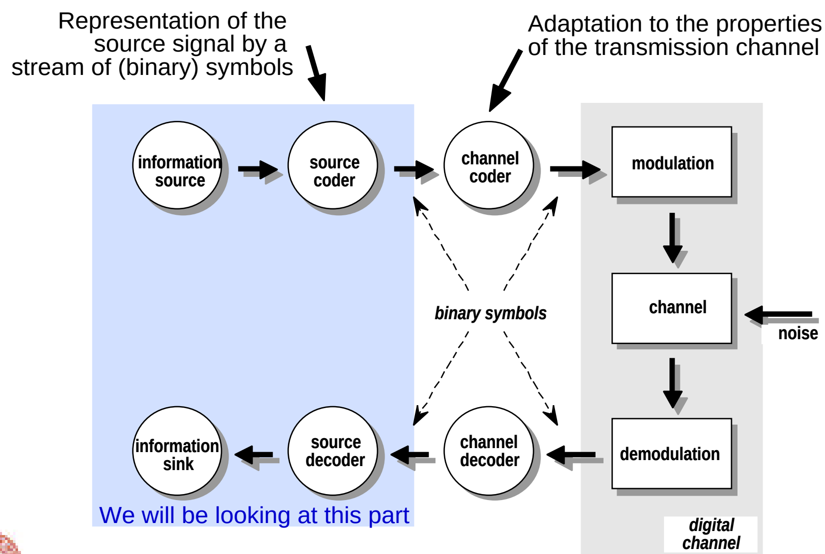
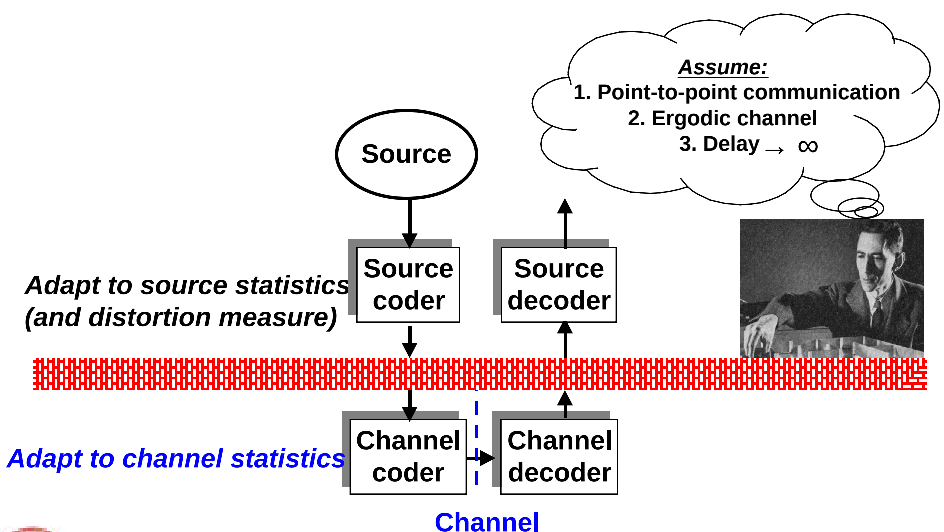
Digital communication system



Shannon's separation principle

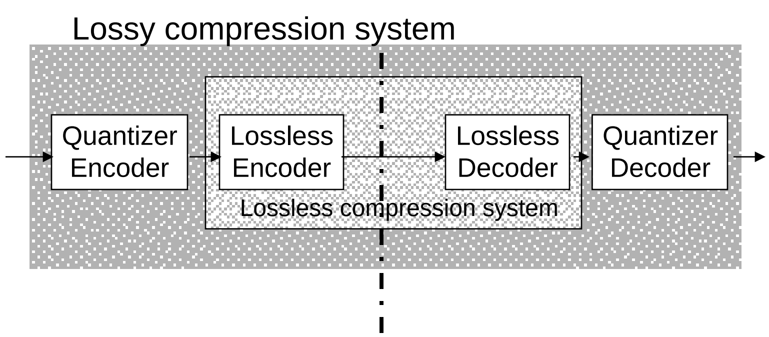


How does compression work?

- Exploit redundancy.
  - Take advantage of patterns in the signal.
  - Describe frequently occuring events efficiently.
  - Lossless coding: completely reversible
- Introduce acceptable deviations.
  - Remove information that the humans cannot perceive.
  - Match the signal resolution (in space, time, amplitude) to the application
  - Lossy coding: irreversible distortion of the signal

Lossless compression in lossy compression systems

- Almost every lossy compression system contains a lossless compression system



- We will discuss the basics of lossless compression first, then move on to lossy compression

Topics in lossless compression

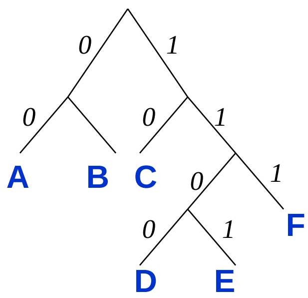
- Binary decision trees and variable length coding
- Entropy and bit-rate
- Huffman codes
- Statistical dependencies in image signals
- Sources with memory
- Arithmetic coding
- Redundancy reduction by prediction

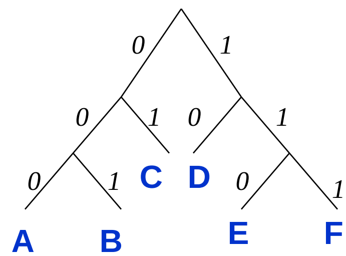
Example: 20 Questions

- *Alice* thinks of an outcome (from a finite set), but does not disclose his selection.
- *Bob* asks a series of yes-no questions to uniquely determine the outcome chosen. The goal of the game is to ask as few questions as possible <u>on average</u>.
- <u>Our goal:</u> Design the best strategy for *Bob.*

Example: 20 Questions (cont.)

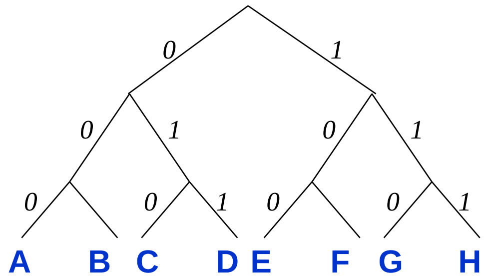
- Observation: The collection of questions and answers yield a binary code for each outcome.





- Which strategy (=code) is better?

Fixed length codes



- Average description length for K outcomes $l_{av} = \log_2 K$
- Optimum for equally likely outcomes
- Verify by modifying tree

Variable length codes

- If outcomes are NOT equally probable:
  - Use shorter descriptions for likely outcomes
  - Use longer descriptions for less likely outcomes
- Intuition:
  - Optimum balanced code trees, i.e., with equally likely outcomes, can be pruned to yield unbalanced trees with unequal probabilities.
  - The unbalanced code trees such obtained are also optimum.
  - Hence, an outcome of probability p should require about

$$\log_2\left(\frac{1}{p}\right) \text{ bits}$$

Entropy of a memoryless source

- Let a memoryless source be characterized by an ensemble U_0 with:

Alphabet $\{a_0, a_1, a_2, ..., a_{K-1}\}$
Probabilities $\{P(a_0), P(a_1), P(a_2), ..., P(a_{K-1})\}$

- Shannon: information conveyed by message "a_k":

$$I(a_k) = -\log(P(a_k))$$

- "Entropy of the source" is the average information contents:

$$H(U_0) = E\{I(a_k)\} = -\sum_{k=0}^{K-1} P(a_k)\log(P(a_k)) = -\sum_{u_0} P(u_0)\log(P(u_0))$$

- For "$\log$" = "$\log_2$" the unit is bits/symbol

Entropy and bit-rate

- Properties of entropy:

$$H(U_0) \geq 0$$
$$\max\{H(U_0)\} = \log(K) \text{ with } P(a_j) = P(a_k) \quad j,k$$

- The entropy $H(U_0)$ is a lower bound for the average word length l_{av} of a decodable variable-length code for the symbols u_0.
- Conversely, the average wordlength l_{av} can approach $H(U_0)$, if sufficiently large blocks of symbols are encoded jointly.
- Redundancy of a code:

$$R = l_{av} - H(U_0)$$

Encoding with variable word length

- A code without redundancy, i.e.

$$l_{av} = H(U_0)$$

is achieved, if all individual code word length

$$l_{cw}(a_k) = -\log(P(a_k))$$

- For binary code words, all probabilities would have to be binary fractions:

$$P(a_k) = 2^{-l_{cw}(a_k)}$$

Example

a_i	$P(a_i)$	redundant code	optimum code
a_0	0.500	00	0
a_1	0.250	01	10
a_2	0.125	10	110
a_3	0.125	11	111

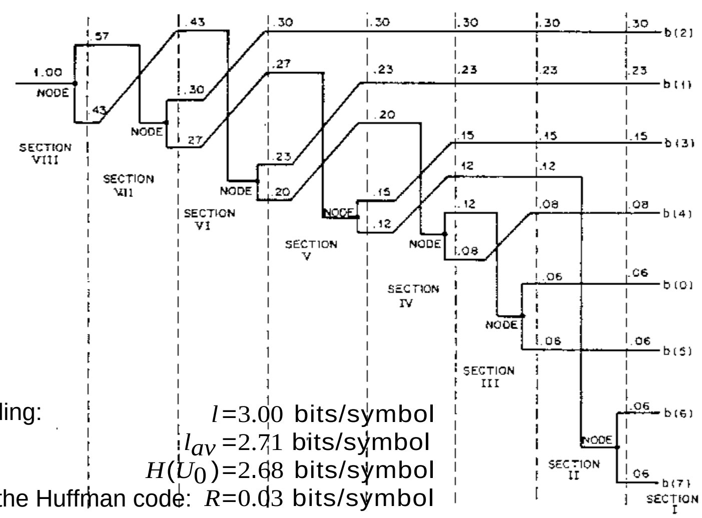
$$H(U_0) = 1.75 \text{ bits / symbol}$$
$$l_{av} = 1.75 \text{ bits / symbol}$$
$$R = 0$$

Huffman-Code

- Design algorithm for variable length codes proposed by Huffman (1952) always finds a code with minimum redundancy.
- Obtain code tree as follows:

1	Pick the two symbols with lowest probabilities and merge them into a new auxiliary symbol.
2	Calculate the probability of the auxiliary symbol.
3	If more than one symbol remains, repeat steps **1** and **2** for the new auxiliary alphabet.
4	Convert the code tree into a prefix code.

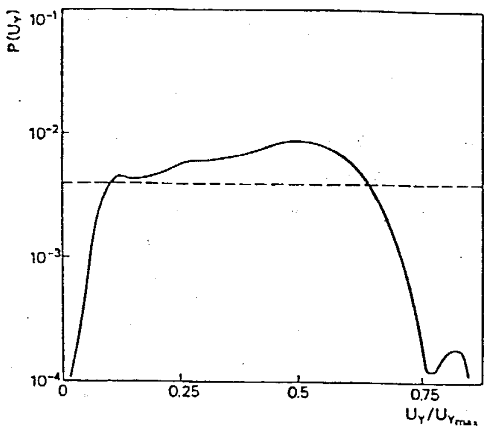
Huffman-Code - Example



Fixed length coding: l=3.00 bits/symbol
Huffman code: l_{av}=2.71 bits/symbol
Entropy $H(U_0)$=2.68 bits/symbol
Redundancy of the Huffman code: R=0.03 bits/symbol

Probability density function of the luminance signal Y

Images: 3 EBU test slides, 3 SMPTE test slides, uniform quantization with 256 levels (8 bits/pixel)



$H(U_Y) = 7.34$ bits / pixel

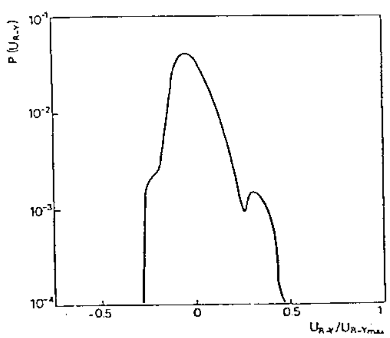
Image with lowest entropy:

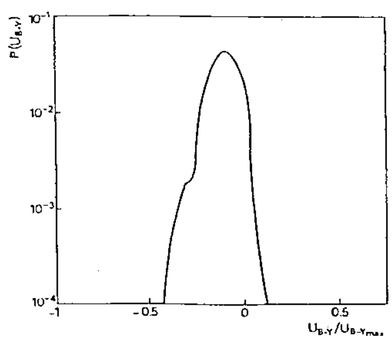
$H_L(U_Y) = 6.97$ bits / pixel

Image with highest entropy:

$H_H(U_Y) = 7.35$ bits / pixel

Probability density function of the color difference signals R-Y and B-Y





$H(U_{R-Y}) = 5.57$ bits/pixel

$H_L(U_{R-Y}) = 4.65$ bits/pixel

$H_H(U_{R-Y}) = 5.72$ bits / pixel

$H_L(U_{B-Y}) = 4.00$ bits / pixel

$H_H(U_{B-Y}) = 5.34$ bits / pixel

Joint sources

- Joint sources generate *N* symbols simultaneously. A coding gain can be achieved by encoding those symbols jointly.
- The lower bound for the average code word length is the joint entropy:

$$\boxed{\begin{aligned} &H(U_1,U_2,\ldots,U_N) \\ &= -\sum_{u_1}\sum_{u_2}\ldots\sum_{u_N} P(u_1,u_2,\ldots,u_N)\log(P(u_1,u_2,\ldots,u_N)) \end{aligned}}$$

- It generally holds that

$$H(U_1,U_2,\ldots,U_N) \le H(U_1) + H(U_2) + \ldots + H(U_N)$$

with equality, if $U_1,U_2,\ldots,U_N$ are statistically independent.

Statistical dependencies between video signal components Y, R-Y, B-Y

- Data: 3 EBU-, 3 SMPTE test slides, each component Y, R-Y, B-Y uniformly quantized to 64 levels

$H_0 = 3\text{x}6$ bits / sample	=	18	bits/sample
$H(U_Y,U_{R-Y},U_{B-Y})$	=	9.044	bits/sample
$H(U_Y)+H(U_{R-Y})+H(U_{B-Y})$	=	11.218	bits/sample
ΔH	=	2.174	bits/sample

- Statistical dependency between R, G, B is much stronger.
- If joint source Y, R-Y, B-Y is treated as a source with memory, the possible gain by joint coding is much smaller.

Markov process

- Neighboring samples of the video signal are **not** statistically independent:

"source with memory"

$$\boxed{P(u_T) \neq P(u_T \mid u_{T-1}, u_{T-2}, \ldots, u_{T-N})}$$

- A source with memory can be modeled by a Markov random process.
- Conditional probabilities of the source symbols u_T of a Markov source of order *N*:

$$P(u_T, Z_T) = P(u_T \mid u_{T-1}, u_{T-2}, \ldots, u_{T-N})$$

state of the Markov source at time *T*

Entropy of source with memory

- Markov source of order *N*: conditional entropy

$$\begin{aligned} H(U_T \mid Z_T) &= H(U_T \mid U_{T-1}, U_{T-2}, \ldots, U_{T-N}) \\ &= E\{-\log(p(U_T \mid U_{T-1}, U_{T-2}, \ldots, U_{T-N}))\} \\ &= -\sum_{u_T} \ldots \sum_{u_{T-N}} p(u_T, u_{T-1}, u_{T-2}, \ldots, u_{T-N}) \log(p(u_T \mid u_{T-1}, u_{T-2}, \ldots, u_{T-N})) \end{aligned}$$

$$\boxed{H(U_T \mid Z_T) \leq H(U_T)}$$ (equality for memoryless sources)

- Average code word length can approach $H(U_T \mid Z_T)$ e.g. with a switched Huffman code.
- Number of states for an 8-bit video signal:

N = 1	→	256 states
N = 2	→	65536 states
N = 3	→	16777216 states