.

(ii) As soon as the bit-string makes sense, map it to some $x_i \in S$.

(iii) Delete bits seen so far.

e.g. (i) $a \rightarrow 01, b \rightarrow 100, c \rightarrow 101$

Is this a prefix code?

(a) Yes

(b) No

(ii) $a \rightarrow 01, b \rightarrow 10, c \rightarrow 101$

No

Notation: For any string r , $|r|$ to denote length of r .

Designing good prefix codes.

Symbols : S

Frequencies: For each $n \in S$,

$$f_n \quad (f_n \in [0,1])$$

Given γ , \leftarrow prefix code
 $\text{average bits per letter}$

$$ABL(\gamma) = \sum_{n \in S} f_n |\gamma(n)|$$

Example:

$$a \rightarrow 01, b \rightarrow 100, c \rightarrow 101$$

$$f_a = 0.4 \quad f_b = 0.3 \quad f_c = 0.3$$

$$\text{ABL}(\gamma) = 2 \times (0.4) + 3(0.3) + 3(0.3) \\ = 2.6.$$

Text has n symbols; $\text{ABL}(\gamma) = S$.

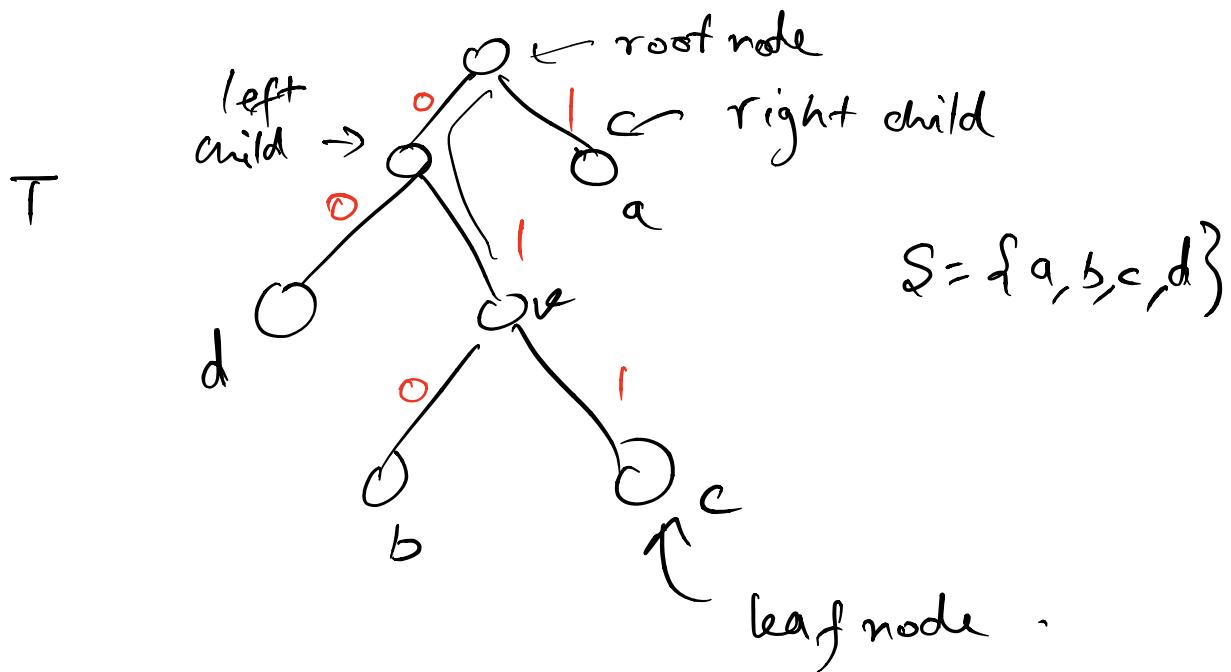
$$\gamma(H) = S_n.$$

$$\hookrightarrow \sum_{n \in S} f_n \cdot n |\gamma(n)| = n \cdot \text{ABL}(\gamma) \\ = n \cdot S.$$

Task: Design γ with minimum ABL .

Prefix codes as λ binary trees.

Labelled binary tree



All the leaf nodes are labelled by distinct symbols from S .

For any node v ,

$$\text{depth}_T(v) = \begin{matrix} (\text{edges}) \\ \text{length of path} \\ \text{from root to } v \end{matrix}$$

→ Given a labelled binary tree T with distinct symbols from S , it specifies a

prefix code γ .

In the example: $\gamma(a) = 1$,

$\gamma(b) = 010$, $\gamma(c) = 011$, $\gamma(d) = 00$

General process: Walk from root to leaf labelled x . Whenever going left add 0 to the end, else add 1.

Obs: γ is a prefix code.

Pf: $x, y \in S$.

If $\gamma(x)$ is a prefix of $\gamma(y)$,

y is a node in a subtree with x as the root node.

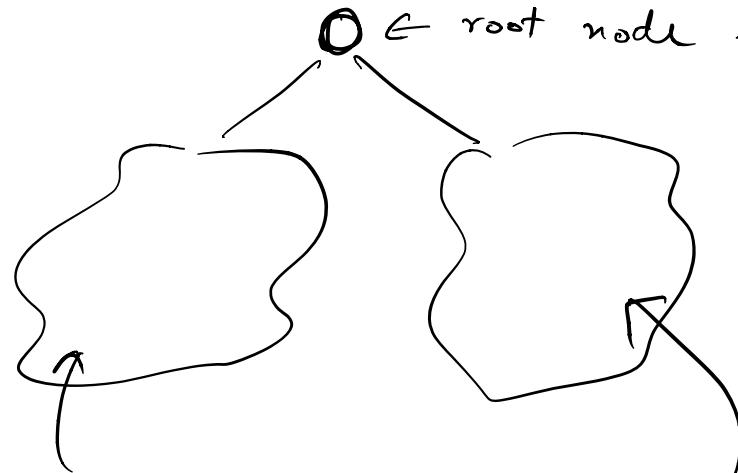
But x, y are leaf nodes.

So this is impossible.

Thus, $\gamma(x)$ is not a prefix of $\gamma(y)$.

Given: γ is prefix code for S .

Construct a labelled binary tree



all $n \in S$

s.t $\gamma(n)$ starts
with O

all $y \in S$

s.t $\gamma(y)$ starts
with P

Recursively apply on the clouds.

→ From now on, let's try to
build a labelled binary tree.

$$ABL(T) = \sum_{x \in \text{Leafnodes}(T)} f_x \text{Depth}_T(x)$$

If T is constructed out of γ ,

Obs. $A\bar{B}L(\gamma) = A\bar{B}L(T)$.

Natural strategy [Fano and Shannon]

Input: S , set of frequencies
 $x \in S$, f_x .

Algorithm: Partition S into S_1 and

S_2 s.t

$\sum_{x \in S_1} f_x$ is as

close to γ_2 as

possible,

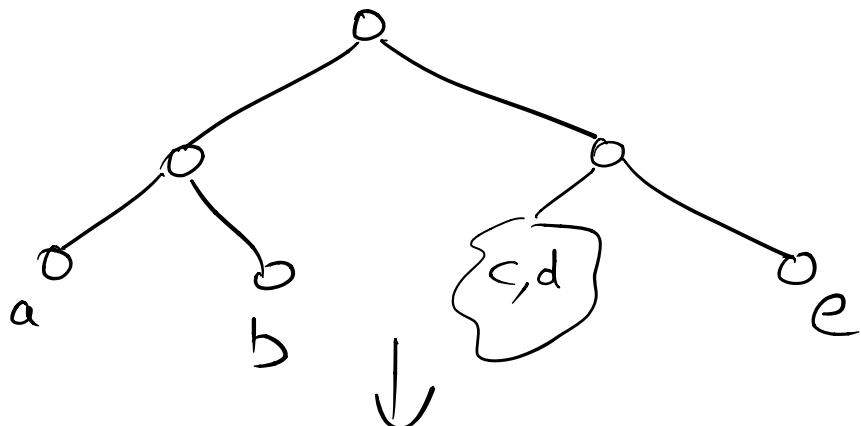
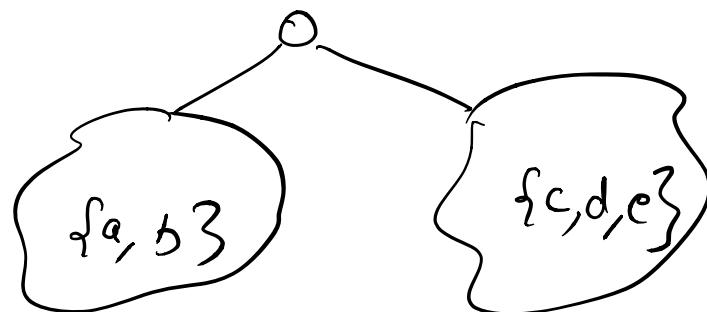
Recurse on S_1 and S_2 .

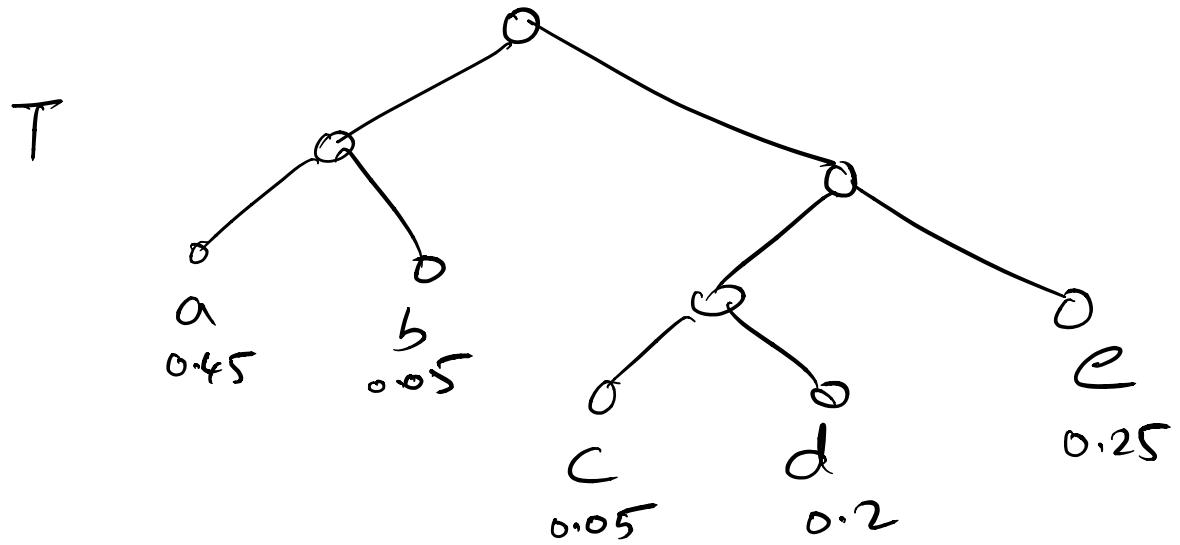
Example: $S = \{a, b, c, d, e\}$.

$$f_a = 0.45 \quad f_b = 0.05$$

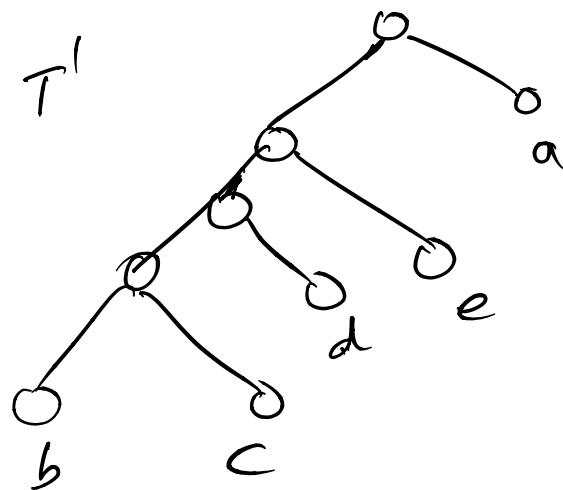
$$f_c = 0.05 \quad f_d = 0.2$$

$$f_e = 0.25$$





$$ABL(T) = 2.25$$



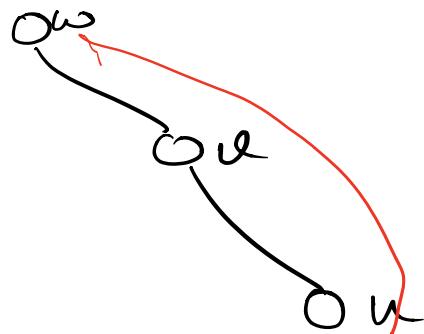
$$ABL(T') = 1.95$$

Huffman codes.

Let T be an optimal prefix code.

Obs T is full. (i.e each non-leaf node has two children)

Pf: Use exchange argument.



[see book]

Obs: Let x, y be leaf nodes in T .

If $f_x < f_y$, then

$\underset{T}{\text{depth}}(x) \geq \underset{T}{\text{depth}}(y)$.

Pf: Exchange argument.

$$T \rightarrow T'$$

$$\text{ALC}(T) - \text{ALC}(T')$$

→ Suppose you are given
an optimal tree without
labels.

Task : Find labelling.

→ Sort according frequencies

→ Label least frequent
symbols with larger

depth.

Algorithm

(i) If S has 2 letters
encode each letter using
1 bit.

(ii) Else

Let u, w be the
least frequent letters in
 S .

$$S' = (S \setminus \{u, w\}) \cup \{\alpha\}$$

\uparrow
new letter

$$f_\alpha = f_u + f_w$$

(iii) Recursively solve on
 s^1 to get tree T^1 .

(iv) Construct tree T for
 T^1 by : removing the
label α from the leaf,
add're a left and
right child , labelling
u and w .

endif .