Huffman Codes.

A-Z } 52
a-z }
0-9 } 10
} 62 + 10 = 72
punctuation.

n ↑

Quiz Syllabus
→ Until MSTs
10:15
11:15

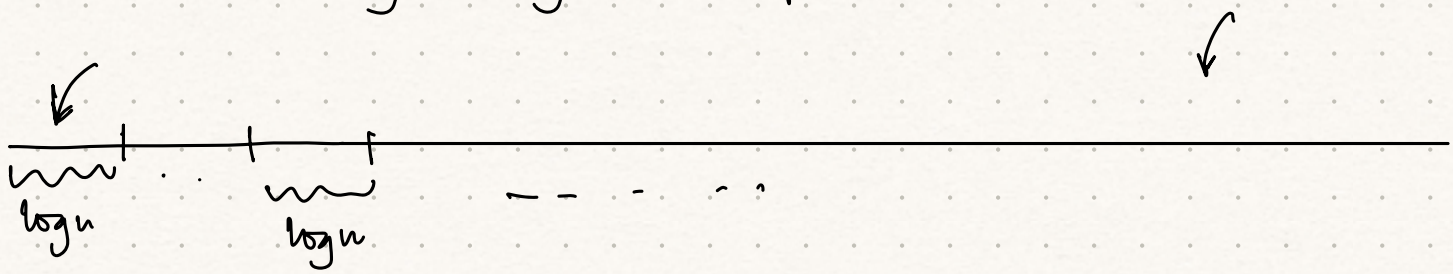
With $\lceil \log n \rceil$ many bits, we can represent n distinct letters

→ English language.
e z, x

(1 000 000) × $\lceil \log n \rceil$

Want an encoding s.t for any pair of letter u, v,
Enc(u) is not a prefix of Enc(v) }
Enc(v) " Enc(u) } .

Trivial Encoding using bin repr.



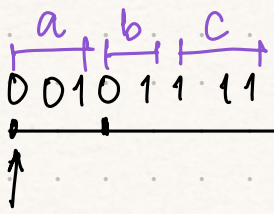
"Algorithm" → Enc(A) · Enc(l) · Enc(g) ... · Enc(m)
Binary string.

Here the length of Enc for all letters is the same.

Suppose, we move to a scheme where the lengths of Enc are different. Then, "not prefix" property helps.

Enc(u) is not a prefix of Enc(v) }
Enc(v) " Enc(u) } .

a = 001
b = 01
c = 111



d = 011

Freq ~ Prob

Letters	a	b	c	d	e
Freq	.32	.25	.2	.18	.05

In other words come up with a lookup table s.t ABL is optimized.

Task: Construct a scheme of Encoding s.t.

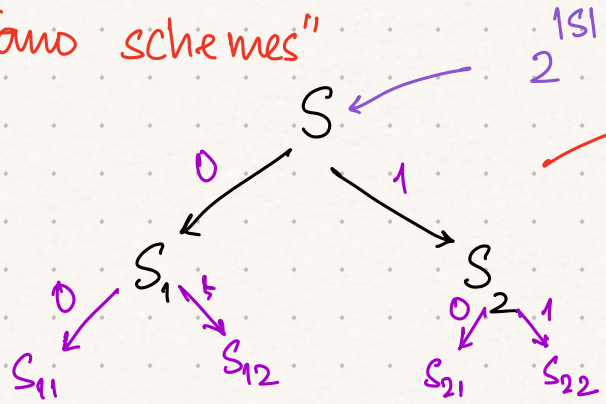
1. Optimize the average bit length (ABL)
2. "not prefix" property holds.

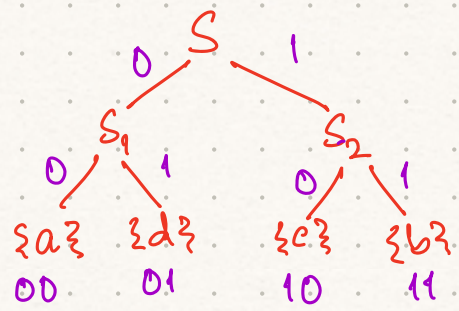
$$ABL(\text{text}) = \sum_{\text{letter} \in \text{text}} \frac{(\#\text{ of occurances of letter}) \cdot |Enc(\text{letter})|}{\text{total length of text}}$$

Want to optimize ABL

$$= \sum_{\text{letter} \in \text{text}} \text{freq.(letter)} \cdot |Enc(\text{letter})|$$

"Shannon-Fano schemes"





s.t Total freq of S_1 ≃ Total freq of S_2

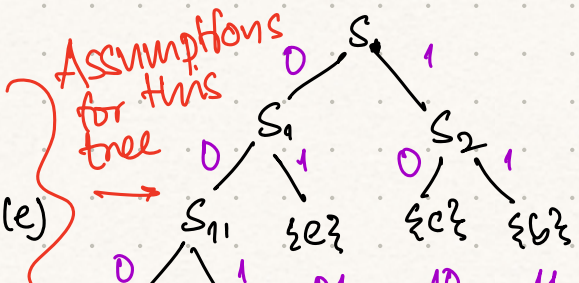
Would give $\log n$ representation if we have a complete binary tree. That

- If we cannot find an equipartition.
- If there is a equipartition, searching for S_1 and S_2 could be tiresome.

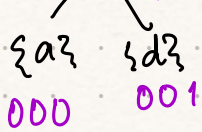
$$\text{freq}(a) + \text{freq}(d) \simeq \text{freq}(e)$$

$$\text{freq}(c) + \text{freq}(b) \simeq \text{freq}(a) + \text{freq}(d) + \text{freq}(e)$$

Assumptions for this tree →



freq(a) ≈ freq(d).



- We are labeling the leaves with letters and not the internal nodes.
- Root to leaf path gives the Enc of the letter assoc. w/ the leaf.

→ Ensure "not prefix" property.

Huffman encoding:

S: Set of letters

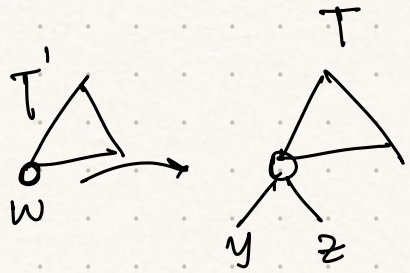
P: List of freq of letters in S.

Fundamental idea:
- Least freq elements must be placed far away from the root.
- Most freq must be closer to the root.

Algo (S, P): ← Handle base cases

- Pick two least freq elements in S (say 'y' and 'z')
- Remove 'y' and 'z' from S and P.
- Add a new letter w to S s.t

$$freq(w) = freq(y) + freq(z)$$



- Call updated lists S' and P'
- Tree T' ← Algo (S', P')
- Obtain a tree T by replacing the leaf w by a node with children 'y' and 'z'.

Return T

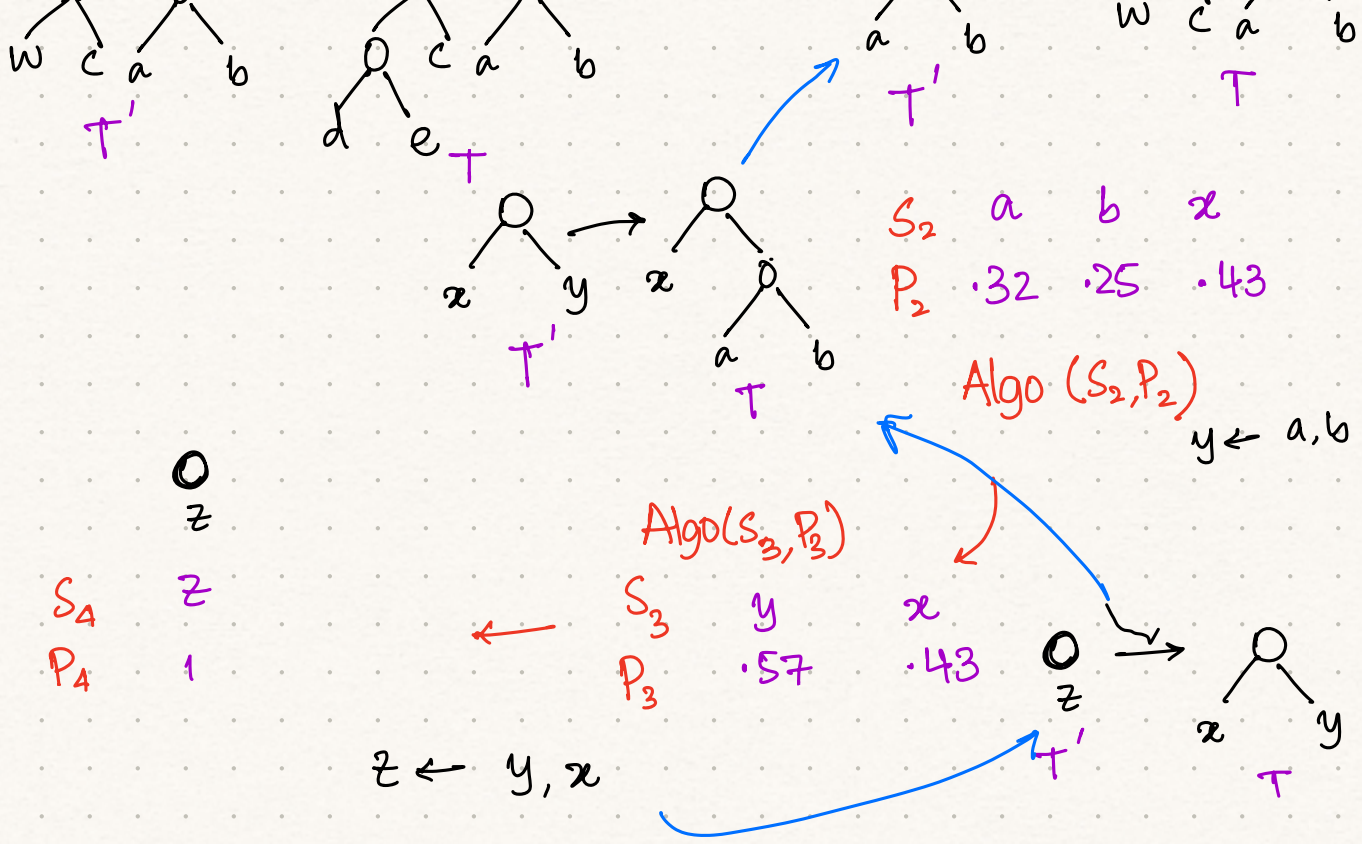
S	a	b	c	d	e
P	.32	.25	.2	.18	.05

→

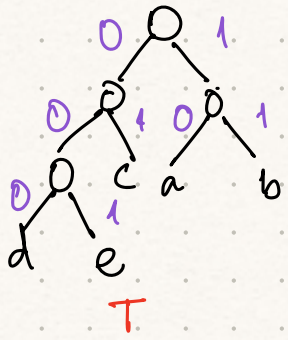
S_1	a	b	c	w
P_1	.32	.25	.2	.23

Algo(S, P)

Algo(S_1, P_1)



S	a	b	c	d	e
P	.32	.25	.2	.18	.05



a — 10
b — 11
c — 01
d — 000
e — 001

$2 \times .32 + 2 \times .25 + 2 \times .2 + 3 \times .18 + 3 \times 0.05 = 2.23$ ← ABL_T

Better than 3

Huffman encoding is optimal

Alphabet.
S ← Set of letters
P ← freq list.

→ The tree and the corresponding encoding is optimal.

$ABL_{HE} \leq ABL_T$ $\forall T$ that follows "not prefix" property

- Proof by induction on $|S|$.

Base case: $|S|=1$ Trivial.

I.H: True for $\forall S$ s.t $|S| \le k-1$.

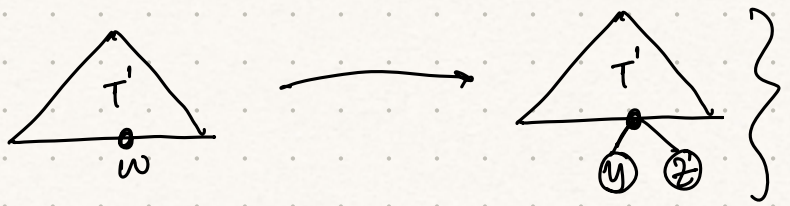
Induction Step: For the sake of contradiction, Huffman encoding gives a tree T but $\exists\ Z$ s.t $ABL_Z < ABL_T$. — (*)

($|S|=k$)

- Picks 2 least freq elements y and z and replaces them in S with a new elem 'w'.

$$S' \leftarrow (S \setminus \{y,z\}) \cup \{w\}. \qquad |S'| = k-1.$$

- From ind. hyp., we get an optimal tree T'.



$depth_T(y) = depth_T(z)$

$freq(w) = freq(y) + freq(z)$

$$ABL_T(S) = \sum_{u \in S} freq(u) \cdot depth_T(u)$$

$$= \sum_{u \in S \setminus \{y,z\}} freq(u) \cdot depth_T(u) + \big(freq(y) + freq(z)\big) \cdot depth_T(y).$$

($S \setminus \{y,z\} = S' \setminus \{w\}$)

$$= \sum_{u \in S' \setminus \{w\}} freq(u) \cdot depth_T(u) + freq(w) \cdot \big(depth_T(w) + 1\big)$$

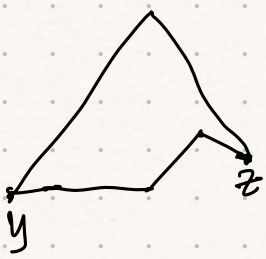
- $\forall u \in S \setminus \{y,z\}$, $depth_T(u) = depth_{T'}(u)$
- $depth_T(y) = depth_{T'}(w) + 1$

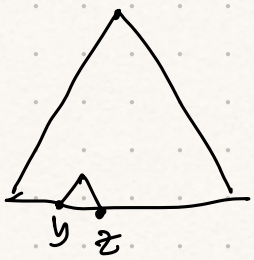
$$= \left(\sum_{u \in S'} freq(u) \cdot depth(u)\right) + freq(w)$$

$$= ABL_{T'}(S') + freq(w) \quad — (1).$$

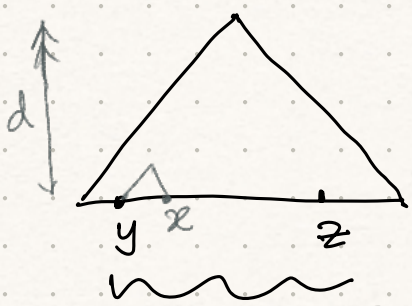
Let us consider Z.

Qu: Where do you think 'y' and 'z' are, in tree Z?





- Every leaf has a sibling



Rest + freq(x)·d + freq(y)·d + freq(z)·d

Subtle exchange / swap argument.

W.L.O.G. assume that 'y' and 'z' are siblings in the tree Z.

Obtain Z' by replacing parent of 'y' and 'z' with w.

$$ABL_Z(S) = ABL_{Z'}(S') + freq(w) \quad \text{---} (2)$$

$$ABL_{Z'}(S') + freq(w) = ABL_Z(S) < ABL_T(S) = ABL_{T'}(S') + freq(w)$$

$$\Downarrow$$

$$ABL_{Z'}(S') < ABL_{T'}(S').$$

Contradicts the optimality of T' w.r.t set S' obtained through I.H.