**Number Base Case**
$N = d_n R^n + d_1 R^1 + d_0 R^0$
the d specifies the Number system -> $d_2$ == binary
can also be written as $R_2$
This can also be used to expand numbers:
$N_{10} 255 = 2 * 10^2 + 5 * 10^1 + 5 * 10^0$
$N_2 110 = 1 * 2^2 + 1 * 2^1 + 0 * 2^0 => N_{10} 6$

**Quantities:**
**N** -> natural numbers | | **Z** -> full numbers
**Q** -> rational numbers | | **R** -> real numbers

**Common number systems:**
Decimal: $N_{10} = n * 10^n..0 * 10^0$
**Binary:** $N_2 = n * 2^n..0 * 2^0$
$2^{10} = 1024, 2^9 = 512, 2^8 = 256, 2^7 = 128, 2^6 = 64,$
$2^5 = 32, 2^4 = 16, 2^3 = 8, 2^2 = 4, 2^1 = 2, 2^0 = 1$
**Hexadecimal:** $N_16 = n * n^{16}..0 * 16^0$
notation: 0 1 2 3 4 5 6 7 8 9 A B C D E F
$16^5 = 1048576, 16^4 = 65536, 16^3 = 4096, 16^2 = 256,$
$16^1 = 16, 16^0 = 1$

**Modulo**
8 mod 4 = (8) -> 0 , 8 mod 3 = (6) -> 2 , 8 mod 5 = (5) -> 3
if x<y in x mod y then the result will always be x!
any negative numbers can be considered as NOTnegative
aka only absolute values! modulo deals with |x|
many programming languages actually do not follow this!
they have their own implementation of modulo.
$5 \equiv 3$ mod 2 -> as 5 mod 2 = 1 and 3 mod 2 = 1

**Codeword length**
**Byte = 8 bit || Word = 16 or 32 bit**
TCP packet = 1024 bit

**Cyclic group**
**Es sei $F(a) = a^3 + a + 1 = 0$,**
- Dann können wir zunächst festhalten
- a = a
- a² = a²  aber
- a³ = ***a*+1**
- a⁴ = a(***a* + 1**) = a² + a
- a⁵ = a(a² + a) = a³ + a² = a² + ***a*+1**
- a⁶ = a(a² + ***a* +1**) = a³ + ***a*² +*a* = *a*+1 + *a*² + *a* = *a*²** + 1
- a⁷ = ***a*(*a*²** + 1) = a³ + a = ***a*+1** + a = 1
- a⁸ = a : der Zyklus beginnt von vorne!
- **{0, 1, *a*, *a*², *a*+1, a² + a, a² + *a*+1,*a*² + 1 }**
- ***{000, 001, 010, 100, 011, 110, 111, 101}***

WHAT THE FUCK

**Result Quantity** the result of all possible outcomes
it is denoted with: $\Omega$
A single element of the result list is: $\omega$ -> $\omega \in \Omega$
The list of results is $|\Omega|$
Example Dice roll: $\Omega = \{1, 2, 3, 4, 5, 6\}$

Probability: $P(A) = \frac{\text{best results}}{\text{all results}} = \frac{|A|}{|\Omega|} = \frac{|A|}{n}$
So what is the probability of rolling a 6?
$P(\text{desired number to roll}) = \frac{\text{only 1 good result!}}{\text{6 possible results}} = \frac{1}{6}$
hence the chance is 1 in 6
Why this complicated method? You can modify desired results!
just change the A in P(A)!

**Inverse Probability: P(inverse) = 1 - P(A)**
dice -> $1 - \frac{1}{6} = \frac{5}{6}$

**Addition rule:**
$P(A \cup B) = P(A) + P(B) - P(A \cap B)$
!!The last part is needed, as otherwise the number
would exceed the possible states!!
$P(A \cup B \cup C) = P(A) + P(B) + P(C) - P(A \cap B) - P(A \cap C)$
$- P(B \cap C) + P(A \cap B \cap C)$

**Amount of possibilities:**
**ordered probes with replication:**
2 coins, head and tail, possibilities? **k**=head/tail=2 **n**=coins=2
$\Omega = n^k = 2^2$

**ordered probes without replication:**
5 dices. How many combinations?
dice numbers = **n** = 6 (1-6), dice amount = **k** = 5
possibilities $= \Omega = \frac{n!}{(n-k)!} = \Omega = \frac{6!}{(6-5)!} = 720$
Or this:
$\Omega = \Pi_n^{n-k+1} n = \Pi_6^{6-5+1} 6 = 2 * 3...5 * 6 = 720$

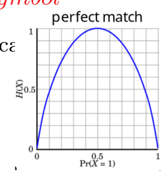
**unordered probes wihout replication:**
25 players, each should only play once with the other.
$\Omega = \frac{n!}{k!(n-k)!}$ -> $\frac{25!}{2!(25-2)!}$ -> $\frac{\text{too big}}{\text{too big}} = 300$
as you can see the bottom is a BIG calculation, so
$\Omega = \frac{\Pi_n^{n-k+1} n}{k!} -> \frac{\Pi_{25}^{25-2+1} 25}{2!} -> \frac{24 * 25}{2} = 300$
Note that **k** can also be defined as the
length of the tuple we want to receive.
-> (Player,Player) - > 2

**Source to Sink Information**

| Nachricht (Darstellung & Bedeutung) | redundant | nicht-redundant |
|---|---|---|
| irrelevant | Zeichenvorrat bei Quelle und Senke verschieden | |
| relevant | vorhersagbar | Information |

# Entropy
information content
this essentially just us how many bits are needed
**k** is base state count -> bit = 2
and **N** is the full number of states
example: list True,False,True,False 4 states total, base 2.
$H_0 = log_k(N)[k] -> H_0 = log_2(4)[bit] = 2$

information flow
essentially information content over time
$$H_0^* = \frac{log_2(N)}{\tau}[\frac{bit}{s}]$$
information quantity / Surprise
$$I(x_k) = -log_2(P(x_k))[bit]$$
Entropy (Surprise per element)
0 means no symbols. 1 means perfect balance 50-50
$$H(X) = \Sigma_{k=1}^N P(x_k) * I(x_k)[\frac{bit}{symbol}]$$
where X is the list of symbols




Sink Redundance / Code Redunca
$$R_Q = H_0 - H(X)[\frac{bit}{symbol}]$$
$$R_c = L - H(X)[\frac{bit}{symbol}]$$
Code Word Length
$$L(x_k) = \text{rounded}(I(x_k))[bit]$$
Median Code Word Length
$$L = \Sigma_{k=1}^N P(x_k) * L(x_k)[\frac{bit}{symbol}]$$
Entropy of the entire Code
$$H_c(X) = \Sigma_{k=1}^N P(x_k) * L(x_k)[\frac{bit}{symbol}]$$
$H_c$ can be a real number -> $H_c \in \mathbb{R}$

| **Für jede beliebige zugehörige Binärcodierung mit Präfixeigenschaft ist die mittlere Codewortlänge nicht kleiner als die Entropie $H(X)$:** | Für jede beliebige Quelle kann eine Binärcodierung gefunden werden, so dass die folgende Ungleichung gilt: |
|---|---|
| $H(X) \leq L$ | $H(X) \leq L \leq H(X) + 1$ |

Sink without memory
$$P(x_k, y_k) = P(x_k) + P(y_i)$$
Sink with memory
$$P(x_k, y_i) = P(x_k) + P(x_k|y_i)$$
Entropy without memory / Combined Entropy
$$H(H, Y) = \Sigma_{x_k}^N \Sigma_{y_i}^N P(x_k, y_i) * (-log_2(P(x_k, y_i))$$
or: $H(X, Y) = H(X) + H(Y)$
Entropy with memory
$$H(H, Y) = \Sigma_{x_k}^N \Sigma_{y_i}^N P(x_k) * P(x_k, y_i) * (-log_2(P(x_k) * P(x_k|y_i))$$

Encoding of Symbols
- **Ordne die Zeichen gemäss ihrer Auftrittswahrscheinlichkeit**
- **Die beiden Zeichen mit der kleinsten Auftrittswahrscheinlichkeit haben die gleiche CW-Länge $L_N$**
- **Sei $L_N$ die mittlere CW-Länge für eine Quelle mit $N$ Zeichen und $L_{N-1}$ die mittlere CW-Länge für den Fall, dass die beiden letzten zu einem einzigen Zeichen zusammengefasst werden, dann gilt:**

$$L_N - (p(x_{N-1}) + p(x_N)) \cdot L(x_N) = L_{N-1} - (p(x_{N-1}) + p(x_N)) \cdot (L(X_N) - 1)$$
$$\Rightarrow L_N = L_{N-1} + p(x_{N-1}) + p(x_N)$$

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0.22 | 0.19 | 0.15 | 0.12 | 0.08 | 0.07 | 0.07 | 0.06 | 0.04 |
| 1 | 2 | 3 | 4 | 8 | 9 | 5 | 6 | 7 |
| | | | | 0 | 1 | | | |
| 0.22 | 0.19 | 0.15 | 0.12 | 0.1 | | 0.08 | 0.07 | 0.07 |
| 1 | 2 | 3 | 6 | 7 | 4 | 8 | 9 | 5 |
| | | | 0 | 1 | | 0 | 1 | |
| 0.22 | 0.19 | 0.15 | 0.14 | | 0.12 | 0.1 | | 0.08 |
| 1 | 2 | 8 | 9 | 5 | 3 | 6 | 7 | 4 |
| | | 00 | 01 | 1 | | 0 | 1 | |
| 0.22 | 0.19 | 0.18 | | | 0.15 | 0.14 | | 0.12 |

continue this pattern until every symbol has a code
note the extra 0 on every step

Run Length Encoding RLE/RLC
- Quelltext w: Agggbbehfffgggg => |w|=15
- Codiert $w_c$ : A3g2beh3f4g => | $w_c$ |= 11

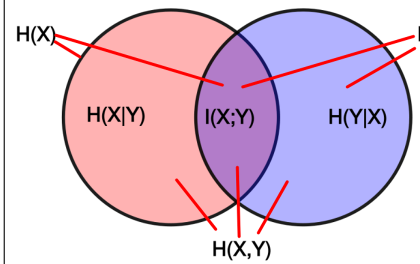
A + 3 x g + 2 x b + e + h + 3 x f + 4 x g

shortening of length
by compressing repetition.

**Encoder and Decoder**
You need to either choose 1 or 0 as the starting
bit. After that the decoder can print out the correct code.

**Chiffre text**
You can "encrypt" your data by shifting the codes by a certain amount.
In the caesar chiffre this is done with the number 4. a -> e
Please do not use this, use RSA or other algorithms.

Conditional Entropy -> Entropy of Y given X
$$H(Y|X) = \Sigma_{k=1}^N \Sigma_{i=1}^N P(x_k, y_i) * (-log_2(\frac{P(x_k, y_i)}{P(x_k)}))$$
Chain Rule
$$H(Y|X) = H(X, Y) - H(X) \,||\, H(Y \backslash X)$$
Bayes Rule
$$H(Y|X) = H(X|Y) - H(X) + H(Y) \,||\, H(Y \backslash X)$$




Transinformation
likelyhood of information being correct at arrival.
$$T = H(X) - H(X|Y) \,||\, H(Y) - H(Y|X)$$
or: $|(X; Y)$

distance to next valid codeword
$$Min_{i,j}(d(x_i, x_j))$$
error detection distance / Hamming distance
the amount of bits that differ from input to output
$$h = Min_{i,j}(d(x_i, x_j))$$
detection distance even/uneven
$$h = 2e + 2 \,||\, h = 2e + 1$$
error correction distance even/uneven
$$e = \frac{h-2}{2} \,||\, \frac{h-1}{2}$$