Source coding / lossless compression

Georgios Tziritas
Computer Science Department
http://www.csd.uoc.gr/~tziritas

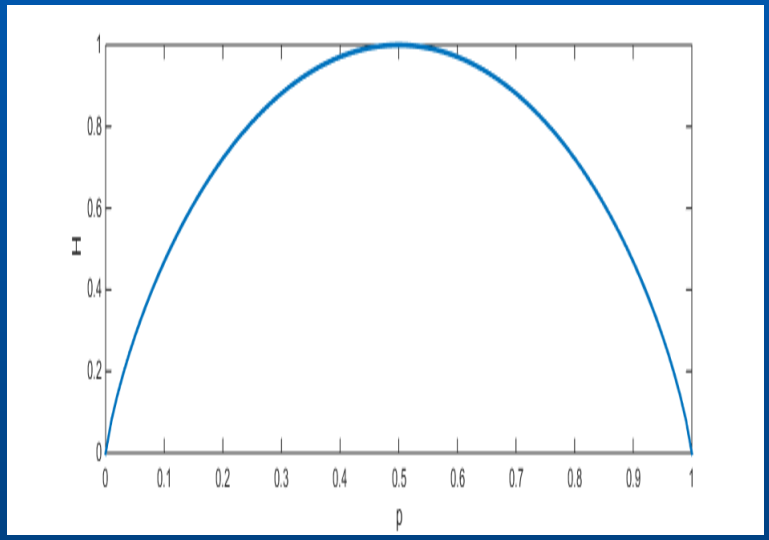
Information coding

Compression: the process of coding that will effectively reduce the total number of bits needed to represent certain information.
If the compression process induce no information loss,
then the compression scheme is **lossless**; otherwise, it is **lossy**.

Information source

Entropy of information source

$$H(S) = -\sum_{n=1}^{N} p_n \log_2 p_n$$



average amount of information contained per symbol in the source *S*

The entropy *H(S)* specifies the lower bound for the average number of bits to code each symbol in *S*

$\bar{l} \geq H(S)$ the **best** we can do

Often-occuring symbols : short codewords

Optimal code Huffman

Prefix free

Properties of the optimal binary code

- More frequently occuring symbols have shorter codewords
  ($p_n > p_m$, $l_n <= l_m$)
- The two symbols with least probabilities will have the same length for their optimal codes, differing only at the last bit.

Huffman coding algorithm

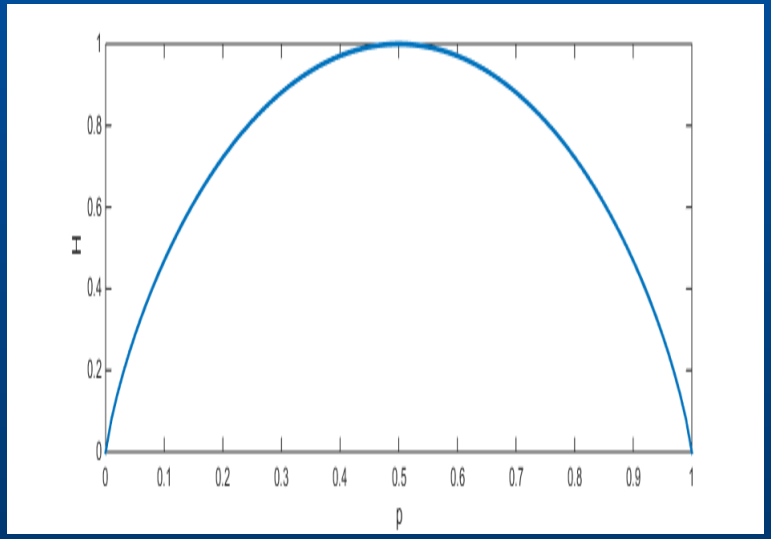
1. Initialization: Put all symbols on a list sorted according to their probability
2. Repeat until the list has only one symbol left:
    1) From the list pick two symbols with the lowest probabilities. Form a Huffman subtree that has these two symbols as child nodes and create a parent node.
    2) Assign the sum of the children's probability to the parent and insert it into the list such that the order is maintained.
    3) Delete the children from the list.
3. Assign a codeword for each leaf based on the path from the root.

Run length encoding (RLE)

Low number of symbols (ex. binary alphabet)
Large probability difference or/and strong correlations

$$\{1 - p, (1 - p)p, \ldots, (1 - p)p^m, \ldots, (1 - p)p^{M-1}, p^M\}$$

Rationale for RLC:
if the information source has the property
that symbols tend to form continuous groups,
then such symbol and the
length of the group can be coded.



Arithmetic coding

Arithmetic coding can treat the whole message as one unit.
Low number of symbols (ex. binary alphabet)
Large probability difference or/and strong correlations
Messages : intervals of [0,1)

Message	Probability	Interval
AA	0.81	[0, 0.81)
AB	0.09	[0.81, 0.9)
BA	0.09	[0.9, 0.99)
BB	0.01	[0.99, 1)

$$F = \sum_{i=1}^{K} b_i 2^{-i} \qquad K >= -\log_2 p + 1 \qquad \text{integer}$$

Arithmetic coding algorithm

```
BEGIN
    low = 0.0;  high = 1.0;  range = 1.0;
    while (symbol != terminator)
    {
        get (symbol);
        low = low + range * Range_low(symbol);
        high = low + range * Range_high(symbol);
        range = high - low;
    }
    output a code so that low <= code < high;
END
```

Lempel-Ziv coding (1977)

String matching

$X(1,L) = X(-m,L-1-m), \quad 0 <= m <= W-1$

Coding *L* (variable length code word) and *m* (fixed length code word)

βββαααααααααααβ . . .
W=4, L=9

Portable Network Graphics

Lempel-Ziv-Welch coding

LZW uses fixed-length codewords to represent
variable-length strings of symbols/characters
that commonly occur together.

LZW places longer and longer repeated entries into a dictionary,
and then emits the *code* for an element, rather than the string itself,
if the element has already been placed in the dictionary

Graphics Interchange Format