Huffman Code

- Data Compression
  - Lossy and Lossless Compression
  - Adaptive and non-Adaptive Compression
  - Fixed and Variable length Codes
- Prefix Free Codes
  - Binary Tree Representation
  - Goodness Measure
- Generic Greedy Algorithm
- Huffman Code
- Optimality and Implementation

Imdad ullah Khan

Data Compression

Data Compression is used in many computer science areas for reduced

- Computational complexity of data processing
- Storage complexity
- Communication complexity

Compression Scheme

Let $\mathcal{D}$ be the set of all possible documents (files/input) and $\mathcal{D}'$ is set of all possible output documents

A compression scheme has two algorithms

- A compression algorithm $f : \mathcal{D} \to \mathcal{D}', \quad f(x) = y$
- A decompression algorithm $g : \mathcal{D}' \to \mathcal{D}, \quad g(y) = x'$

Lossy and Lossless compression

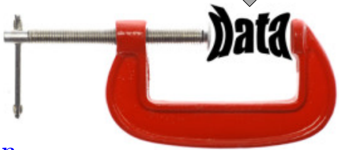
A compression scheme is

- **Lossless** if $g(y) = f^{-1}(y) = x$ for all $x \in \mathcal{D}$ such that $f(x) = y$
  - Used in Huffman code, .gif, .png
- **Lossy** if $g(y) \sim x$ for all $x \in \mathcal{D}$ such that $f(x) = y$
  - Similarity between $g(y)$ and x is measured by some error function
  - Used in .mp3, .mpg, .jpg

Data Compression

source: percona.com

data



Lossless Compression

Huffman

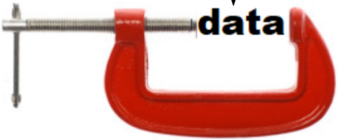
Shannon Fano

Lossy Compression

JPEG

MPEG

= original data



$\simeq$ original data

Adaptive and non-adaptive

A Compression scheme can be

- **Non-adaptive:** – Assumes prior knowledge of the data
  - e.g. '*e*' is the most common character in English language documents
  - 'the' is the most common word
- **Adaptive:** – Assumes no prior knowledge of the data
  - can build such knowledge (e.g. frequencies in the input document)
  - this knowledge will be adaptive to the actual document

Binary codes

A binary code is a compression scheme with $\mathcal{D}'$ as bit-strings

Suppose a file D is 100,000 characters long

D has only 6 unique characters (symbols)

Frequencies of each character is as follows

Characters and their frequencies in F

	a	b	c	d	e	f	Total
Frequency in 000's	45	13	12	16	9	5	100

Find a binary code that encodes D using minimum number of bits

Fixed versus Variable length codes

Fixed Length Code

- Fixed number of bits for each symbol (character)
- e.g. ASCII (7 bits) and Unicode (UTF-8, UTF-16)
- ASCII can represent $2^7 = 128$ symbols

Variable Length Code

- Variable number of bits for each symbol
- Can use fewer bits for more frequent symbols
- e.g. Huffman code
- Difficult to find, needs compression scheme

Fixed versus Variable length codes

Characters	a	b	c	d	e	f	Total
Frequency	45k	13k	12k	16k	9k	5k	100k
Fixed-Length Code	000	001	010	011	100	101	3×100k
Variable Length Code	0	101	100	111	1101	1100	224k

Variable length code uses about 25% less space

Fixed versus Variable length codes

Characters	a	b	c	d
Frequency	5	3	1	1
Fixed-Length Code	00	01	10	11
Variable Length Code	0	10	110	111

Let the string be **a a b b a a a b c d**

- Encoding under the fixed length code and it's length is

  00 00 01 01 00 00 00 01 10 11 $\rightarrow 2 \times 10 = 20$ bits

- Encoding under the variable length code and it's length is

  0 0 10 10 0 0 0 10 110 111 $\rightarrow 1 \cdot 5 + 2 \cdot 3 + 3 \cdot 1 + 3 \cdot 1 = 17$ bits