SHANNON'S THEORY

Shannon's theory of Communication has been the cornerstone in laying the foundations for the design of secure ciphers. It consists of three important parameters:

1. Diffusion

2. Confusion

3. Unconditional Security

In the sequel we explain in more detail these three principles.

ITERATION OF CIPHERS (DIFFUSION)

An iteration of a fixed transformation may initially display convincingly good encryption qualities, but may fail in the end to be a good encryption. This we discuss in the sequel with an example.

Consider the unit square $S = \{(x, y) : 0 \leq x, y < 1\}$ with toroidal wrap-around and the transformation $T(x, y) = (y, x')$ such that

$$y' = \begin{cases} x + y - 1 & \text{if } x + y \geq 1 \\ x + y & \text{if } 0 \leq x + y < 1 \end{cases}$$

The affine distortion of the picture is given by the matrix

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

Notice that if f_n is n-th Fibonacci number then

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n = \begin{pmatrix} f_{n-1} & f_n \\ f_n & f_{n+1} \end{pmatrix}$$

Observe that

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^4 = \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} + 3 \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^8 = \begin{pmatrix} 13 & 21 \\ 21 & 34 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + 3 \begin{pmatrix} 4 & 7 \\ 7 & 11 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^{16} = \begin{pmatrix} 610 & 987 \\ 987 & 1597 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + 21 \begin{pmatrix} 29 & 47 \\ 47 & 76 \end{pmatrix}$$

This gives fixed points for the transformation T, i.e., points (x, y) such that $T(x, y) = (x, y)$. E.g., the last equation implies

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^{16} \begin{pmatrix} i/21 \\ j/21 \end{pmatrix} = \begin{pmatrix} i/21 \\ j/21 \end{pmatrix} + \begin{pmatrix} 29i + 47j \\ 47i + 76k \end{pmatrix}$$

which gives 400 fixed points for the transformation T^{16}, with coordinates $(i/21, j/21)$, $0 < i, j < 21$. (Note that the toroidal property implies that both coordinates of the rightmost vector are 0.)

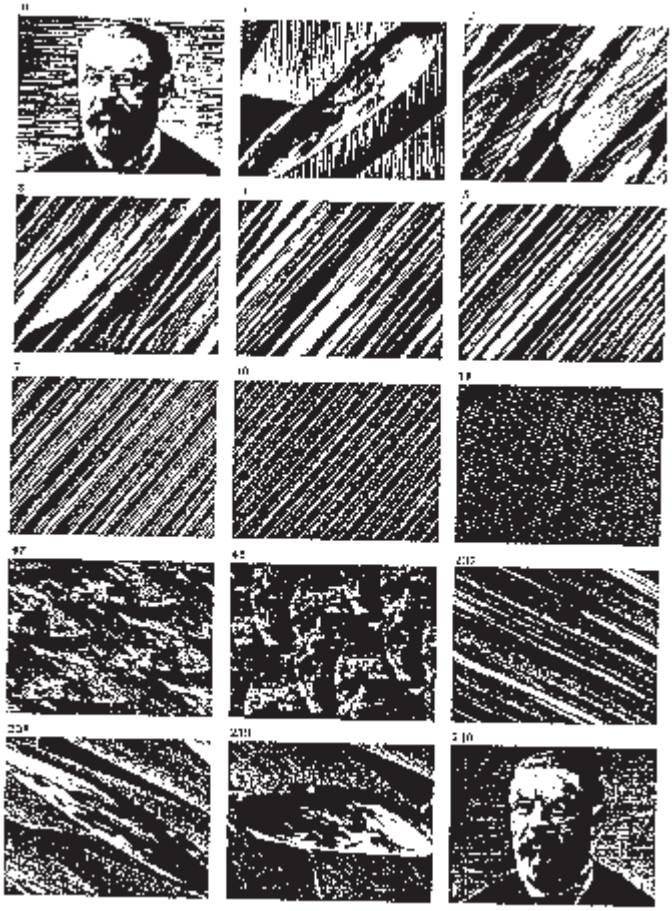
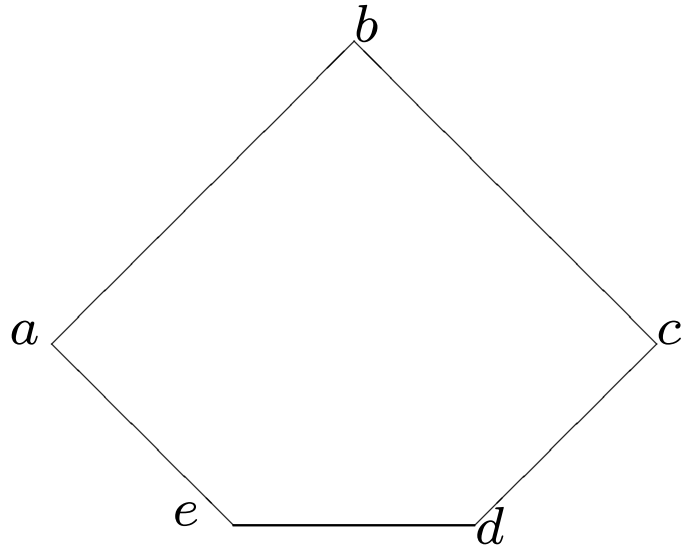


Fig. 72. Poincaré's resurrection

The Priciple of Confusion

Shannon posed the following question: Suppose we want to transmit messages across a channel (where symbols may be distorted). What is the maximum rate of transmission s.t. the receiver may recover the original message without errors?

If V is the set of symbols to be transmitted let us define the graph $G = (V, E)$, where $E(a, b) \Leftrightarrow$ symbol a may be confused with symbol b.



All five symbols could be used for transmission. But since we want to avoid errors we can send at most 2: either a, c or a, d or b, d or b, e or c, d.

I.e., an independent set of vertices, and the **information rate** is $1 = \log 2$.

The graph G thus defined is also called **confusion** graph of the symbols.

We can improve the information rate by transmitting n symbols at a time, for $n = 1, 2, 3, \ldots$

How can the pair xy be confused with the pair $x'y'$? If one of the following holds:

$x = x'$ and $y \neq y'$ can be confused
$y = y'$ and $x \neq x'$ can be confused
$x \neq x'$ can be confused and $y \neq y'$ can be confused

This amounts to taking the product of a graph with itself, i.e. $G, G^2 := G \times G, G^3 := G \times G \times G, \ldots$

We call G^n the **confusion** graph for strings of length n.

For a graph G let $\alpha(G)$ be the size of the largest independent set.

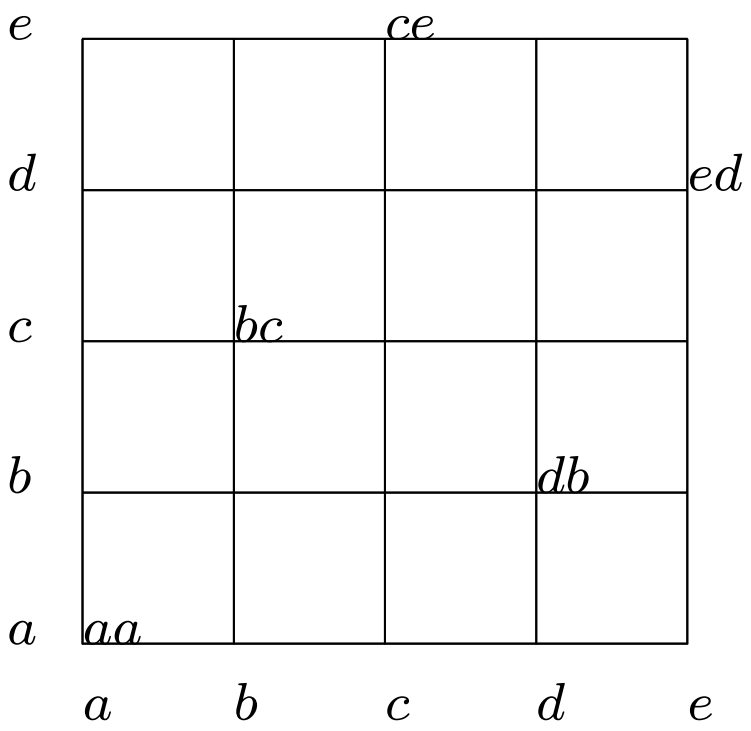
The information rate for strings of length n is defined by

$$\frac{\log_2 \alpha(G^n)}{n} = \log_2((\alpha(G^n))^{1/n})$$

Easy to see that $\alpha(G^n) \geq (\alpha(G))^n$.

Shannon defined the **zero-error capacity** of G by $\Theta(G) := \sup_n(\alpha(G^n))^{1/n}$. Thus for the 5-cycle C_5, $\alpha(C_5) = 2$ and the information rate is 1.

Can we compute $\Theta(C_5)$? Here is the graph $C_5 \times C_5$.



The set aa, bc, ce, db, ed is independent. So $\alpha(C_5 \times C_5) = 5$. It follows that $\Theta(C_5) \geq \sqrt{5}$ and hence the information rate per two symbols is at least

$$\frac{\log 5}{2} \approx 1.16 > 1$$

It is non-trivial to show that in fact $\Theta(C_5) = \sqrt{5}$! A famous open problem is to compute $\Theta(C_7)$ and more generally $\Theta(C_n)$, when n is odd!

PRODUCT CIPHERS (Creating Confusion)

For simplicity consider cryptosystems such that $P = C$.

Given two cryprosystems S_1, S_2 with respective keyspaces K_1, K_2 we define the product cryptosystem $S_1 \times S_2$ to have keyspace $K_1 \times K_2$, encryption rule

$$E_{(k_1,k_2)}(x) = E_{k_2}(E_{k_1}(x))$$

and decryption rule

$$D_{(k_1,k_2)}(y) = D_{k_1}(D_{k_2}(y))$$

Example: Define the multiplicative cipher M by the encryption rule $E_a(x) = ax$ mod 26, where gcd$(a, 26) = 1$. If S is the shift cipher then it is easy to show that $S \times M = M \times S$ (i.e., S and M commute) is the affine cipher.

The product operation $\times$ on ciphers is associative but not commutative.

We can also iterate the product operation to define new ciphers. Given a cipher S define

$$\begin{aligned} S^2 &= S \times S \\ S^{n+1} &= S^n \times S \end{aligned}$$

If $S^2 = S$ then the cipher is called **idempotent**. Similarly, we can look for the smallest integer n such that $S^n = S$ (if any).

Example: The Shift, Substitution, Affine, Hill, Vigenere, and Permutation ciphers are all idempotent.

It makes no sense to iterate idempotent ciphers. But even if it is not idempotent there is no guarantee that (repeated) iteration will lead to a good cipher (see next example).

The idea of product ciphers and iterations are used to build DES-like ciphers.

DIFFUSION AND CONFUSION

Shannon introduced the concepts of **Diffusion** and **Confusion** in order to capture the two building blocks of any cryptosystem.

Ciphers are constructed as "products" of these two basic operations. In ideal ciphers an appropriate composition of these operations should lead to statistics such that the ciphertext is independent of the key.

In **Diffusion** the statistical structure of the plaintext is dissipated. This is achieved by having each ciphertext character be affected by many plaintext characters, e.g., "adding" k successive plaintext letters to get a ciphertext letter (for k sufficient large).

In **Confusion** the relationship between the statistics of the ciphertext and encryption key are made as complex as possible. For this we use complex substitution algorithms, e.g., permutations).

Unconditional Security

We will develop a theory of unconditional security using the framework of probability theory. Given random variables X, Y we define:

$p(x) = \mathsf{Pr}[X = x], p(y) = \mathsf{Pr}[Y = y]$.

joint prob.: $p(x, y) = \mathsf{Pr}[X = x, Y = y]$.

conditional prob.: $p(x|y) = \mathsf{Pr}[X = x|Y = y]$.

independence: $p(x, y) = p(x)p(y)$.

Formula: $p(x, y) = p(x|y)p(y) = p(y)p(y, x)$.

Bayes' Theorem: $p(x|y) = \frac{p(x)p(y|x)}{p(y)}$, assuming $p(y) > 0$.

X, Y independent $\Leftrightarrow p(x|y) = p(x) \Leftrightarrow p(y|x) = p(y)$, for all x, y.

DISTRIBUTIONS IN CRYPTOSYSTEMS

We assume that a given key is used for only one encryption.

Probability distributions are given on the plaintext space P and key space K: $p_P(x), p_K(k)$.

It is reasonable to assume that they are independent, because the key is chosen in advance without knowledge of what the plaintext will be.

These two distributions induce a distribution on the cipherspace C: $p_C(y)$, which we now estimate.

Let $C(k) = \{E_k(x) : x \in P\}$ be the set of possible ciphertexts which are encrypted by the key k.

The following calculations can be made by anyone who knows the probability distributions of the plaintext and keyspace.

For a given ciphertext $y \in C$ we have:

$$p_C(y) = \sum_{\{k:y \in C(k)\}} p_K(k) p_P(D_k(y))$$

We can compute easily the conditional probability that $y \in C$ is the ciphertext given that $x \in P$ is the plaintext:

$$p_C(y|x) = \sum_{\{k:x=D_k(y)\}} p_K(k)$$

Hence, using Bayes' formula we can also compute the conditional probability that x is the plaintext given that y is the ciphertext:

$$p_C(x|y) = \frac{p_P(x) \sum_{\{k:x=D_k(y)\}} p_K(k)}{\sum_{\{k:y \in C(k)\}} p_K(k) p_P(D_k(y))}$$

A cryptosystem has **perfect secrecy** if $p_P(x|y) = p(x)$, for all $x \in P$ and $y \in C$.

Example

Let $P = \{a, b\}$, $K = \{k_1, k_2, k_3\}$, with $p_P(a) = 1/4, p_P(b) = 3/4$, $p_K(k_1) = 1/2$, and $p_K(k_2) = p_K(k_3) = 1/4$. Let $C = \{1, 2, 3, 4\}$.

Encryption E_k:

	a	b
k_1	1	2
k_2	2	3
k_3	3	4

Probability Distribution of p_C:

$$\begin{aligned} p_C(1) &= 1/8 \\ p_C(2) &= 3/8 + 1/16 = 7/16 \\ p_C(3) &= 3/16 + 1/16 = 1/4 \\ p_C(4) &= 3/16 \end{aligned}$$

Conditional Probability Distributions:

$$\begin{aligned} p_P(a|1) &= 1 & p_P(b|1) &= 0 \\ p_P(a|2) &= 1/7 & p_P(b|2) &= 6/7 \\ p_P(a|3) &= 1/4 & p_P(b|3) &= 3/4 \\ p_P(a|4) &= 0 & p_P(b|4) &= 1 \end{aligned}$$

Shift Cipher has Perfect Secrecy

Assume that the 26 keys have equal probability 1/26. p_C is computed as follows:

$$p_C(y) = \sum_{k \leq 25} p_K(k) p_P(D_k(y)) = \sum_{k \leq 25} \frac{p_P(y-k)}{26}$$

Since $\{y - k : y \in C\}$ is a permutation of the 26 ciphercharacters we must have

$$\sum_{k \leq 25} \frac{p_P(y-k)}{26} = \frac{1}{26} \sum_{k \leq 25} p_P(y) = \frac{1}{26}$$

For every x, y there is a unique k (namely, $k = y - x$) such that $E_k(x) = y$. Hence $p_C(y|x) = p_K(y - x) = 1/26$. Using Bayes' formula we have:

$$p_P(x|y) = \frac{p_P(x) p_C(y|x)}{p_C(y)} = \frac{p_P(x)/26}{1/26} = p_P(x)$$

Hence, the shift cipher is unbreakable provided that a new random key is used to encrypt every plaintext character.

Perfect Secrecy

W.l.o.g. we may assume $p_C(y) > 0$, $\forall y \in C$ (if not, then y is never used and can be omitted).

Assume the cryptosystem has perfect secrecy.

By Bayes' theorem: $p_P(x|y) = p_P(x), \forall x \in P, y \in C \Leftrightarrow p_C(y|x) = p_C(y), \forall x \in P, y \in C$

Fix $x \in P$. For each $y \in C$ we have $p_C(y|x) = \sum_{\{k:x=D_k(y)\}} p_K(k) = p_C(y) > 0$.

Hence, for each $y \in C$ there is at least one key $k \in K$ such that $E_k(x) = y$.

It follows that $|K| \geq |C|$. In any cryptosystem we must always have $|C| \geq |P|$. Hence,

$$|K| \geq |C| \geq |P|$$

What happens in the boundary case

$$|K| = |C| = |P|?$$

Continued

(**Shannon's Theorem**) Suppose the cryptosystem (P, C, K, E, D) satisfies $|K| = |C| = |P|$. Then it has perfect secrecy $\Leftrightarrow$ (a) every key is used with equal probability $1/|K|$ and (b) $\forall x \in P \exists! k \in K(E_k(x) = y)$.

($\Rightarrow$) Assume the cryptosystem provides perfect secrecy, i.e., $p_P(x|y) = p_P(x), \forall x \in P, y \in C$. By the previous observation, for each $x \in P$, and $y \in C$ there is at least one key $k \in K$ such that $E_k(x) = y$.

Hence, $|C| = |\{E_k(x) : k \in K\}| \leq |K|$. By assumption $|C| = |K|$. Hence also $|K| = |\{E_k(x) : k \in K\}|$. It follows there do not exist keys $k_1 \neq k_2$ such that $E_{k_1}(x) = E_{k_2}(x) = y$.

This proves part (b): $\forall x \in P \exists! k \in K(E_k(x) = y)$ of the implication.

Continued

Now we prove part (a). Fix $y \in C$. Put $n = |K|$ and let $P = \{x_1, x_2, \ldots, x_n\}$ and $K = \{k_1, k_2, \ldots, k_n\}$ in such a way that $E_{k_1}(x_1) = E_{k_2}(x_2) = \cdots = E_{k_n}(x_n) = y$

The perfect secrecy condition and Bayes' theorem imply:

$$\begin{aligned} p_P(x_i) = p_P(x_i|y) &= \frac{p_C(y|x_i)p_P(x_i)}{p_C(y)} \\ &= \frac{p_K(k_i)p_P(x_i)}{p_C(y)} \end{aligned}$$

This implies $p_K(k_i) = p_C(y)$, for all $i = 1, 2, \ldots, n$ and completes the proof of part (b).

The converse, i.e. that parts (a) and (b) imply perfect secrecy is the same as the proof of the perfect secrecy of the shift cipher proved earlier.

(Vernam) One-time Pad

Shannon's Theorem offers a rigorous proof of the perfect secrecy of Vernam's cipher, first proposed by Vernam in 1917. The cryptosystem can be described as follows. $|P| = |C| = |K| = (Z_2)^n$.

The key $k = (k_0, k_1, \ldots, k_{n-1})$ is a random sequence of bits generated by some "good random generator".

A plaintext $x = (x_0, \ldots, x_{n-1})$ is encrypted by $E_k(x) = (x_0 + k_0 \bmod 2, \ldots, x_{n-1} + k_{n-1} \bmod 2)$

A ciphertext $y = (y_0, \ldots, y_{n-1})$ is derypted by $D_k(y) = (y_0 + k_0 \bmod 2, \ldots, y_{n-1} + k_{n-1} \bmod 2)$

Unfortunately, it is vulnerable to known plaintext attack: k can be computed from x and $E_k(x)$.

Also, the fact that $|K| \geq |P|$ means that the "amount" of keys to be communicated is as big as the "amount" of plaintext.

ENTROPY

We considered the case when a key is used for only one encryption. What happens when the same key is used on more than one plaintext? The basic tool is the **entropy** which also measures the uncertainty of a system.

Consider a random variable X which takes on a finite set of values with probability distribution $p(X) : \Pr[X = x]$.

Question: What is the information gained by an event that takes place with probability distribution $p(X)$?

Equivalent Question: If the event has not taken place what is the uncertainty about the outcome?

We call this quantity entropy of X and denote by $H(X)$.

We are interested in quantifying "information" in the sense of probability theory.

This is not the same as information in everyday language.

The concept has nothing to do with "meaning". Instead it is related more to "surprise".

1. There was snow in Ottawa in July.

2. I will have dinner tomorrow.

3. JQQS QRVQ UQAT QQAB

The first sentence is most surprising, the second has meaning but no surprise, and the third is garbage!

Given an experiment with n equiprobable outcomes $E_1, E_2, \ldots, E_n$ how much information is conveyed on the average by a message M telling us which of the outcomes has actually occurred?

It seems reasonable to take as measure of this information the average length of the message provided M is written in an "economical" way.

For example, suppose we use "binary code" of length t, a sequence $b_1 b_2 \cdots b_t$ of bits. Obviously there are 2^t such sequences of length t. To identify uniquely all the events we must choose t s.t. $n < 2^t$. Hence, $t = \log n$ would be a reasonable definition of the amount of information in the message.

Let the events $E_1, E_2, \ldots, E_n$ have probabilities $p_1, p_2, \ldots, p_n$, respectively. Now perform an experiment N times and send a message M conveying the result of the whole series of M trials. An experiment is a sequence $E_{i_1}, E_{i_2}, \ldots, E_{i_N}$ with E_{i_k} the outcome of the k-th trial.

If N_k is the number of occurrences of the event E_k then $p_k \approx \frac{N_k}{N}$, by the law of large numbers; hence $N_k \approx Np_k$ and $N = N_1 + N_2 + \cdots + N_n$. It follows, the total number of outcomes is

$$s_n \approx \frac{N!}{N_1! \cdots N_n!} \tag{1}$$

This suggests the average number t of bits needed to encode the message M should satisfy $t = \frac{\ln s_n}{N}$. Using (1) and Stirling's approximation

$$s_n \approx \frac{\sqrt{2\pi N} N^N e^{-N}}{\sqrt{2\pi N_1} N_1^{N_1} e^{-N_1} \cdots \sqrt{2\pi N_n} N_n^{N_n} e^{-N_n}}$$

It follows that

$$\begin{aligned} \ln s_n &\approx N \ln N - \textstyle\sum_{k=1}^{N} N_k \ln(Np_k) \\ &\approx N \ln N - \textstyle\sum_{k=1}^{N} Np_k \ln(Np_k) \\ &= N \ln N - \textstyle\sum_{k=1}^{N} Np_k \ln N - \sum_{k=1}^{N} Np_k \ln p_k \\ &= -N \textstyle\sum_{k=1}^{N} p_k \ln p_k \end{aligned}$$

Hence,

$$t = \frac{\ln s_n}{N} = -\sum_{k=1}^{N} p_k \ln p_k$$

Example 1: Let X denote the toss of a fair coin with $\Pr[X = heads] = \Pr[X = tails] = 1/2$. The information or entropy of a coin toss is 1 bit, because it takes one bit to encode heads or tails. Similarly the entropy of n independent coin tosses is n.

Example 2: Let X denote a random variable that takes on the values x_1, x_2, x_3 with $\Pr[X = x_1] = 1/2, \Pr[X = x_2] = \Pr[X = x_3] = 1/4$. The "most efficient" encoding of the three events is to encode $X = x_1$ with 0, and $X = x_2, X = x_3$ with $01, 11$, respectively. The average number of bits in this encoding is

$$\frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 = \frac{3}{2}$$

The examples suggest that the number of bits required to encode an event $X = x$ might be approximately $-\log \Pr[X = x]$, which gives rise to the following definition.

The entropy of the random variable X is defined to be

$$H(X) = -\sum_x \Pr[X = x] \log \Pr[X = x]$$

For a cryptosystem, we are interested in the entropies $H(K), H(P), H(C)$ of the random variables of K, P, C.

Remark: If $\Pr[X = x] = 0$, we define $\Pr[X = x] \log \Pr[X = x] = 0$. ($\lim_{t \to \infty} t \log t = 0$.)

Let the r.v. X take on values $x_1, , \ldots, x_n$.

If $\Pr[X = x_i] = 1/n$, $\forall i$, then

$$H(X) = -\sum_{i=1}^{n} \Pr[X = x_i] \log \Pr[X = x_i] = \log n$$

Also, it is easy to see that $H(X) = 0 \Leftrightarrow \exists i(\Pr[X = x_i] = 1 \& \forall j \neq i(\Pr[X = x_j] = 0))$

(Previous) Example

Let $P = \{a, b\}$, $K = \{k_1, k_2, k_3\}$ with corresponding distributions

$\cdot$	$p_P(\cdot) =$
a	$1/4$
b	$3/4$

$\cdot$	$p_K(\cdot) =$
k_1	$1/2$
k_2	$1/4$
k_3	$1/4$

Let $C = \{1, 2, 3, 4\}$. The distribution p_C is:

$$\begin{aligned} p_C(1) &= 1/8 \\ p_C(2) &= 3/8 + 1/16 = 7/16 \\ p_C(3) &= 3/16 + 1/16 = 1/4 \\ p_C(4) &= 3/16 \end{aligned}$$

We can easily calculate the entropies

$$\begin{aligned} H(P) &= -\tfrac{1}{4}\log\tfrac{1}{4} - \tfrac{3}{4}\log\tfrac{3}{4} && \approx 0.81 \\ H(K) &= -\tfrac{1}{2}\log\tfrac{1}{2} - 2\tfrac{1}{4}\log\tfrac{1}{4} && = 1.5 \\ H(C) &= && \approx 1.85 \end{aligned}$$

HUFFMAN ENCODINGS

Let X be a r.v. which takes on a finite set of values. An encoding of X is a function $f : X \to \{0,1\}^*$. We can extend f to a function $f : X^* \to \{0,1\}^*$ in the obvious way. Given a finite string $x_1, \ldots, x_n$ of events we define

$$f(x_1, \ldots, x_n) = f(x_1)|| \cdots ||f(x_n),$$

where $||$ is string concatenation. If a string $x_1 \cdots x_n$ is produced by a memoryless source according to a probability distribution on X then the probability of the string $x_1 \cdots x_n$ is equal to $p(x_1) \cdots p(x_n)$.

Example: Let $X = \{a, b, c, d\}$ and consider the following three encodings.

$f(a) = 1$	$f(b) = 10$	$f(c) = 100$	$f(d) = 1000$
$g(a) = 0$	$g(b) = 10$	$g(c) = 110$	$g(d) = 111$
$h(a) = 0$	$h(b) = 01$	$h(c) = 10$	$h(d) = 11$

Note that both f, g are injective but h is not because $h(ac) = h(ba) = 010$.

HUFFMAN ENCODINGS AND ENTROPY

For any enconding of X and probability distribution on X we define the efficiency of the encoding

$$L(f) = \sum_{x \in X} p(x)|f(x)|$$

Question: How do we minimize $L(f)$?

Theorem: There is an algorithm (known as Huffman algorithm) that computes a "minimal" encoding. Moreover, the encoding produced by the algorithm satisfies

$$H(X) \leq L(f) \leq H(X) + 1$$

The algorithm is as follows: Begin with the probability distribution of X and assign code $\emptyset$ to each element. In each iteration two elements with lowest probability are combined into one element with probability the sum of the two probabilities. The smaller of the two is assigned "0" and the larger "1". When only one element remains then the coding for each $x \in X$ can be constructed by following the sequence of elements "backwards".

Example: Let $X = \{a, b, c, d, e\}$ have the probability distribution $p(a) = .05, p(b) = .1, p(c) = .12, p(d) = .13, p(e) = .6$ Huffman's algorithm works as follows

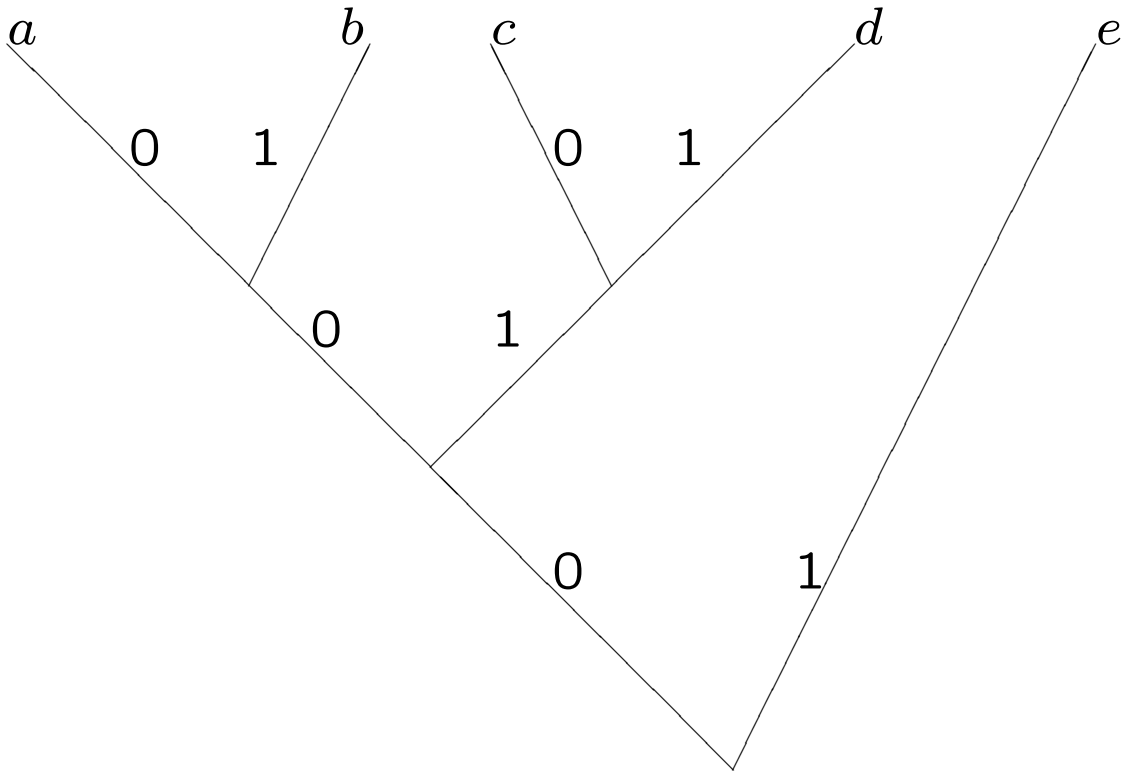
a	b	c	d	e
.05	.10	.12	.13	.60
0	1			
.15		.12	.13	.60
		0	1	
.15			.25	.60
0			1	
	.40			.60
	0			1
			1.0	

Which gives the encoding

symbol	a	b	c	d	e
f(symbol)	000	001	010	011	1

We see easily that $L(f) = 1.8$ and $H(X) = 1.7402$.

The Huffman Tree



The conditional entropy measures the average amount of information about X that is revealed by Y. For X, Y r.v. we define the **conditional entropy** by

$$\begin{aligned} H(X|y) &= -\sum_x p(x|y) \log p(x|y) \\ H(X|Y) &= -\sum_y p(y) H(X|y) \end{aligned}$$

PROPERTIES OF ENTROPY

Let the r.v. X take on values $x_1, , \ldots, x_m$ and let $p_i = \Pr[X = x_i]$. Let the r.v. Y take on values $y_1, , \ldots, y_n$ and let $q_j = \Pr[Y = y_j]$.

1. $H(X) \leq \log m$, with equality $\Leftrightarrow \forall i (p_i = 1/m)$.

2. $H(X, Y) \leq H(X) + H(Y)$, with equality $\Leftrightarrow$ X, Y are independent.

3. $H(X, Y) = H(Y) + H(X|Y)$.

4. $H(X|Y) \leq H(X)$, with equality $\Leftrightarrow X, Y$ are independent.

We need Jensen's inequality: If f is continuous and strictly concave on the interval I (**example:** $\log x$) and $a_i > 0, \forall i,\ a_1 + \cdots + a_m = 1$ then

$$\sum_{i=1}^{m} a_i f(x_i) \leq f\left(\sum_{i=1}^{m} a_i x_i\right),$$

with equality $\Leftrightarrow x_1 = \cdots = x_m = 1/m$.

Proof of 1.

$$\begin{aligned} H(X) &= -\textstyle\sum_{i=1}^{m} p_i \log p_i \\ &= \textstyle\sum_{i=1}^{m} p_i \log(1/p_i) \\ &\leq \log \textstyle\sum_{i=1}^{m} p_i (1/p_i) \\ &= \log m, \end{aligned}$$

with equality $\Leftrightarrow p_1 = \cdots = p_m = 1/m$.

Proof of 2. Denote by $r_{i,j} = \Pr[X = x_i, Y = y_j]$ and observe that

$$p_i = \sum_{j=1}^{n} r_{i,j},\ \ q_j = \sum_{i=1}^{m} r_{i,j}.$$

$$\begin{aligned} H(X)+H(Y) &= -\sum_{i=1}^{m} p_i \log p_i - \sum_{j=1}^{n} q_j \log q_j \\ &= -\sum_{i=1}^{m}\sum_{j=1}^{n} r_{i,j} \log p_i \\ &\quad -\sum_{j=1}^{n}\sum_{i=1}^{m} r_{i,j} \log q_j \\ &= -\sum_{i=1}^{m}\sum_{j=1}^{n} r_{i,j} \log(p_i q_j), \end{aligned}$$

and

$$H(X,Y) = -\sum_{i,j} r_{i,j} \log r_{i,j} = \sum_{i,j} r_{i,j} \log \frac{1}{r_{i,j}}$$

It follows from (**??**) and Jensen's inequality that

$$\begin{aligned} H(X,Y) - H(X) - H(Y) &= \sum_{i,j} r_{i,j} \log \frac{p_i q_j}{r_{i,j}} \\ &\leq \log \sum_{i,j} r_{i,j} \frac{p_i q_j}{r_{i,j}} \\ &= \log \sum_{i,j} p_i q_j = 0, \end{aligned}$$

with equality if and only if $r_{i,j}/p_i q_j = c, \forall i,j$, for some constant c. We conclude that $c = 1$ using the equation

$$1 = \sum_{i,j} r_{i,j} = c \cdot \sum_{i,j} p_i q_j = c \cdot \left(\sum_i p_i\right)\left(\sum_j q_j\right) = c$$

Proofs of 3 & 4: are easy.

ENTROPY AND CRYPTOSYSTEMS

Key equivocation: $H(K|C)$.

Theorem: For any cryptosystem $H(K|C) = H(K) + H(P) - H(C)$.

Observe that by property 3,

$$\begin{aligned} H(K,P,C) &= H(K,P) + H(C|K,P) \\ H(K,P,C) &= H(K,C) + H(P|K,C) \end{aligned}$$

Since the key and plaintext determine uniquely the ciphertext, i.e., $y = E_k(x)$, we have that $H(C|K,P) = 0$. Hence, we have $H(K,P,C) = H(K,P)$. Similarly, key and ciphertext determine uniquely plaintext, i.e. $x = D_k(y)$. Hence, $H(P|K,C) = 0$ and $H(K,P,C) = H(K,C)$.

Since K, P are independent $H(K,P) = H(K) + H(P)$.

Hence, $H(K,P,C) = H(K,P) = H(K) + H(P)$.

$H(K|C) = H(K,C) - H(C) = H(K,P,C) - H(C) = H(K) + H(P) - H(C)$.

(Previous) Example

Let $P = \{a, b\}$, $K = \{k_1, k_2, k_3\}$, with $p_P(a) = 1/4, p_P(b) = 3/4$, $p_K(k_1) = 1/2$, $p_K(k_2) = p_K(k_3) = 1/4$. Let $C = \{1, 2, 3, 4\}$.

$$\begin{array}{lll} H(P) & = -\frac{1}{4}\log\frac{1}{4} - \frac{3}{4}\log\frac{3}{4} & \approx 0.81 \\ H(K) & = -\frac{1}{2}\log\frac{1}{2} - 2\frac{1}{4}\log\frac{1}{4} & = 1.5 \\ H(C) & = & \approx 1.85 \end{array}$$

The equivocation of this cryptosystem is given by the formula

$$\begin{aligned} H(K|C) &= H(K) + H(P) - H(C) \\ &= 1.5 + 0.81 - 1.85 \\ &= 0.46 \end{aligned}$$

This is the amount of information about the key which is revealed by the ciphertext.

SPURIOUS KEYS

Suppose we are looking at ciphertext only attacks in a given cryptosystem. Let us assume that a plaintext $x = x_1 \cdots x_n$ from a natural language, say english, is enciphered as ciphertext $y = y_1 \cdots y_n$.

In general, it may be possible to rule out certain keys, but many "possible" keys remain, only one of which is the correct one. The remaining possible but incorrect keys are called **spurious**.

Example: Consider the ciphertext WNAIW which was encrypted using the shift cipher. By analyzing the ciphertext it is easy to see that there are only two meaningful texts: *river* and *arena*. Of the two keys one is **correct** (with shift 5) and the other is **spurious** (with shift 22).

Question: Is there a bound on the (expected) number of spurious keys?

ENTROPY AND REDUNDANCY

Let P^n be the random variable which has as probability distribution all n-grams of plaintext in a (natural) language.

For a given natural language L, the **entropy** of L is defined by

$$H_L = \lim_{n\to\infty} \frac{H(P^n)}{n},$$

and is the average amount of information per letter in a meaningful string of plaintext. (Note that a random string of letters would have entropy $\log 26 \approx 4.76$.) The **redundancy** of L is given by

$$R_L = 1 - \frac{H_L}{\log |P|}.$$

A random language has entropy $\log |P|$ (and redundancy 0) and so R_L measures the fraction of "excess" characters which are thought of as redundant.

Example: For the english language tabulations have been made of $H(P^n)$ yielding

$$\begin{aligned} H(P) &\approx 4.19 \\ H(P^2) &\approx 3.90 \end{aligned}$$

By performing experiments in computing trigrams, etc., in the limit we obtain that $1.0 \leq H_L \leq 1.5$, i.e. the average information content in English is about 1.5 bits per letter. Substituting

H_L	R_L
1.00	$1 - 1.00/\log 26 = 0.79$
1.25	$1 - 1.25/\log 26 = 0.74$
1.50	$1 - 1.50/\log 26 = 0.69$

So redundancy in the English language ranges from 69% to 79%, i.e. if $R_L = 1.25$ we can find a Huffnan encoding of n-grams (for n sufficiently large) that will compress english to about a quarter its original length.

Theorem: Consider a cryptosystem with $|P| = |C|$, keys are chosen equiprobably, and R_L is the redundancy of the underlying language. For n sufficiently large, the expected number (denoted by $\bar{s}_n$) of spurious keys of a given string of ciphertext of length n satisfies

$$\bar{s}_n \geq \frac{|K|}{|P|^{nR_L}} - 1$$

Proof: Consider the set of "possible" keys given that $y \in C^n$ is the ciphertext, i.e.,

$$K(y) = \{k \in K : \exists x \in P^n (E_k(x) = y \& p_P(x) > 0)\}$$

Since exactly one of the "possible" keys is correct we have that $|K(y)| - 1$ is the number of spurious keys.

The expected (average) number of spurious keys is given by

$$\begin{aligned} \bar{s}_n &= \sum_{y \in C^n} p(y)(|K(y)| - 1) \\ &= \sum_{y \in C^n} p(y)|K(y)| - \sum_{y \in C^n} p(y) \\ &= -1 + \sum_{y \in C^n} p(y)|K(y)| \end{aligned}$$

By the previous theorem we have $H(K|C^n) = H(K) + H(P^n) - H(C^n)$.

For n sufficiently large we have $H(P^n) \approx nH_L = n(1 - R_L) \log |P|$ and clearly, $H(C^n) \leq n \log |C|$. Since, $|C| = |P|$ we have that

$$H(K|C^n) \geq H(K) - nR_L \log |P|. \qquad (2)$$

However, we have

$$\begin{array}{rl} H(K|C^n) & = \sum_{y \in C^n} p(y) H(K|y) \\ & = \sum_{y \in C^n} p(y) \log |K(y)| \\ & \leq \sum_{y \in C^n} p(y) |K(y)| \\ & = \log(\bar{s}_n + 1) \end{array} \qquad (3)$$

Combining inequalities (2), (3) we obtain the inequality

$$\log(\bar{s}_n + 1) \geq H(K) - nR_L \log |P|.$$

When keys are equiprobable, $H(K) = \log |K|$ and the theorem follows.

UNICITY DISTANCE

The **unicity distance** of a cryptosystem is the value of n at which the expected number of spurious keys becomes zero, i.e., the average amount of ciphertext required for an opponent to be able to uniquely compute the key given enough computing time. If we set $\bar{s}_n = 0$ and solve for n we obtain the formula

$$n \approx \frac{\log |K|}{R_L \log |P|}$$

Example 1: In the substitution cipher $|P| = 26$ and $|K| = 26!$. If we take $R_L = 0.75$ we obtain as unicity distance the value

$$n \approx \frac{\log(26!)}{0.75 \log 26} \approx 25,$$

which means given a cipher of at least 25 characters a unique decryption is possible.

Example 2: In the shift cipher $|P| = |K| = 26$ and the unicity distance is approximately 1.33.