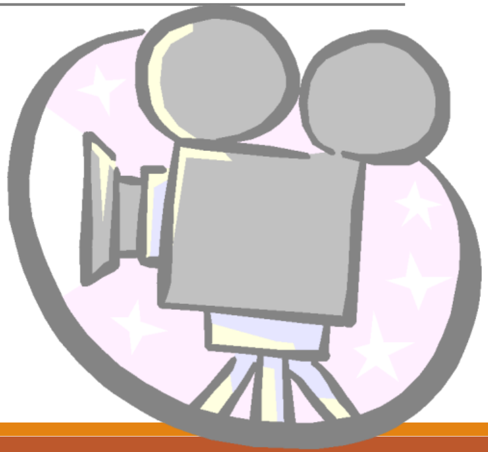
Image Processing

CS-317/CS-341



Outline

- Image Compression Fundamentals
    - Coding Redundancy
    - Interpixel Redundancy
    - Psychovisual Redundancy
    - Fidelity Criteria

Image Compression

- The term data compression refers to the process of reducing the amount of data required to represent a given quantity of information Data Information.

- Various amount of data can be used to represent the same information

- Data might contain elements that provide no relevant information : *data redundancy*

- Data redundancy is a central issue in image compression. It is not an abstract concept but mathematically quantifiable entity

For human eyes, the image will still seems to be the same even when the Compression ratio is equal 10

Human eyes are **less sensitive** to those **high frequency** signals

Our eyes will **average fine details** within the small area and record only the overall intensity of the area, which is regarded as a **lowpass** filter.

Why do We Need Compression?

- For data STORAGE and data TRANSMISSION

Images take a lot of storage space:

1024 x 1024 x 32 bits images requires 4 MB

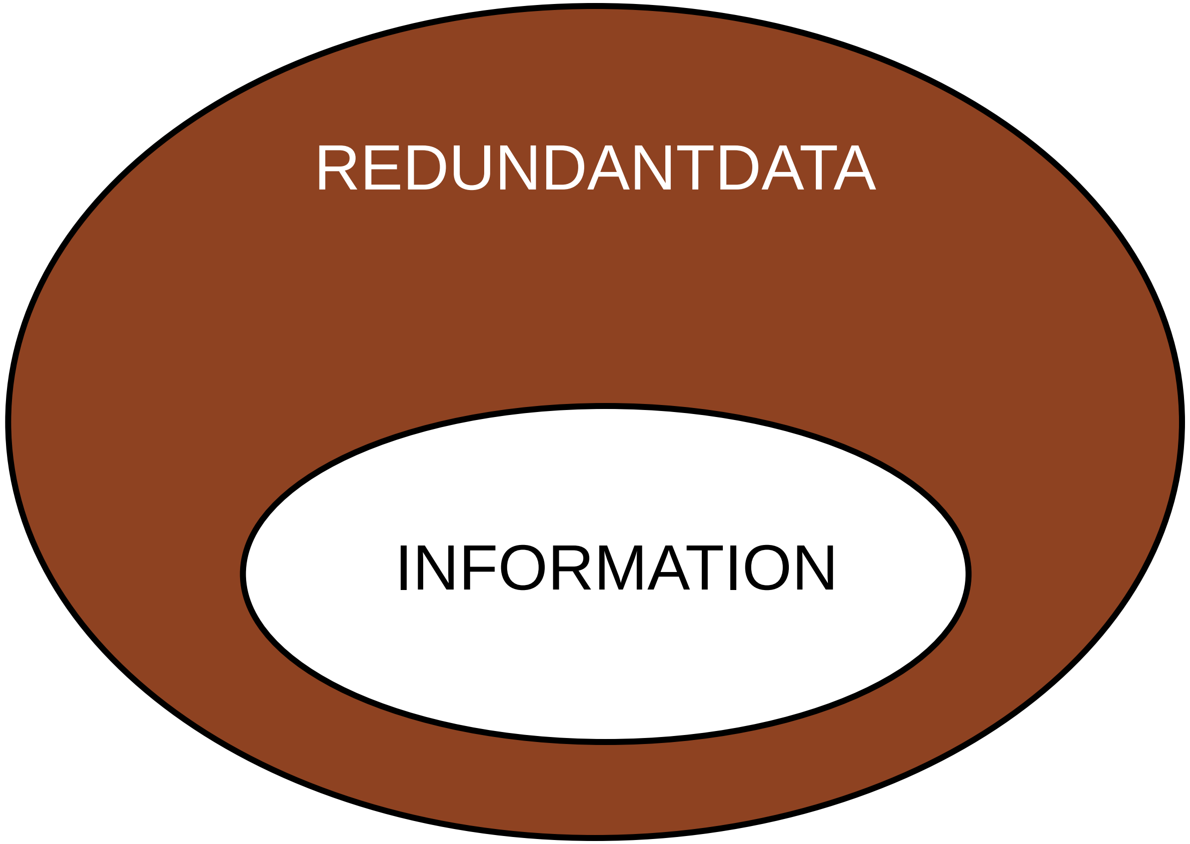
suppose you have some video that is 720 x 480 x 24 bits x 30 frames per second , 1 minute of video would require 1.54 GB

A two hour movie consists of $2.24 \times 10^{11} \approx 224$ GB of data :27 DVD of 8.5 GB are needed to store it.

Many bytes take a long time to transfer slow connections – suppose we have 56,000 bps

- 4MB will take almost 10 minutes
- 1.54 GB will take almost 66 hours

Information vs Data



DATA = INFORMATION + REDUNDANT DATA

Why Can We Compress?

Goals of compression

- Remove redundancy
- Reduce irrelevance

irrelevance or perceptual redundancy

- not all visual information is perceived by eye/brain, so throw away those that are not.

Data Redundancy

Let n_1 and n_2 denote the number of information carrying units in two data sets that represent the same information

The relative redundancy R_D is define as :

$$R_D = 1 - \frac{1}{C_R}$$

where C_R, commonly called the compression ratio, is

$$C_R = \frac{n_1}{n_2}$$

Data Redundancy

If $n_1 = n_2$, C_R=1 and R_D=0 ⟶ *no redundancy*

If $n_1 >> n_2$, $C_R \rightarrow \infty$ and $R_D \rightarrow 1$ ⟶ *high redundancy*

If $n_1 << n_2$, $C_R \rightarrow 0$ and $R_D \rightarrow -\infty$ ⟶ *undesirable*

A compression ration of 10 (10:1) means that the first data set has 10 information carrying units (say, bits) for every 1 unit in the second (compressed) data set.

Data Redundancy

$$R_D = 1 - \frac{1}{C_R}$$

If $C_R = \frac{10}{1}$, then $R_D = 1 - \frac{1}{10} = 0.9$

(90% of the data in dataset 1 is redundant)

if $n_2 = n_1$, then C_R=1, R_D=0

if $n_2 \ll n_1$, then $C_R \rightarrow \infty$, $R_D \rightarrow 1$

Fundamentals

Basic data redundancies:

1. Coding redundancy
2. Inter-pixel redundancy
3. Psycho-visual redundancy

Coding redundancy

our quantized data is represented using codewords.

if the size of the codeword is larger than is necessary to represent all quantization levels, then we have coding redundancy .

An 8-bit has 256 distinct levels of intensity in an image .

But if there are only 16 different grey levels in a image than it will require only 4 bit instead of 8 bits.

Coding redundancy can also arise due to the use of fixed-length codewords.

Example:

4	8	12
15	7	11
1	6	2

(a)

0100	1000	1100
1111	0111	1011
0001	0110	0010

(b)

100	1000	1100
1111	111	1011
1	110	10

(c)

(b) Number of bits required=3×3×4
=36 bits

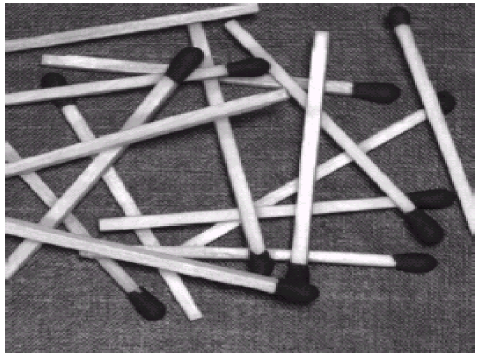
(c) Number of bits required=3×3+4×4+1+2
=9+16+1+2=29 bits

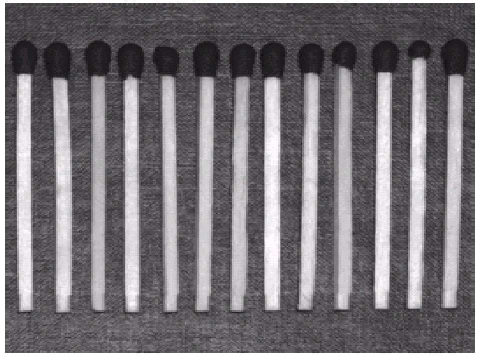
Inter-pixel Redundancy

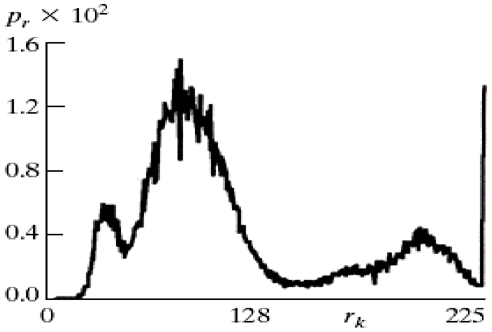
Here the two pictures have
Approximately the same
Histogram.

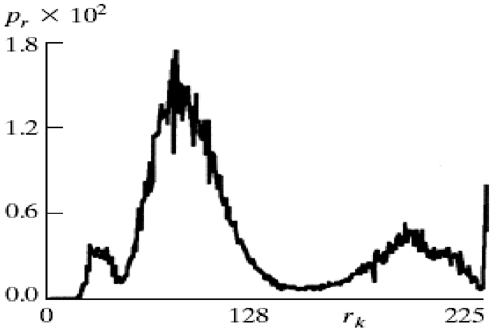
We must exploit Pixel
Dependencies.

Each pixel can be estimated
From its neighbors.









Interpixel redundancy

- Interpixel redundancy implies that any pixel value can be reasonably predicted by its neighbors (i.e., correlated).

Psychovisual redundancy

The human eye does not respond with equal sensitivity to all visual information.

It is more sensitive to the lower frequencies than to the higher frequencies in the visual spectrum.

Idea: discard data that is perceptually insignificant!

Fidelity Criteria

The general classes of criteria :

1. Objective fidelity criteria
2. Subjective fidelity criteria

Fidelity Criteria

Objective fidelity:

Level of information loss can be expressed as a function of the original and the compressed and subsequently decompressed image.

Fidelity Criteria

The error between two image of size M * N is given by:

$$e(x,y) = \hat{f}(x,y) - f(x,y)$$

So, the total error between the two images is

$$\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[\hat{f}(x,y) - f(x,y)]$$

The root-mean-square error averaged over the whole image is

$$e_{rms} = \frac{1}{MN}\sqrt{[\hat{f}(x,y) - f(x,y)]^2}$$

$$\hat{f}(x,y) - reconstructed$$

$$f(x,y) - original$$

Fidelity Criteria

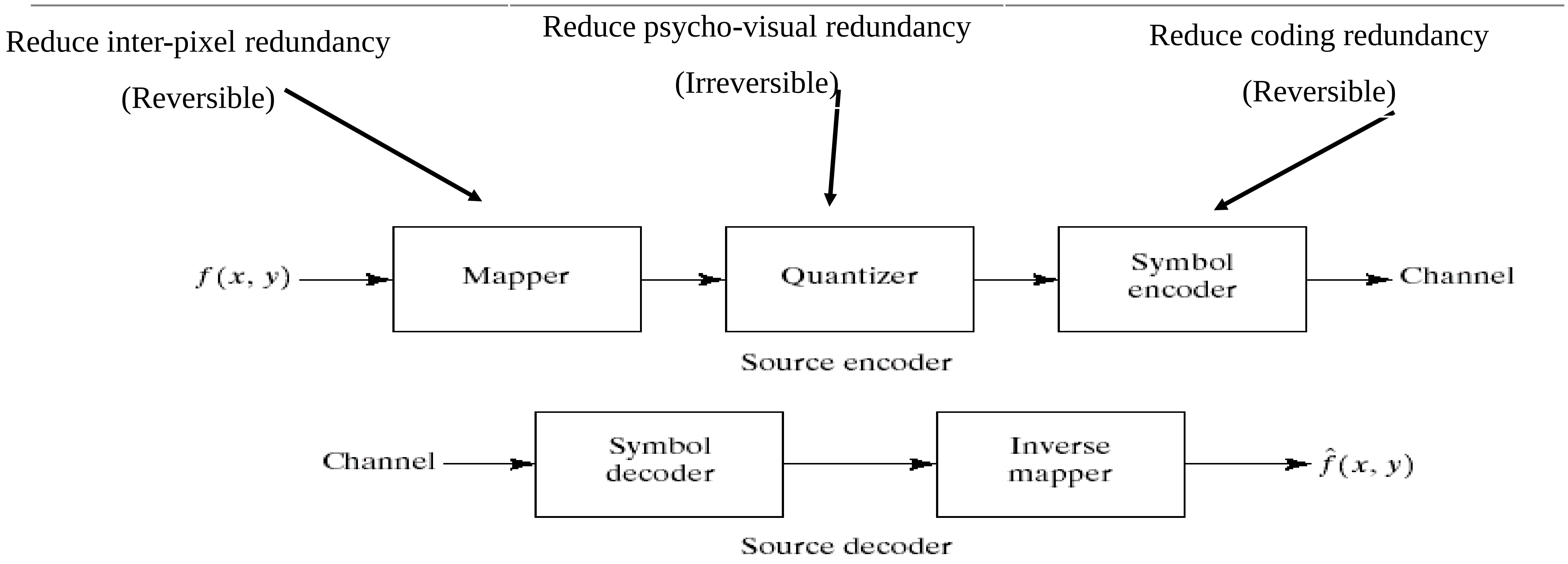
Subjective fidelity (Viewed by Human):

- By absolute rating
- By means of side-by-side comparison of $f(x, y)$ and $\hat{f}(x, y)$

TABLE 8.3
Rating scale of the Television Allocations Study Organization. (Frendendall and Behrend.)

Value	Rating	Description
1	Excellent	An image of extremely high quality, as good as you could desire.
2	Fine	An image of high quality, providing enjoyable viewing. Interference is not objectionable.
3	Passable	An image of acceptable quality. Interference is not objectionable.
4	Marginal	An image of poor quality; you wish you could improve it. Interference is somewhat objectionable.
5	Inferior	A very poor image, but you could watch it. Objectionable interference is definitely present.
6	Unusable	An image so bad that you could not watch it.

Image Compression model



(a) Source encoder and (b) source decoder model.

Classification

- Lossless compression
  - lossless compression for legal and medical documents, computer programs
  - exploit only code and inter-pixel redundancy
- Lossy compression
  - digital image and video where some errors or loss can be tolerated
  - exploit both code and inter-pixel redundancy and psycho-visual perception properties

Lossless compression

APPLICATIONS:

- MEDICAL IMAGING
- SATELLITE IMAGING
-

THEY PROVIDE: COMPRESSION RATIO OF 2 TO 10.

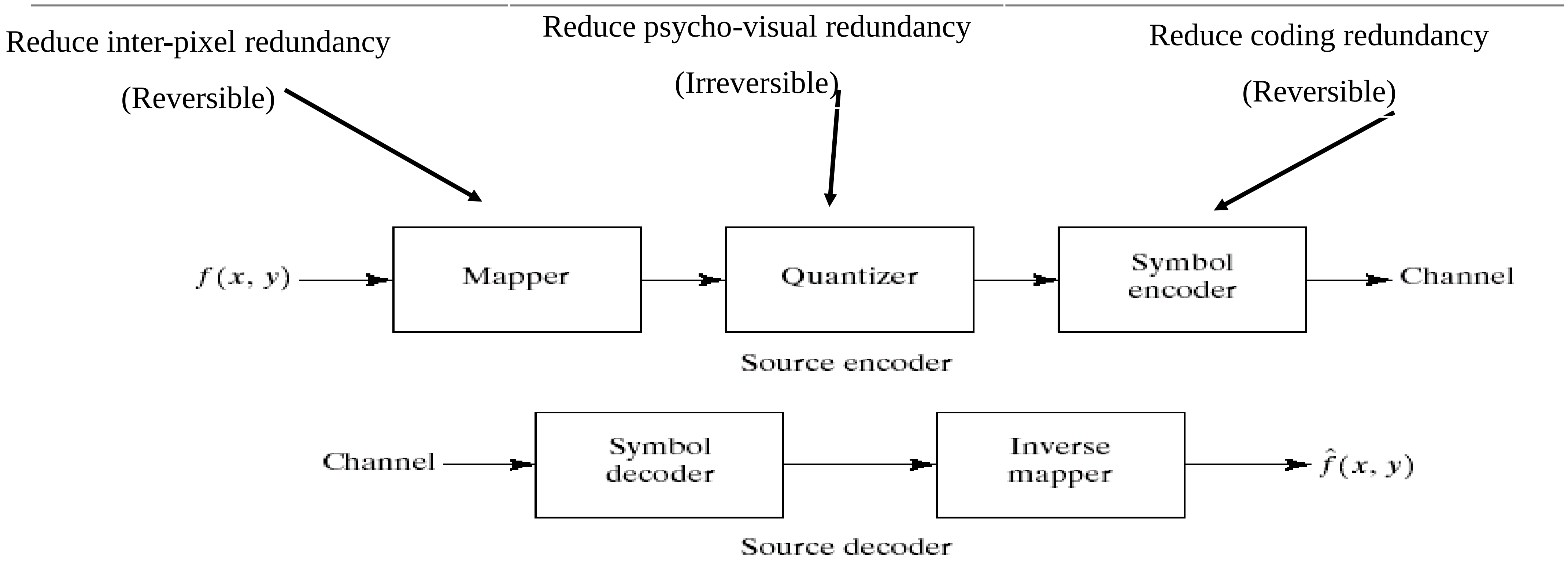
Techniques of Lossless Compression

Huffman Coding

Arithmetic Coding

Run Length Coding

Image Compression model



(a) Source encoder and (b) source decoder model.

Lossless or Error-Free Compression

Variable-length Coding

Huffman coding (optimal code)

Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

FIGURE 8.11 Huffman source reductions.

Lossless or Error-Free Compression

Variable-length Coding

Huffman coding

FIGURE 8.12 Huffman code assignment procedure.

Original source			Source reduction							
Sym.	Prob.	Code	1		2		3		4	
a_2	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
a_6	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
a_1	0.1	011	0.1	011	0.2	010	0.3	01		
a_4	0.1	0100	0.1	0100	0.1	011				
a_3	0.06	01010	0.1	0101						
a_5	0.04	01011								

$$L_{avg} = (0.4)(1) + (0.3)(2) + (0.1)(3) + (0.1)(4) + (0.06)(5) + (0.04)(5)$$

$$= 2.2\, bits / symbol$$

$$entropy = 2.14\, bits / symbol$$

Lossless or Error-Free Compression

Variable-length Coding

Huffman coding

FIGURE 8.12
Huffman code assignment procedure.

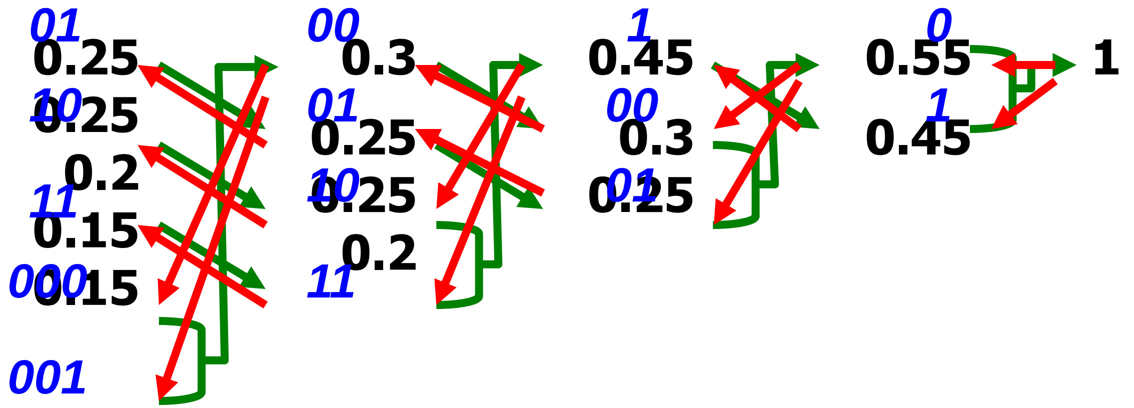
Original source			Source reduction							
Sym.	Prob.	Code	1		2		3		4	
a_2	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
a_6	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
a_1	0.1	011	0.1	011	0.2	010	0.3	01		
a_4	0.1	0100	0.1	0100	0.1	011				
a_3	0.06	01010	0.1	0101						
a_5	0.04	01011								

Example:

010100111100 = a3 a1 a2 a2 a6

Huffman Coding

Codeword length	Codeword	X	Probability
2	01	1	0.25
2	10	2	0.25
3	11	3	0.2
3	000	4	0.15
3	001	5	0.15



Lossless or Error-Free Compression

Variable-length Coding

Huffman coding

- Variable length code whose length is inversely proportional to that character's frequency
- must satisfy non-prefix property to be uniquely decodable
- two pass algorithm
  - first pass accumulates the character frequency and generate codebook
  - second pass does compression with the codebook

Lossless or Error-Free Compression

Variable-length Coding

Huffman coding

- create codes by constructing a binary tree
  - 1. consider all characters as free nodes
  - 2. assign two free nodes with lowest frequency to a parent nodes with weights equal to sum of their frequencies
  - 3. remove the two free nodes and add the newly created parent node to the list of free nodes
  - 4. repeat step2 and 3 until there is one free node left. It becomes the root of tree

Suggested Readings

- **Digital Image Processing by Rafel Gonzalez, Richard Woods, Pearson Education India, 2017.**

- **Fundamental of Digital image processing by A. K Jain, Pearson Education India, 2015.**

Thank you