

HUFFMAN

CODE

Very popular and simple algorithm to construct a prefix-free code for a given source with a certain probability distribution for the source alphabet

Huffman code

Very Efficient



Yields code with
the shortest average
length $\Rightarrow$ optimal
code.

Let us describe this with the aid of an example.

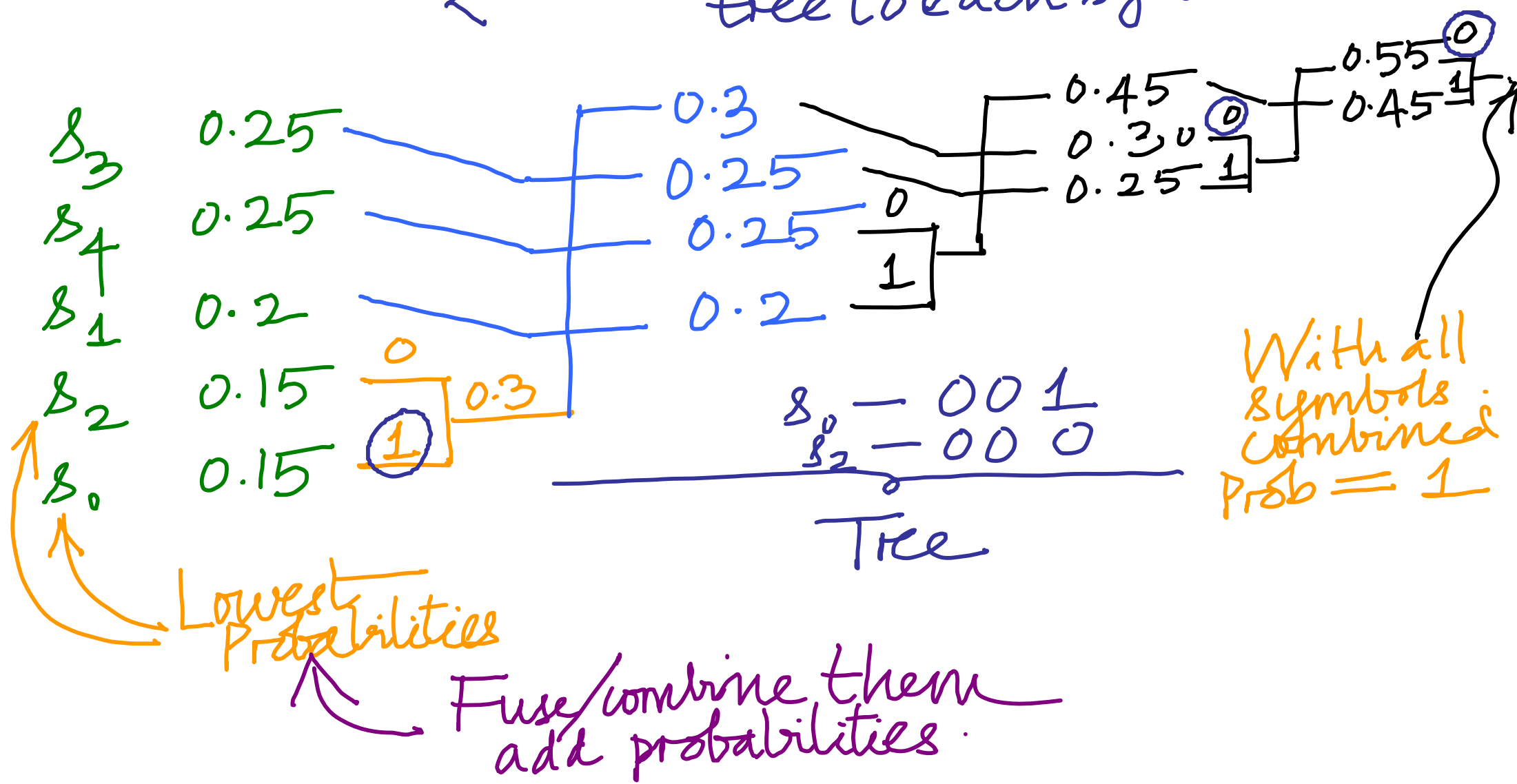
$$X \in S = \{s_0, s_1, s_2, s_3, s_4\}$$

$$\left. \begin{array}{l} P(s_0) = 0.15 \\ P(s_1) = 0.2 \\ P(s_2) = 0.15 \\ P(s_3) = 0.25 \\ P(s_4) = 0.25 \end{array} \right\} \text{sum} = 1$$

To construct Huffman code for given source.

Step # 1: List symbols in decreasing order of probability.

< To find code, Backtrack along tree to each symbol.

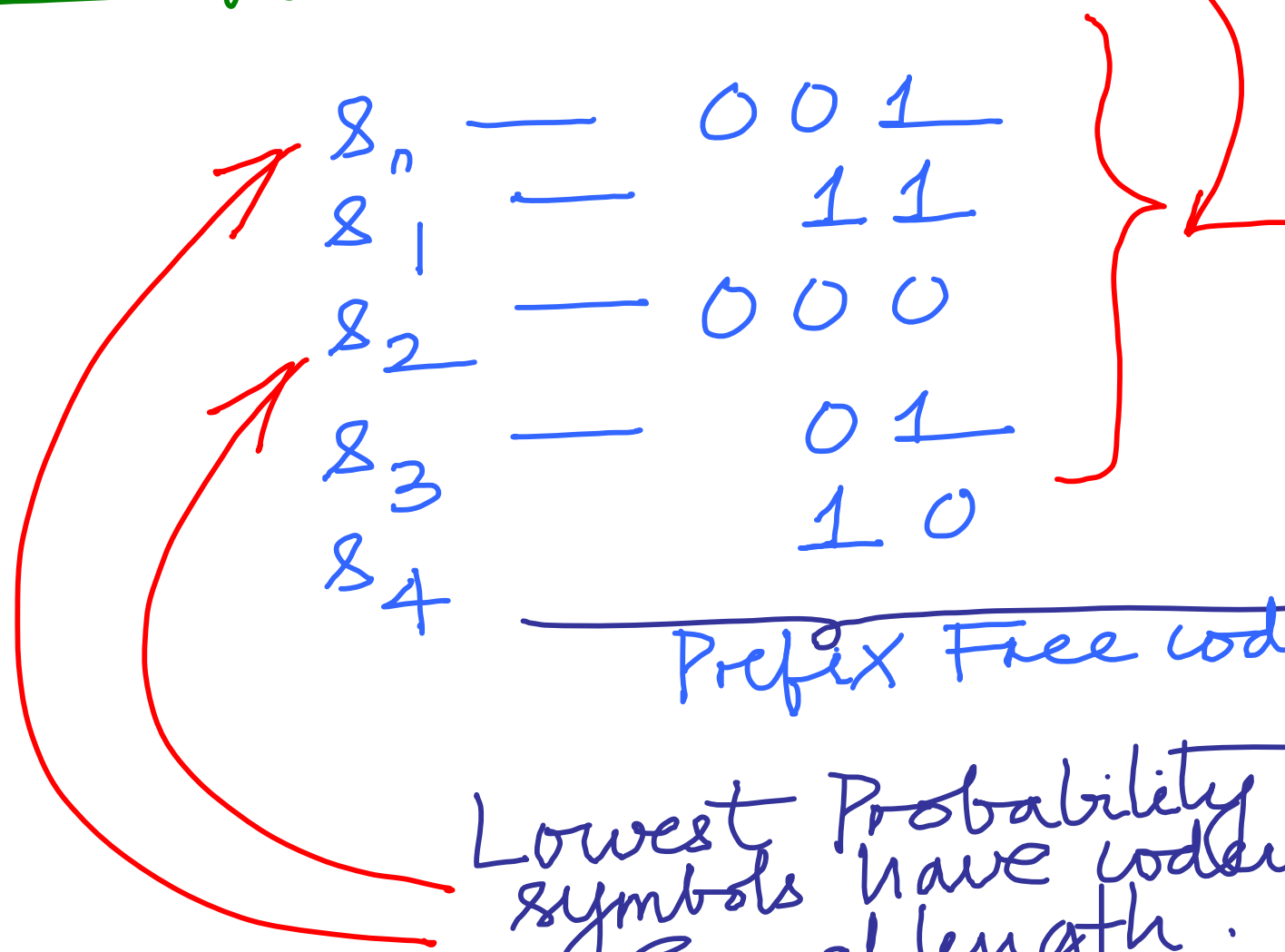


Step #2: Fuse the two symbols with the lowest probabilities & add their probabilities.

Step #3: Assign 0, 1 to the two branches.

Step #4: Reorder probabilities and start from step #2.

Huffman Code:



s_0	—	001
s_1	—	11
s_2	—	000
s_3	—	01
s_4		10

Prefix Free code.

Lowest Probability
symbols have codewords
of Equal length.



No symbol is an ancestor of any other symbol.

$\Rightarrow$ Prefix-Free code.

Average Code Length :

	code	Length	Prob
s_0	001	3	0.15
s_1	11	2	0.2
s_2	000	3	0.15
s_3	01	2	0.25
s_4	10	2	0.25

$$\begin{aligned}\text{Average code length} &= \sum_{i=0}^{M-1} p_i l_i \\ &= 3 \times 0.15 + 0.2 \times 2 + 0.15 \times 3\end{aligned}$$

$$+ 0.25 \times 2 + 0.25 \times 2$$

$$= 0.45 + 0.4 + 0.45$$

$$+ 0.5 + 0.5$$

$$L = 2.3 \text{ bits/symbol.}$$

Average Length
for Huffman code.

$$H(X) = 2 \times 0.15 \times \log_2 \left(\frac{1}{0.15} \right)$$

$$+ 0.2 \times \log_2 \left(\frac{1}{0.2} \right)$$

$$+ 2 \times 0.25 \times \log_2 \left(\frac{1}{0.25} \right)$$

$$H(X) = 2.2855$$

$$L - H(X) = 2.3 - 2.2855$$

$= 0.0145$ bits/sym

0.0145 bits/symbol from lower bound.

Huffman code very efficient

By coding over larger and larger block lengths achieve average code length arbitrarily close to Entropy.