Foundations of Information Processing

Information and data compression

Considerations about **information**

Consideration 1:

How information can be defined?

Consideration 2:

Does information contain the value?

Information is processed data

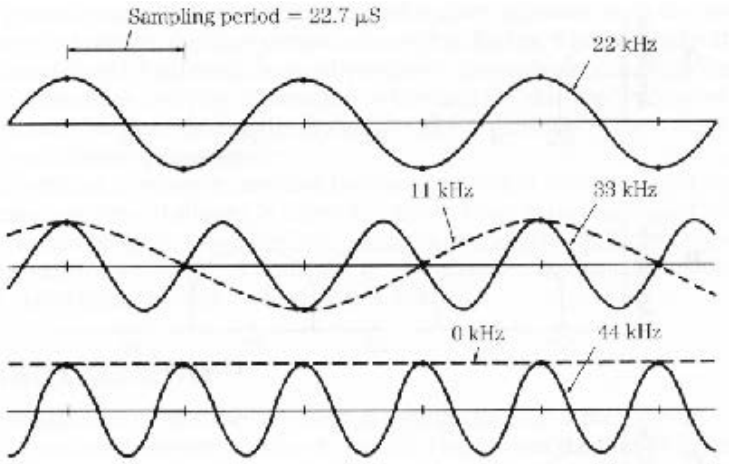
which is usually

foundation of knowledge.

Information: from analog to discrete signals

Brookshear, J.G. *Computer Science - An overview*, 7th ed. Addison Wesley, 2003

- Data vs. information vs. knowledge.
- Real-world signals: data content/information and preserving it.
- Digital (time-dependent) signals and analog-digital converters.
- Conversion of from a continuous signal to a discrete one:
  - Swedish-born American engineer of electronics Harry Nyquist (Yale U, Bell Labs) proposed for the sampling frequency $F_s > 2 \cdot F_{max}$ where $F_s/2$ is called the Nyquist frequency.
- The sampling frequency must be more than twice larger than the maximum frequency.
- Aliasing: if sampling is not frequent enough, the higher frequencies than the sampling frequency appear as the lower frequencies, i.e., wrong alias (see the figure).

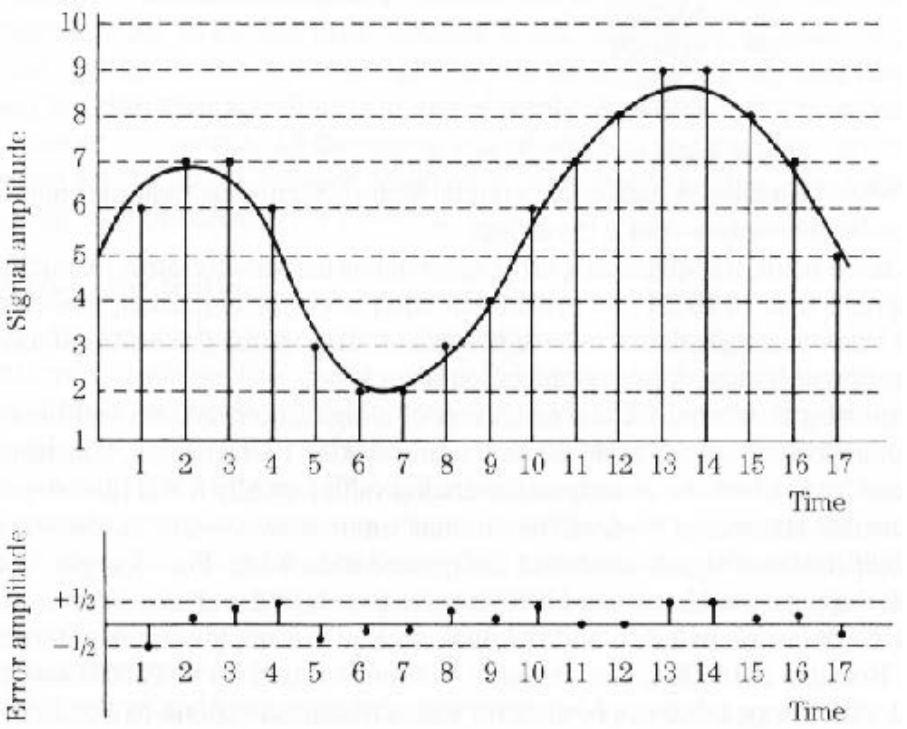


Information: quantization

Brookshear, J.G. *Computer Science - An overview*, 7th ed. Addison Wesley, 2003

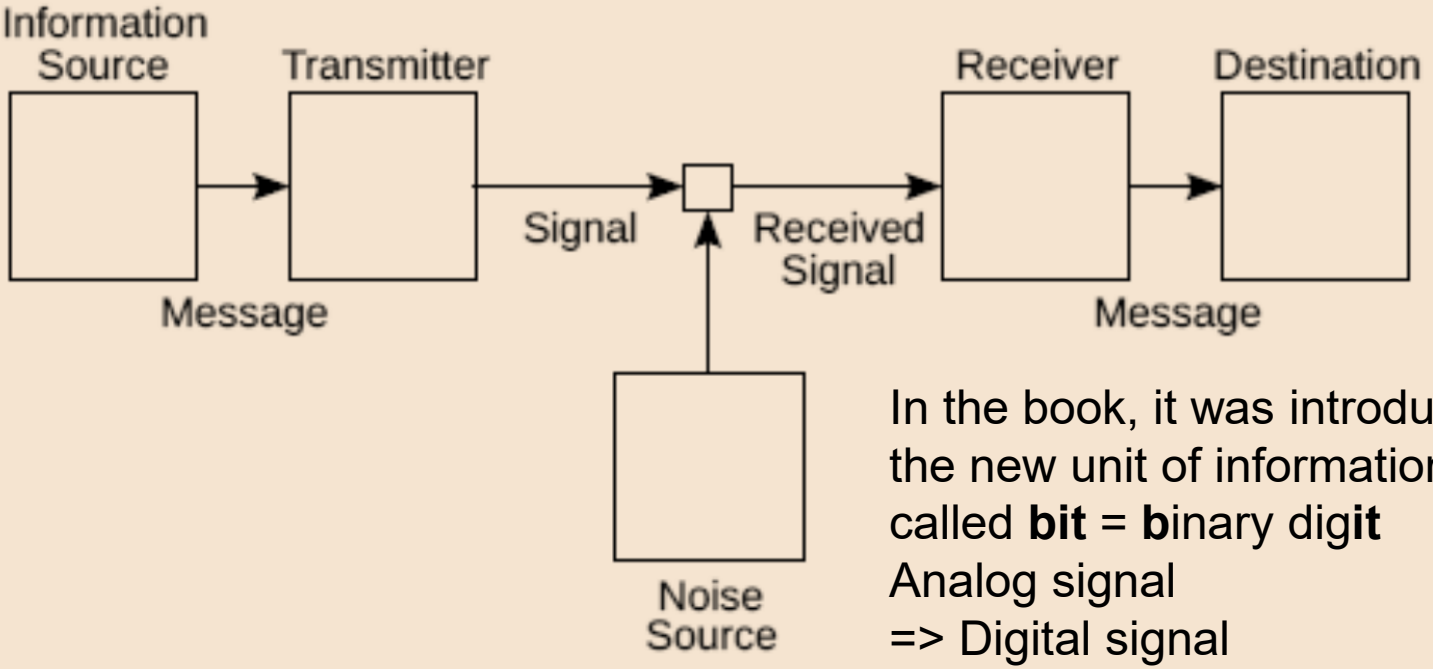
- Quantization from the continuous signal to the discrete signal.
- Quantization noise: sampling frequency and resolution of signal values.
- The difference between an input value and its quantized value (such as rounding error) is referred to as quantization error.
- This rounding from a continuous signal to a discrete signal is done by selected accuracy (resolution).
- See the figure:
  - How frequently to sample?
  - How accurately to define a discrete value?

  => quantization noise



Shannon's theory of communication

- Information:
  - Processed data: information content.
  - Probability of the result: probability of the event.
- Communication between two parties by an American mathematician and an engineer Claude Shannon, MIT, Bell Labs (the father of information theory): A Mathematical Theory of Communication, 1948, Later: A →The.
- The signal is being sent through the channel.



In the book, it was introduced the new unit of information, called **bit** = **b**inary dig**it**
Analog signal
=> Digital signal

Information content

Information content in the event A:

$$i(A) = \log_b \frac{1}{P(A)} = -\log_b P(A)$$

where

- P(A) = the probability of the event A.
- b = the base of the logarithmic function.
  - The character set of the information unit.
  - The number of characters for use/how many events.

Weather forecast: information content

Atacama Desert is an extremely dry region in Chile and Peru:

- A local record: no rain in 400 years. The whole dessert: 40 years.
- The probability that tomorrow will be dry is 100 % → *P*(dry) = 1.
- The probability of rain is then *P*(rain) = 0.
- Two possible events which one of them occurs → base =2.
- Information content that it is dry:

$$i(dry) = log_2 \frac{1}{P(dry)} = log_2 \frac{1}{1} = log_2 1 = log_2 2^0 = 0$$

Lappeenranta:

- The probability that tomorrow will be dry is 50 % → *P*(dry) = 0.5
- Two possible events which one of them occurs → base =2.
- Information content that it is dry:

$$i(dry) = log_2 \frac{1}{P(dry)} = log_2 \frac{1}{0.5} = log_2 2 = log_2 2^1 = 1$$

Entropy

- The average level of information of a random variable X in the outcomes of the events.
- The shortest average length of the message in bits to transfer information.
- Shannon entropy:
  - Fair coin flipping (heads or tails): 1 bit/toss
  - Always a head (or always a tail): 0 bit/toss
- The entropy is
  - the average number of bits

    which is needed

    to encode one symbol.
- This defines the minimum capacity for a channel for reliable (lossless) binary data transfer.

Entropy

- The weighted average of information of symbols:

$$H(X) = \sum_{i=1}^{n} p(x_i) log_b \left(\frac{1}{p(x_i)}\right)$$

$$= -\sum_{i=1}^{n} p(x_i) log_b p(x_i)$$

- Example: coin flipping

$$p(x_1) = p(x_2) = \frac{1}{2}$$

$$H(X) = -\left(\frac{1}{2} log_2 \frac{1}{2} + \frac{1}{2} log_2 \frac{1}{2}\right) = -\left(-\frac{1}{2} + -\frac{1}{2}\right) = 1.0$$

→ One bit is enough to encode: for example, heads (bit 1) and tails (bit 0).

How many bits on the average are needed for efficient representation?

- A language contains four characters: #,%,@,?
- The probabilities to appear in the language: P(#)=1/2, P(%)=1/4, P(@)=1/8, P(?)=1/8
- How many *bits* at least are needed on the average to represent one character?

$$H(X) = -\sum_{i=1}^{n} p(x_i) log_b p(x_i)$$

$$= -\left(\frac{1}{2} \cdot log_2 \frac{1}{2} + \frac{1}{4} \cdot log_2 \frac{1}{4} + \frac{1}{8} \cdot log_2 \frac{1}{8} + \frac{1}{8} \cdot log_2 \frac{1}{8}\right)$$

$$= -\left(\frac{1}{2} \cdot -1 + \frac{1}{4} \cdot -2 + \frac{1}{8} \cdot -3 + \frac{1}{8} \cdot -3\right) = 1.75 \text{ bit}$$

- How the characters should be encoded?

  → See the next slides about data compression.

Considerations about **data compression**

Consideration 1:

Does information contain redundancy?

Consideration 2:

Could information be stored
in a more compressed way
by encoding it again?

Consideration 3:

What is
the most compressed representation
for
the best encoding?

The amount of digital information is increasing very rapidly due to digitalization and image, video, and audio applications.

The representation of data can be changed by encoding it again.

By efficient encoding of information, a need of memory can be optimized.

Data compression

Brookshear, J.G. *Computer Science - An overview*, 7th ed. Addison Wesley, 2003

- Motivation for data compression.
- Lossless compression.
- Lossy compression.
- Fixed-length code.
- Variable-length code.
- Data compression methods.
- Examples, especially Huffman coding.

Motivation: why to save bits?

- Compressed data need less space in transfer and storing

  ⇒ increased speed and

  less memory capacity needed

  ⇒ more robust performance in general.
- In most cases data can be compressed without affecting information contained by data
  - Information is not lost at all (lossless),
  - or it is not lost “too” much (near lossless).
- Entropy defines the limit
  - how many bits/symbol on the average

    are needed at minimum to encode information.

Encoding strings into bits

- For each symbol in the queue (for example, for a character in a string or for a letter in text) it is selected the code word of its own bounded by the following equation:

$$\frac{N}{L} \geq \log_b K$$

N is the lenght of the code word

L is the length of the string to be encoded

b is the size of the code alphabet

K is the size of the alphabet to be encoded

- Example:

When the alphabet contains K=8 characters which are encoded to the string of the length L=4, and the code alphabet contains 0 and 1 so the size b=2, so

$$N \geq L \log_b K = 4 \log_2 8 = 4 \log_2 2^3 = 4 * 3 = 12 \text{ bits}$$

are needed for encoding.

Approaches for compression

- Compression ratio: how much encoding compresses.
- Fixed-length/variable-length code.
- Lossless compression:
  - The original source information is not lost or changed so it can be restored to its original form.
  - The compression ratio is limited.
  - What does define this limit?
- Lossy compression:
  - The original source information is partly lost so it cannot be restored to its original form.
  - The compression ratio is better.
  - Is too much information lost?

Gray code

- Frank Gray, a physicist, Purdue University, Bell Labs, the code invented in 1947, 1954 named as the Gray code.
- Binary representation where an ordering of the binary numeral system is such that two successive values differ in only one bit.
- Originally developed for electromechanical switches.
- Mainly related to error correction in digital communications where the parity is being checked.
  - The number of 1s in the string is odd => 1.
  - The number of 1s is even => 0.
  - In the Gray code neighboring codes always contain the different parity.

7	1	0	0
6	1	0	1
5	1	1	1
4	1	1	0
3	0	1	0
2	0	1	1
1	0	0	1
0	0	0	0

Run-length encoding (RLC)

- The binary representation of a string consists of queues of ones (1) and zeros (0).
- Thus, let us store runs of data as the counts of the same value in a row.
- In practice in the binary numeric system, the number of consecutive bits of the same value (1 or 0) is stored in encoding.
  - In the beginning, the first number in encoding is the count of 1s.
  - If the first bit in the string is 0 then the first number is set to 0, and then the next number is set to the count of 0s.
- Example: 1s and 0s and their counts are illustrated in colors

000110101000011111000101110011111 →

032111145311324

Fixed-length encoding

- Lossless compression.
  - Each word (or character) to be encoded has a bit pattern of its own.
  - The number of bits in the pattern is $N \geq L \log_b K$.
- Wenglish consists of 5-character words with the alphabet a-z (in total 26 different characters).
  - See http://www.inference.org.uk/mackay/itprnn/ps/65.86.pdf where the cited figure 2.1 can be found at http://www.inference.org.uk/mackay/itprnn/ps/22.40.pdf
  - Wenglish is a kind of the English language, but it contains only words of 5 characters.
- How many bits/words are needed for encoding of this language so that every word has its own bit pattern?

How many bits/word?

- The alphabet contains 26 characters (from a to z).

  => the size of the alphabet $K = 26$
- The characters are coded as words of 5 characters.

  => the length of the string to be encoded $L = 5$
- The code alphabet contains bits 1 and 0

  => The size of the code alphabet $b = 2$

$$N \geq L log_b K \Rightarrow N \geq 5 \cdot log_2 26 \Rightarrow N \geq 23.5 \text{ bits}$$

Thus, Wenglish would require 24 bits/word.

- In a computer this would require 4 bytes = long integer.
- One byte is 8 bits.
- There is no representation of 3 bytes in a computer.

Fixed-length encoding: lossy compression

- Lossy compression:
  - The own bit patters for only the most common words.
  - Rare words do not have the own bit pattern.
  - A calculated risk to lose information in order to increase data compression.
- There are 32000 typically used words in Wenglish so let us these words have individual bit patterns of their own, and the rest of words are represented with 8000 words.
  - How many bits/word with 32000+8000=40000 words?

$$N \geq log_2 40000 \Rightarrow N \geq 15.29$$

  - 16 bits/word => 2 bytes in a computer.
  - Thus, 2 bytes instead of 4 bytes as in the previous example.

Variable-length codes: lossless compression

- Huffman code:
  - David Huffman, Am. computer scientist, MIT, 1952, UC Santa Cruz.
  - Most frequent characters are encoded with shortest bit patterns.
- LZW code:
  - Lempel, Ziv, Welch, 1978/1984, Technion, Israel & MIT.
  - Most frequent words are saved in the table and given as short codes as possible.
- Q code:
  - Whole sentences are encoded with fixed-length code words.
  - QRA = What ship or coast station is that?
- Arithmetic coding:
  - Rather than separating the input into component symbols and replacing each with a code, arithmetic coding encodes the entire message into a single number, an arbitrary-precision fraction as $0.0 \leq q < 1.0$.

Huffman coding

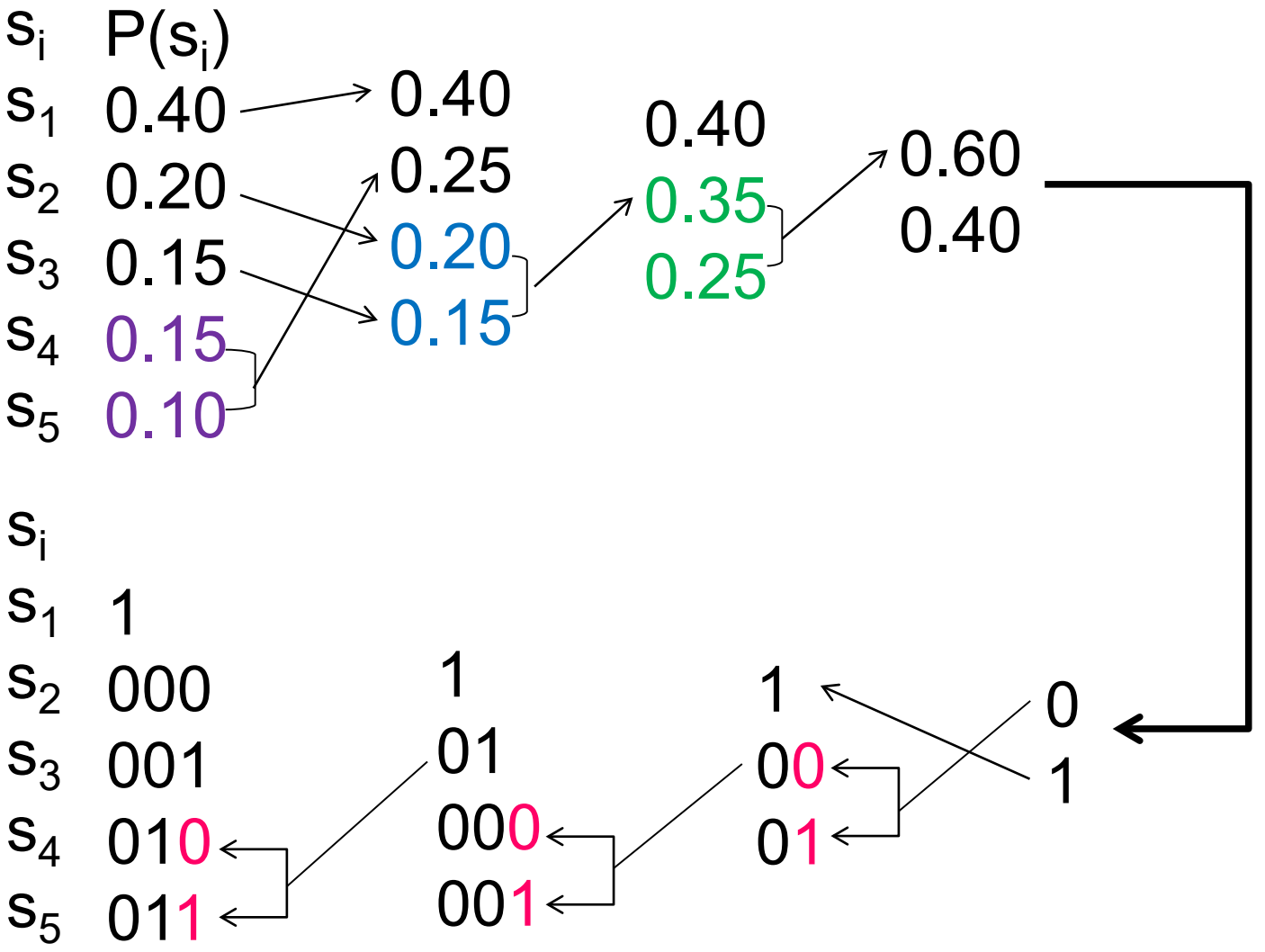
The estimated probability or frequency of occurrence for symbols must be known in order to use this coding compression algorithm:

1. Add the probabilities of the two *least probable* symbols and place the addition in the list instead of them.
2. Repeat until two probabilities are left:
   a) 0 is assigned as a code word to the one.
   b) 1 is assigned as a code word to the other one.
3. Add 0 and 1 in the right to the code word of the united probabilities.

Next, let us see examples.

Example: Huffman code

P(s_i) = the estimated probability of occurrence of the symbol s_i



Example: what is theoretically the best possible compression?

- Entropy sets the limit to optimal encoding (bits/symbol on the average):

$$H(X) = \sum_{i=1}^{n} p(x_i) log_b \left(\frac{1}{p(x_i)}\right) =$$

$$0.4 \log_2\left(\frac{1}{0.4}\right) + 0.2 \log_2\left(\frac{1}{0.2}\right) + 0.15 \lg_2\left(\frac{1}{0.15}\right) + 0.15 \log_2\left(\frac{1}{0.15}\right) + 0.10 \log_2\left(\frac{1}{0.10}\right)$$

$$= 2.1464 \text{ bits/symbol}$$

- This is the best possible compression theoretically.
- Note that without compression to directly encode 5 symbols requires 3 bits/symbol (you cannot directly encode/decode 5 symbols with 2 bits only):

s_1	000
s_2	001
s_3	010
s_4	011
s_5	100

Example: how efficient is our encoding?

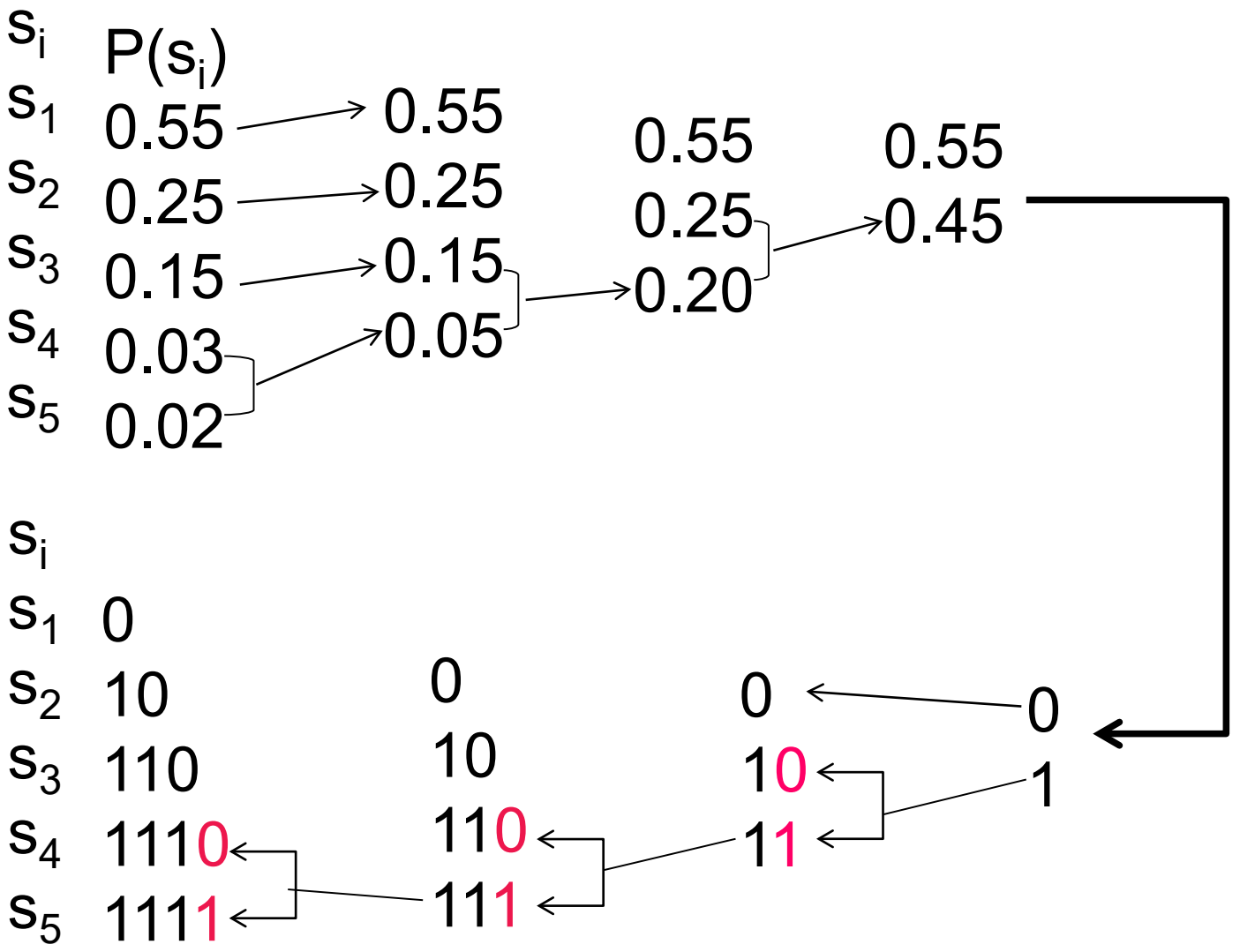
- What is the average length in bits in the message of 100 Huffman coded symbols in our example?:

$P(s_i)$		
0.40	s_1	0
0.20	s_2	000
0.15	s_3	001
0.15	s_4	010
0.10	s_5	011

- 40 times the symbol s_1 (code 0), 20 times s_2 (code 000), etc.
- The average length is $40 * 1 + (20 + 15 + 15 + 10) * 3 = 220$ bits.
- So only a little bit more than the best possible according to entropy:

  $100 * 2.1464 => 215$ bits (rounding upwards).
- Without compression: $3 * 100 = 300$ bits.

Huffman code: another example



The probabilities affect the representation of each symbol in bits = bit patterns change according to the probabilities.

Is the average length longer than in the previous example (compute)?

What happens if all symbols are as probable?
=> The algorithm does not work since it is based on probabilities.

Differential pulse-code modulation (DPCM)

The difference to the neighboring values only is stored.

187 and its neighbors are given as an example in colors.

187	187	176	199	176	187	187
187	176	199	172	199	176	187
176	199	172	172	172	199	176
199	172	172	178	172	172	199

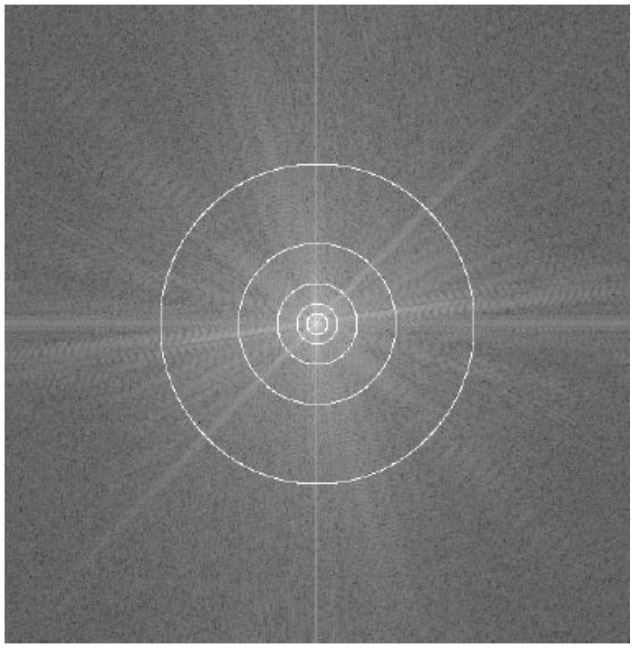
187	0	11	-23	23	-11	0
0	11	-23	27	-27	23	-11
11	-23	27	0	0	-27	23
-23	27	0	-6	6	0	-27

Lossy compression

- Motivation:
  - The better compression ratio by allowing a loss of information.
- For example, transform coding.
- Example: Discrete Fourier Transform (DFT):
  - An image transformed to the frequency domain from the spatial domain describes the image frequencies (details in the image).
  - The original image can be fully restored from the transformed image based on frequency information.
    - Trick: the less frequency information stored, the better compression ratio (and the more original information lost).
  - In practice, Discrete Cosine Transform (DCT) is applied since its efficient to implement.
  - Next, an example is shown how to compress data.

Example: Fourier Transform for compression

- An image (in the left) transformed to the frequency domain (in the right).
- The DC component is in the middle (the origin).
- The compression ratio and the loss of information depend on how much frequency values are stored:
  - The larger the circle around the origin, the less loss of information.
  - The smaller the circle, the more compression.



Summary

- Information is processed data which is usually foundation of knowledge.
- The entropy in information theory is the average number of bits which is needed to encode one symbol.
- Encoded information can be compressed (to encode again) when data elements contain redundant information.
- Compression can be lossless (no original information lost) or lossy.
- The entropy defines the lower boundary to the number of bits per data element for optimal encoding.