Honors Class (Foundations of) Informatics

Technische Universiteit
Eindhoven
University of Technology

Tom Verhoeff

Department of Mathematics & Computer Science
Software Engineering & Technology

`www.win.tue.nl/~wstomv/edu/hci`

What is Information?

$\boxed{?}$

Qualitative Definition of Information

You receive (consume) information when obtaining a (possibly partial) answer to a question; i.e., information reduces uncertainty

The *amount of information* depends on:

- The *size* of the reduction in uncertainty:

  More answers possible $\Rightarrow$ more information

  E.g.: The outcome of a *coin flip* versus a *die roll*

- The *probabilities* involved:

  Lower probability of an answer $\Rightarrow$ more information

  E.g.: The answer "No" versus "Yes" to "Will you marry . . . ?"

Anti-Information

People like/want to consume *information* (obtain more certainty)

They even are willing to pay in order to get into a situation where they can enjoy the consumption of information: *gambling*

A casino sells *anti-information* (uncertainty), and
subsequently provides *information* (certainty) as the game evolves;
once you know the outcome, the fun is over

Noise on a communication channel increases uncertainty

Shannon's Quantitative Definition of Information

Let the possible answers be A_i $(i \in S)$ with probabilities p_i satisfying

$$0 \le p_i \le 1 \text{ for all } i \in S$$
$$\sum_{i \in S} p_i = 1$$

Amount of information in answer A_i equals $\mathcal{I}(A_i) = \log_2 \frac{1}{p_i} = -\log_2 p_i$

$\mathcal{I}(A_i)$ can (also) be viewed as the amount of *surprise* in A_i

Unit of information: receiving an answer whose probability is 0.5

$p_A = \frac{1}{2}$, thus $\mathcal{I}(A) = \log_2 \frac{1}{0.5} =$ 1 bit

bit = binary digit (equiprobable choice among two options)

Properties of Shannon's Information Measure

- $\mathcal{I}(A) \to \infty$ if $p_A \to 0$ (impossible answer, never occurs)
- $\mathcal{I}(A) = 0$ (no information) if $p_A = 1$ (certainty): $-\log_2 1 = 0$
- $0 \leq \mathcal{I}(A) < \infty$ for all $0 < p_A \leq 1$
- $\mathcal{I}(A) < \mathcal{I}(B)$ if and only if $p_A > p_B$
- $\mathcal{I}(AB) \leq \mathcal{I}(A) + \mathcal{I}(B)$ ($\mathcal{I}$ is subadditive)
- $\mathcal{I}(AB) = \mathcal{I}(A) + \mathcal{I}(B)$ if A and B are *statistically independent*, where AB stands for receiving answer A followed by answer B

Entropy: Mean Amount of Information

Information Source S: repeatedly answers questions (an oracle)

Message (answer) stored in memory or communicated over a channel

Entropy $\mathcal{H}(S)$: mean (expected) amount of information per message

Examples of *stochastic* sources:

- *independent identically distributed* (i.d.d.) random variables also known as *discrete memoryless source*

  Entropy $\mathcal{H}(S) = \sum_{i \in S} p_i \mathcal{I}(i) = -\sum_{i \in S} p_i \log_2 p_i$

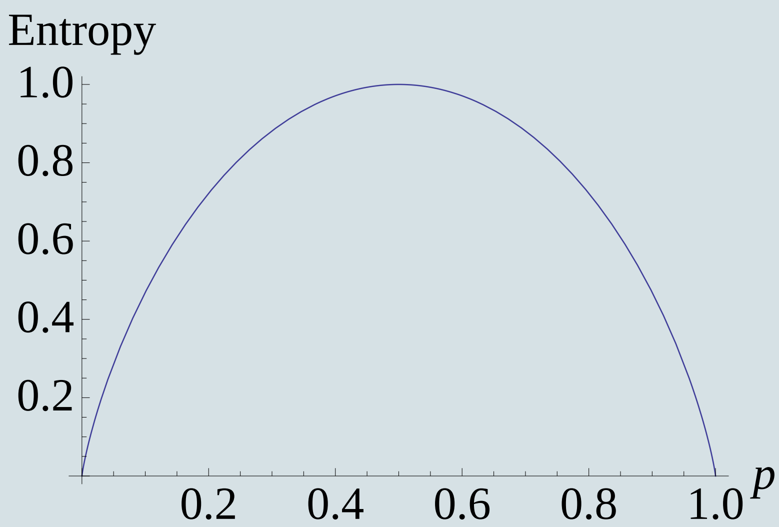
- *Markov proces* (dependencies possible); good model for language

Entropy: Example 1

Source: Two messages, with probabilities p and $1 - p = q$

Entropy $\mathcal{H}(p) = -p \log_2 p - q \log_2 q$

p	1/2	1/3	1/4	1/5	1/9	1/10
H	1	0.918296	0.811278	0.721928	0.503258	0.468996



Entropy: Example 2

Source: N messages, each with probability $p = 1/N$

Entropy $\mathcal{H}(N) = -Np \log_2 p = \log_2 N$

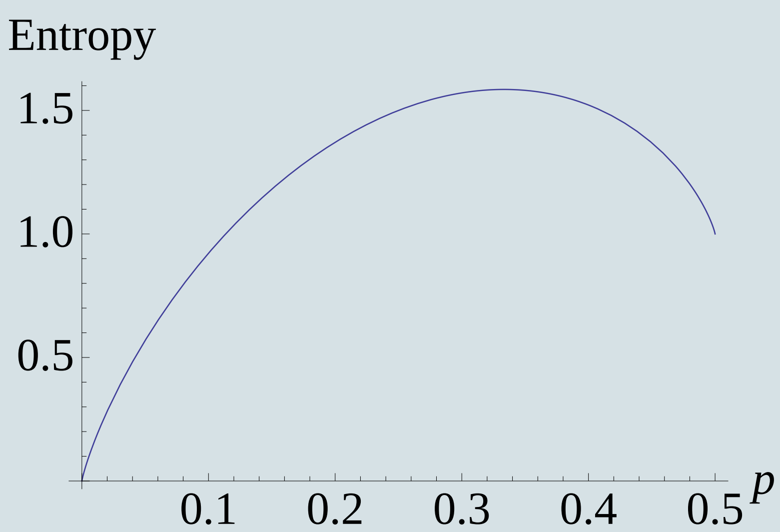
N	2	3	4	5	6	7	8
H	1	1.58496	2	2.32193	2.58496	2.80735	3

Entropy: Example 3

Source: Three messages, with probabilities p, p, and $1 - 2p = q$

Entropy $\mathcal{H}(p) = -2p \log_2 p - q \log_2 q$

p	1/3	1/4	1/5	1/6	1/7	1/8	1/9
H	1.58496	1.5	1.37095	1.25163	1.14883	1.0612	0.986427



Properties of Entropy

- Entropy bounds: $0 \leq \mathcal{H}(S) \leq \log_2 N$, for source with N messages
- $\mathcal{H}(S) = 0$ if and only if $p_A = 1$ for some message $A \in S$ (certainty)
- $\mathcal{H}(S) = \log_2 N$ if and only if $p_A = \frac{1}{N}$ for all A (max. uncertainty)

Entropy in physics measures the *amount of disorder* in a system

Source Coding Theorem

Source Coding Theorem (Shannon, 1948):

On average, each message can be encoded in $\approx \mathcal{H}$ bits, where $\mathcal{H}$ is the entropy of the message source.

More precisely:

For every $\varepsilon > 0$,

there exist *lossless encoding/decoding algorithms* such that each message is encoded in $\leq \mathcal{H} + \varepsilon$ bits on average,

and

no algorithm can achieve $< \mathcal{H}$ bits per message on average.

This theorem motivates the relevance of entropy.

Source Coding Theorem: Proof

The proof of this theorem is too involved to present here.

However, it is noteworthy that basically a ‘random’ code works.

The more messages are packed together and ‘randomly’ encoded, the better it approaches the entropy.

The engineering challenge is to find codes with practical encoding and decoding algorithms.

Source Coding Theorem: Example 1

- Two messages, A and B, each with probability 0.5 ($\mathcal{H} = 1$)

  Encode A as 0, and B as 1

  Mean number of bits per message: $0.5 \times 1 + 0.5 \times 1 = 1$

- Two messages, A and B with probabilities 0.2 and 0.8 ($\mathcal{H} = 0.72$)

  Encoding A as 0, and B as 1 gives a mean of 1 bit / message

message sequence	A	BA	BBA	BBB
probability	0.2	0.16	0.128	0.512
encode as	00	010	011	1
bits / message	2	3/2	1	1/3

$0.2 \times 2 + 0.16 \times 3/2 + 0.128 \times 1 + 0.512 \times 1/3 =$ 0.94 bits/message

Source Coding Theorem: Example 2

Three messages, A, B, and C, each with probability 1/3 ($\mathcal{H} = 1.58$)

Encode A as 00, B as 01, and C as 10: 2 bits / message

Can be improved (on average):

$3^3 = 27$ sequences of 3 messages (all with the same probability)

Encode each sequence of 3 messages in 5 bits ($2^5 = 32 \geq 27$)

Mean number of bits per message: $5/3 =$ 1.67

$3^5 = 243$ sequences of 5 messages, encode in 8 bits ($2^8 = 256 \geq 243$)

Mean number of bits per message: $8/5 =$ 1.6

Source Coding Theorem: Example 3

Three messages: A, B, C, with probabilities 1/4, 1/4, 1/2 ($\mathcal{H} = 1.5$)

Encode A as 00, B as 01, and C as 10: 2 bits / message

Can be improved (on average):

Encode A as 00, B as 01, and C as 1: 1.5 bits / message

Practical Source Coding

- For discrete memoryless source *with known probabilities*:

  Huffman Prefix Code: construct optimal encoding tree

  Possibly, apply to *blocks of messages*

- In practice, probabilities are not known and may be dependent:

  Lempel-Ziv Code: parse input, build dictionary, encode

  Dictionary contains shortest newly encountered subsequences

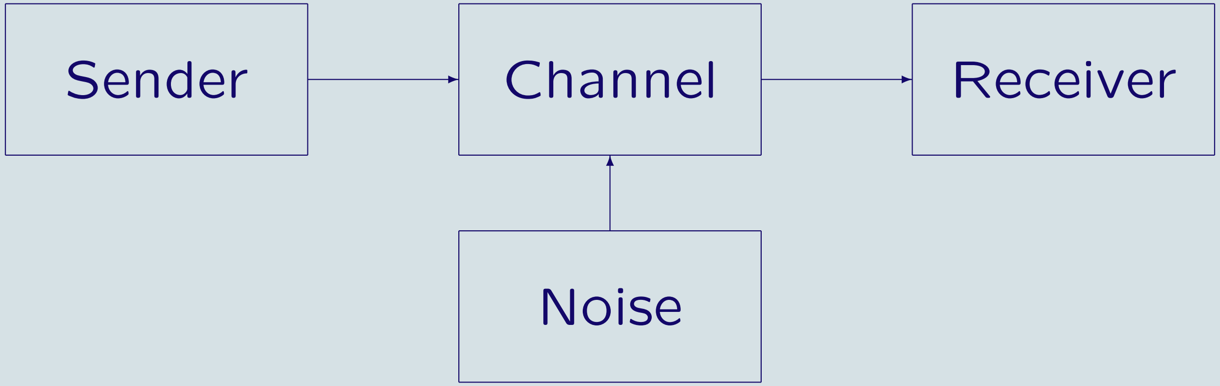
  Applied in the `zip` compression standard

Price paid for better source codes: higher latency

Noisy Channels

The capacity of a communication channel measures how many bits, on average, it can deliver reliably per transmitted bit.

A noisy channel corrupts the transmitted messages ‘randomly’.

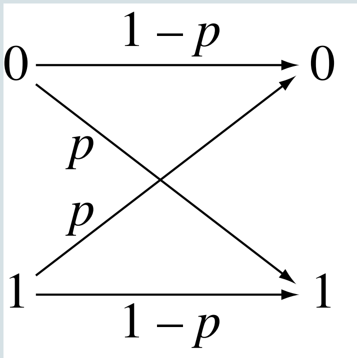


The *entropy of the noise* must be subtracted from the *raw capacity* (i.e., 1) to obtain the *effective capacity*.

Noisy-Channel Models

Some forms of noise can be modeled as a discrete memoryless source that is ‘added’ (modulo 2) to the transmitted message bits:

> A bit is transmitted erroneously (flipped) with probability p and transmitted correctly with probability $1 - p = q$.



Also known as binary symmetric channel with bit-error probability p

Other models: binary erasure, bursty bit error (cf. scratch on CD)

Noisy-Channel Example

Binary symmetric channel:

- $p = \frac{1}{2}$: $\mathcal{H}(p) = 1$, no information can be transmitted
- $p = \frac{1}{12}$: $\mathcal{H}(p) = 0.413817$, so < 0.6 bits can be transmitted

  Out of every 7 bits, $7 \times 0.413817 = 2.89672$ are 'useless', and only 4.10328 bits remain for information.

What if $p > \frac{1}{2}$?

Noisy-Channel Coding Theorem

Noisy-Channel Coding Theorem (Shannon, 1948):

Given: a channel with capacity C and a source with entropy $\mathcal{H}$.

If $\mathcal{H} < C$, then for every $\varepsilon > 0$,

there exist encoding/decoding algorithms
such that the source is transmitted with a residual error $< \varepsilon$,

and

if $\mathcal{H} > C$, then the source cannot be reproduced without a loss of at least $\mathcal{H} - C$.

This theorem motivates the relevance of channel capacity.

Noisy-Channel Coding Theorem: Proof

The proof of this theorem is, again, too involved to present here.

However, again, a ‘random’ code basically works.

The more messages are packed together and ‘randomly’ encoded, the better it approaches the capacity.

The engineering challenge is to find codes with practical encoding and decoding algorithms.

Error Control Coding

The Noisy-Channel Coding Theorem does not promise *error-free* transmission.

It only states that the *residual error* can be made as small as desired.

Idea: Use excess capacity $C - \mathcal{H}$ to transmit error-control information.
Encoding is imagined to consist of source bits and error-control bits.
Code rate = number of source bits / number of bits in encoding

Error-control information is redundant but protects against noise.

Two techniques for error control:

- Error-detecting code with feedback and retransmission
- Error-correcting code (a.k.a. forward error correction)

Error-detecting Codes

- Append a parity control bit:

  Append extra (redundant) bit making total number of 1's *even*

  Can *detect* a single bit error, but cannot *correct* it;
  code rate $= k/(k+1)$, for k source bits

  Appending a parity bit to each source bit repeats the source;
  code rate $= 1/2$

- Append a Cyclic Redundancy Check (CRC, generalized parity):

  E.g., 32 check bits computed from the source bits

  Only for *detection*, not *correction*

Error-detecting Decimal Codes in Practice

- Dutch Bank Account Number (IBAN)
- International Standard Book Number (ISBN)
- Burgerservicenummer (BSN, Dutch Citizen's Service Number)
- Student Identity Number at TU/e

These all use a single check digit (or X in ISBN-10)

International Bank Account Numbers (IBAN) use two check digits

Main goal: detect human error (single digit, or neighbor transposition)

Error-correcting Codes: Example 1

Repetition code: Repeat each source bit k times

Example: Source bits 10110 are encoded as 111000111111000

Code rate $= 1/3$ (so, this introduces a lot of redundancy)

Can correct a single bit error per encoded source bit:

Decode by *majority voting*

Cannot correct two bit errors (in that case, decoding makes it worse!)

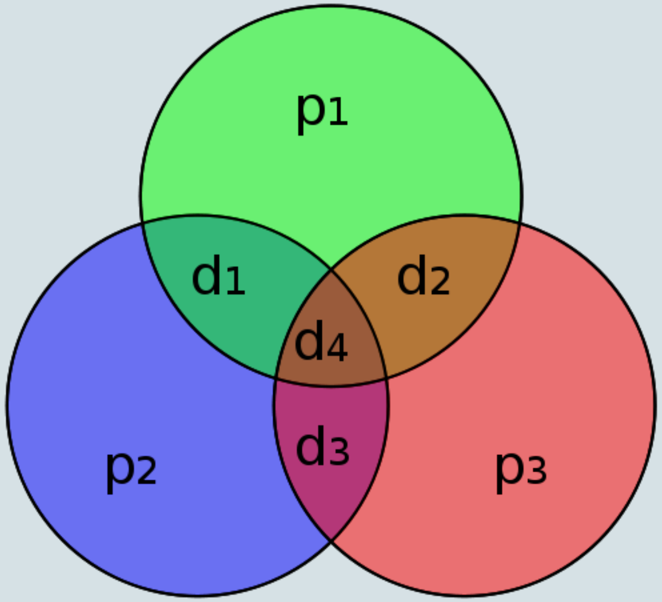
Error-correcting Codes: Example 2

Hamming(7,4) error-correcting code

Every block of 4 source bits is encoded in 7 bits; code rate $= 4/7$

Encoding algorithm:

- Place the 4 source bits in the circles at positions d_i
- Compute 3 parity bits p_i such that each circle contains an *even* number of 1's



Decoding algorithm can *correct* 1 error: redo the encoding and use differences in the recomputed party bits to locate an error

Practical Noisy-Channel Coding

Used in (deep-space) communication, on audio CDs and DVDs, ...

Convolutional codes versus Block codes

Most systematic codes known are bad; good block codes:

- Concatenated Reed-Solomon codes (on CD, DVD)
- Low-Density Parity-Check codes (LDPC)
- Turbo codes

Price paid for better channel codes: higher latency

Consult an expert

Summary

- Notion of information, entropy, capacity of noisy channel
- Storage and communication of information
- Source coding: compress data, remove redundancy

  Source Coding/Lossless Compression Theorem establishes limits
- Channel coding: protect data against random errors (noise), add redundancy

  Noisy-Channel Coding Theorem establishes limits
- Encryption: protect data against unauthorized access (cf. cryptography)