Part 1 - Prof. Taricco

1. Illustrate the basic concepts on probability: probability spaces, events, outcomes, probability function, σ-algebra assumptions.

Probability is based on the concept of probability space, consisting in the following three components:

1. A set containing all possible outcomes: Ω.
2. A set of events, $\mathcal{F}$, whose elements are subsets of outcomes: $\mathcal{F} \subseteq \Omega$.
3. A probability function $P : \mathcal{F} \mapsto [0, 1]$, assigning a probability to every event.

The probability function is a normalized measure

1. In the discrete case (when Ω is a finite or countably infinite set), the probability of the event $E \in \mathcal{F}$ is the sum of the probabilities of the outcomes in E:

$$P(E) = \sum_{\omega \in E} P(\omega).$$

   The outcomes are events themselves with positive probabilities $P(\omega)$.

2. In the continuous case, it is an integral of the probability density function (pdf) over the event $E \in \mathcal{F}$:

$$P(E) = \int_{\omega \in E} d\mu(\omega).$$

   In this case, the outcomes are uncountable and are not events.
3. The set Ω is also an event: $\Omega \in \mathcal{F}$. The following normalization holds:

$$P(\Omega) = 1,$$

   that is, the probability of the set of all possible outcomes is equal to 1.

Given two events A, B, we can build the union $A \cup B$ and the intersection $A \cap B$:

- $A \cup B =$ set of outcomes in A or in B.
- $A \cap B =$ set of outcomes in A and in B.

Some additional technical assumptions on $\mathcal{F}$ are made in order that probabilities can be calculated properly (σ-algebra assumptions):

- If $E \in \mathcal{F}$ also its complement $\Omega \backslash E \in \mathcal{F}$.
- If $E_n \in \mathcal{F}$ for $n = 1, 2, 3, \ldots$ (possibly a countable infinite sequence of event), then $\cup_n E_n \in \mathcal{F}$, as well.

- The probability of the intersection is called joint probability:

$$P(A, B) \equiv P(A \cap B)$$

- The conditional probability is

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

- An important result is the total probability law.
- If the events $B_i, i = 1, 2, \ldots$ form a partition of Ω (i.e., $\uplus_i B_i = \Omega$ and $B_i \cap B_j = \emptyset$ for $i \neq j$), then:

$$P(A) = \sum_i P(A, B_i) = \sum_i P(A \mid B_i) P(B_i)$$

2. Illustrate the basic properties of probabilities.

Probability is based on the concept of probability space, consisting in the following three components:

1. A set containing all possible outcomes: Ω.
2. A set of events, $\mathcal{F}$, whose elements are subsets of outcomes: $\mathcal{F} \subseteq \Omega$.
3. A probability function $P : \mathcal{F} \mapsto [0, 1]$, assigning a probability to every event.

The probability function is a normalized measure

1. In the discrete case (when Ω is a finite or countably infinite set), the probability of the event $E \in \mathcal{F}$ is the sum of the probabilities of the outcomes in E:

$$P(E) = \sum_{\omega \in E} P(\omega).$$

   The outcomes are events themselves with positive probabilities $P(\omega)$.

2. In the continuous case, it is an integral of the probability density function (pdf) over the event $E \in \mathcal{F}$:

$$P(E) = \int_{\omega \in E} d\mu(\omega).$$

   In this case, the outcomes are uncountable and are not events.
3. The set Ω is also an event: $\Omega \in \mathcal{F}$. The following normalization holds:

$$P(\Omega) = 1,$$

   that is, the probability of the set of all possible outcomes is equal to 1.

Given two events A, B, we can build the union $A \cup B$ and the intersection $A \cap B$:

- $A \cup B =$ set of outcomes in A or in B.
- $A \cap B =$ set of outcomes in A and in B.

Some additional technical assumptions on $\mathcal{F}$ are made in order that probabilities can be calculated properly (σ-algebra assumptions):

- If $E \in \mathcal{F}$ also its complement $\Omega \backslash E \in \mathcal{F}$.
- If $E_n \in \mathcal{F}$ for $n = 1, 2, 3, \ldots$ (possibly a countable infinite sequence of event), then $\cup_n E_n \in \mathcal{F}$, as well.

- The probability of the intersection is called joint probability:

$$P(A, B) \equiv P(A \cap B)$$

- The conditional probability is

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

- An important result is the total probability law.
- If the events $B_i, i = 1, 2, \ldots$ form a partition of Ω (i.e., $\uplus_i B_i = \Omega$ and $B_i \cap B_j = \emptyset$ for $i \neq j$), then:

$$P(A) = \sum_i P(A, B_i) = \sum_i P(A \mid B_i) P(B_i)$$

3. Define random variables, expectation, and their basic parameters.

A random variable is a measurable function X:Ω → E from a set of possible outcomes Ω to a measurable space E. It is informally described as a variable whose values depend on outcomes of a random phenomenon.

A discrete random variable X is characterized by its probability distribution

$$p_X(x_n) = P(X = x_n)$$

for $n = 1, 2, \ldots, N$ (where N may become infinity).

The expectation operator $\mathbb{E}[\cdot]$ is defined by

$$\mathbb{E}[\varphi(X)] = \sum_n \varphi(x_n) p_X(x_n)$$

for an arbitrary function $\varphi(\cdot)$.

Since the expected value of every constant is the constant itself, we obtain by definition:

$$\mathbb{E}[1] = \sum_n p_X(x_n) = 1.$$

This property holds for all probability distributions.

The mean of X is $\mu_X = \mathbb{E}[X] = \sum_n x_n p_X(x_n)$.

The second moment of X is $\mu_X^{(2)} = \mathbb{E}[X^2] = \sum_n x_n^2 p_X(x_n)$.

The variance of X is $\sigma_X^2 = \mathbb{E}[(X - \mu_X)^2] = \mu_X^{(2)} - \mu_X^2$.

The square root of the variance is called standard deviation.

4. Introduce the basic concepts on information sources.

An information source is a device that outputs (at a certain rate) a bit sequence representing any kind of information.

An information source is characterized as a sequence of discrete random variables (X_n) and the set of all possible probabilities

$$P(X_{\mathcal{S}} = x_{\mathcal{S}})$$

where $\mathcal{S}$ are all possible index sets, $X_{\mathcal{S}} \triangleq \{X_n\}_{n\in\mathcal{S}}$, $x_{\mathcal{S}} \triangleq \{x_n\}_{n\in\mathcal{S}}$, and $x_n \in \mathcal{X} \triangleq \{\xi_1, \ldots, \xi_M\}$, the source alphabet.

The general definition of an information source is often specialized by one of the following additional properties:

- Stationarity:

  $$P(X_{\mathcal{S}} = x_{\mathcal{S}}) = P(X_{\mathcal{S}+\Delta} = x_{\mathcal{S}})$$

  for every index set $\mathcal{S} = \{n_1, \ldots, n_{|\mathcal{S}|}\}$ and integer offset Δ, where

  $$\mathcal{S} + \Delta \triangleq \{n_1 + \Delta, \ldots, n_{|\mathcal{S}|} + \Delta\}.$$

  If we take $\mathcal{S} = \{n\}$, a single-index set, we can see that for a stationary source we have

  $$P(X_n = x) = P(X_{n+\Delta} = x) = p_X(x)$$

  independently of the index n.

This means that statistical properties of the information source do not change over time, it does not mean that the sequence doesn't change over time but how it changes does not change over time.

Markovianity: A Markovian source $(X_n)_{n=0}^{\infty}$ with memory L satisfies the property

$$\begin{aligned} P(X_n = x_n | X_{\{0:n-1\}} = x_{\{0:n-1\}}) \\ = P(X_n = x_n | X_{\{n-L:n-1\}} = x_{\{n-L:n-1\}}) \end{aligned}$$

In other words, X_n depends only on the previous L source symbols $X_{\{n-L:n-1\}}$.

Notation: Given two integers a, b, the notation $a : b$ represents all the integers between a and b if $a \leq b$, i.e.:

$$a, a+1, \ldots, b$$

Otherwise, if $a > b$, it represents the empty set $\emptyset$.

Independence: An independent information source satisfies the property

$$P(X_{\mathcal{S}} = x_{\mathcal{S}}) = \prod_{n \in \mathcal{S}} P(X_n = x_n)$$

for every index set $\mathcal{S}$ and all possible $x_{\mathcal{S}}$.

A stationary independent information source satisfies the property

$$P(X_{\mathcal{S}} = x_{\mathcal{S}}) = \prod_{n \in \mathcal{S}} p_X(x_n)$$

for every index set $\mathcal{S}$ and a given probability distribution $p_X(x)$.

5. Define the basic concepts from information theory until the definition of entropy.

A basic idea from information theory is assigning an information measure to events, which is what we are going to investigate in the following.

This measure is called information content and is given by

$$I(E) \triangleq \log_2 \frac{1}{P(E)} \text{ bits.}$$

Therefore:

- unlikely events possess high information content;
- common events possess low information content;
- the information content increases slowly (because of the logarithm) with the inverse of the probability.

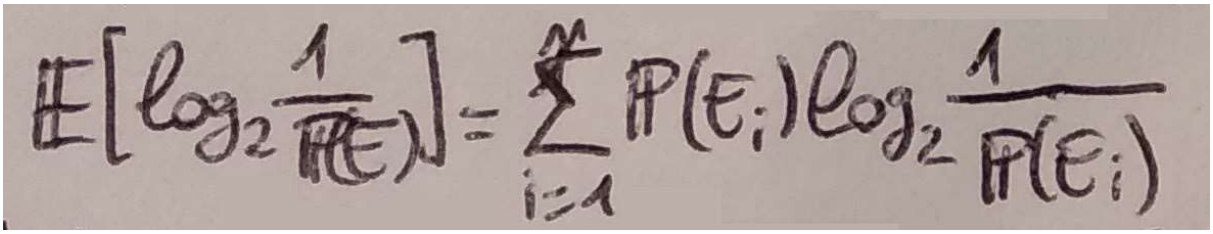
The average information content of a random variable is called entropy of the random variable.

information content tells the information quantity of a given “language” unit in a given context.

Unlikely events possess high information content while common events possess low information content, this denote the fact that a common unit is redundant therefore its information content is low, for instance as it happens in a spoken and written language for articles.

Another important measure is the average information content of a random variable, called entropy.

Given an event space where all events are independent the entropy is given by:



Given a random variable X with discrete alphabet $\mathcal{X}$ and probability distribution $p_X(x)$ defined for all $x \in \mathcal{X}$, we define its entropy as:

$$H(X) \triangleq -\sum_{x \in \mathcal{X}} p_X(x) \log_2 p_X(x) \text{ bits..}$$

Sometimes entropy is measured in nats by taking $\ln(\cdot) \equiv \log_e(\cdot)$ instead of $\log_2(\cdot)$.

If $p_X(x) = 0$, we assume $0 \log_2 0 = 0$, since $\lim_{\varepsilon \downarrow 0} \varepsilon \log_2 \varepsilon = 0$.

Since probabilities are not greater than 1, the entropy is always nonnegative: $H(X) \geq 0$.

We define the entropy function of a probability vector

$$\boldsymbol{p} = (p_1, \ldots, p_N),$$

such that $0 \leq p_i \leq 1$ and $\sum_{i=1}^{N} p_i = 1$, as follows:

$$H(\boldsymbol{p}) = H(p_1, \ldots, p_N) \triangleq \sum_{i=1}^{N} p_i \log_2 \frac{1}{p_i} = -\sum_{i=1}^{N} p_i \log_2 p_i$$

The entropy of a random variable measures the uncertainty of its value and it is maximum when the distribution is uniform (in that case H(x) = Log(2)N)

Entropy inequalities:

$$0 \leq H(\boldsymbol{p}) \leq \log_2 N$$

6. Prove in detail all the entropy inequalities.

Entropy inequalities:

$$0 \leq H(\boldsymbol{p}) \leq \log_2 N$$

Since probabilities are not greater than 1, the entropy is always nonnegative: $H(X) \geq 0$.
The lower bound is trivial since $-p_i \log_2 p_i > 0$ for $p_i > 0$ and is equal to 0 for $p_i = 0$.

$$\begin{aligned} H(\boldsymbol{p}) - \log_2 N &= \frac{1}{\ln 2} \sum_{i=1}^{N} p_i \ln \frac{1}{p_i} - \frac{\ln N}{\ln 2} \\ &= \frac{1}{\ln 2} \sum_{i=1}^{N} p_i \ln \frac{1}{p_i} - \frac{1}{\ln 2} \sum_{i=1}^{N} p_i \ln N \\ &= \frac{1}{\ln 2} \sum_{i=1}^{N} p_i \ln \frac{1}{N p_i} \leq \frac{1}{\ln 2} \sum_{i=1}^{N} p_i \left(\frac{1}{N p_i} - 1 \right) \\ &= \frac{1}{\ln 2} \sum_{i=1}^{N} \left(\frac{1}{N} - p_i \right) = \frac{1-1}{\ln 2} = 0 \end{aligned}$$

logarithmic inequality $\ln x \leq x - 1$

Given a discrete random variable X and a function $\varphi(x)$ we have the following property:

$$H(\varphi(X)) \leq H(X).$$

Proof

Let $\boldsymbol{p}$ be the probability vector corresponding to the joint distribution of X, so that $H(X) = H(\boldsymbol{p})$.
The probability vector of $\varphi(X)$ is obtained by adding subsets of probabilities from $\boldsymbol{p}$ since

$$P(\varphi(X) = z) = \sum_{\varphi(x)=z} P(X = x)$$

The inequality follows by repeatedly applying the property

$$H(p_1 + p_2, \boldsymbol{p}_3) \leq H(p_1, p_2, \boldsymbol{p}_3),$$

which is equivalent to

$$(p_1 + p_2) \log_2 \frac{1}{p_1 + p_2} \leq p_1 \log_2 \frac{1}{p_1} + p_2 \log_2 \frac{1}{p_2},$$

and

$$p_1 \log_2 \frac{p_1}{p_1 + p_2} + p_2 \log_2 \frac{p_2}{p_1 + p_2} \leq 0,$$

which holds because both arguments of the logarithms are ≤ 1.

$$\begin{aligned} H(X|Y) &= \sum_x \sum_y p_y \left(-p_{x|y} \log p_{x|y} \right) \\ &\leq \sum_x \left\{ \left(-\sum_y p_y \, p_{x|y} \right) \left(\log \sum_y p_y \, p_{x|y} \right) \right\} \\ &= \sum_x -p_x \log p_x \\ &= H(X). \end{aligned}$$

7. Define the joint and conditional entropies and show that the entropy of a function of a discrete random variable is lower than the entropy of the random variable itself.

The joint entropy of two random variables X and Y is defined as

$$H(X,Y) \triangleq -\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{XY}(x,y)\log_2 p_{XY}(x,y).$$

The concept can be extended directly to more than two random variables.

Notice that the joint entropy is invariant to permutations of random variables. For example, $H(X,Y,Z) = H(Z,Y,X)$.

The conditional entropy of two random variables X and Y is defined as

$$H(X|Y) \triangleq -\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{XY}(x,y)\log_2 p_{X|Y}(x|y).$$

Given a discrete random variable X and a function $\varphi(x)$ we have the following property: $H(\varphi(X)) \leq H(X)$.

Proof

Let $\boldsymbol{p}$ be the probability vector corresponding to the joint distribution of X, so that $H(X) = H(\boldsymbol{p})$.

The probability vector of $\varphi(X)$ is obtained by adding subsets of probabilities from $\boldsymbol{p}$ since $P(\varphi(X)=z) = \sum_{\varphi(x)=z} P(X=x)$

$$H(p_1+p_2,\boldsymbol{p}_3) \leq H(p_1,p_2,\boldsymbol{p}_3),$$

$$(p_1+p_2)\log_2\frac{1}{p_1+p_2} \leq p_1\log_2\frac{1}{p_1} + p_2\log_2\frac{1}{p_2},$$

$$p_1\log_2\frac{p_1}{p_1+p_2} + p_2\log_2\frac{p_2}{p_1+p_2} \leq 0,$$

which holds because both arguments of the logarithms are ≤ 1.

8. Define the mutual information and the relative entropy with their relationships and inequalities.

The mutual information represents as the quantity of information about a random variable X obtained observing another random variable Y.
If X and Y are independent, the observation of Y doesn't tell anything about X and the mutual information is zero.
The mutual information between X and Y is defined as

$$I(X;Y) \triangleq \sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{XY}(x,y)\log_2\frac{p_{XY}(x,y)}{p_X(x)p_Y(y)}$$

The mutual information is always nonnegative. In fact,

$$\begin{aligned}
-I(X;Y) &= \frac{1}{\ln 2}\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{XY}(x,y)\ln\frac{p_X(x)p_Y(y)}{p_{XY}(x,y)} \\
&\leqslant \frac{1}{\ln 2}\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{XY}(x,y)\left(\frac{p_X(x)p_Y(y)}{p_{XY}(x,y)}-1\right) \\
&= \frac{1}{\ln 2}\left(\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_X(x)p_Y(y) - \sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{XY}(x,y)\right) \\
&= \frac{1}{\ln 2}\left(\sum_{x\in\mathcal{X}} p_X(x)\sum_{y\in\mathcal{Y}} p_Y(y) - \sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{XY}(x,y)\right) \\
&= 0.
\end{aligned}$$

$$\begin{aligned}
I(X;Y) &= H(X)+H(Y)-H(X,Y) \\
&= H(X)-H(X|Y) \\
&= H(Y)-H(Y|X).
\end{aligned}$$

The property $I(X;Y)\geq 0$ implies the inequality
$H(X|Y) = H(X) - I(X;Y) \leq H(X)$,
conditioning reduces the entropy
Notice that the mutual information is invariant to the exchange of the random variables. For example, $I(X;Y) = I(Y;X)$.

$$I(X;Y) = \sum_x \sum_y p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)} = \sum_x \sum_y p(x,y) \left\{ \log_2 \frac{1}{p(x)} + \log_2 \frac{1}{p(y)} - \log_2 \frac{1}{p(x,y)} \right\} =$$

$$= \underbrace{\sum_x \sum_y p(x,y) \log_2 \frac{1}{p(x)}}_{= \sum_x \left\{ \sum_y p(x,y) \right\} \log_2 \frac{1}{p(x)} = \sum_x p(x) \log_2 \frac{1}{p(x)} = H(x)} + \underbrace{\sum_x \sum_y p(x,y) \log_2 \frac{1}{p(y)}}_{\text{allo stesso modo} = H(y)} - \underbrace{\sum_x \sum_y p(x,y) \log_2 \frac{1}{p(x,y)}}_{= \text{entropy of the joint distribution} = H(X,Y)} = (*)$$

$p(x)$

seconda formulazione

allo stesso modo terza formulazione

$$(*) = \underbrace{H(x) + H(y) - H(x,y)}_{\text{prima formulazione della mutual information}} = H(x) - \underbrace{[H(x,y) - H(y)]}_{\sum_x \sum_y p(x,y) \left\{ \log_2 \frac{1}{p(x,y)} - \log_2 \frac{1}{p(y)} \right\} =} = \underline{H(x) - H(x|y)} = \underline{H(y) - H(y|x)}$$

$$= \sum_x \sum_y p(x,y) \log_2 \frac{1}{\frac{p(x,y)}{p(y)}} = \sum_x \sum_y p(x,y) \log_2 \frac{1}{p(x|y)} =$$

$p(x|y)$

$= H(X|Y)$

in quanto l'entropia è sempre ≥ 0 allora $H(x) - H(x|y) \leq H(x)$ così come $H(y) - H(y|x) \leq H(y)$ per questa ragione $I(X;Y) \leq \min\{H(x), H(y)\}$

9. Define the entropy rate of a general information source and a Markovian source.

An information source is an infinite sequence of random variables therefore its entropy tends to infinity. For this reason we are not interested in the entropy of an information source as much as we are in its entropy rate defined as:

$$\bar{H} = \lim_{n\to\infty} \frac{1}{n} H(X_{1:n})$$

For a stationary independent source, we have

$$\begin{aligned} \bar{H} &= \lim_{n\to\infty} \frac{1}{n} H(X_{1:n}) \\ &= \lim_{n\to\infty} \frac{1}{n} [H(X_1) + \cdots + H(X_n)] \\ &= H(X) \end{aligned}$$

where $H(X) = H(X_1) = \cdots = H(X_n)$.

If the source is stationary, but not independent, we can show that

$$\bar{H} = \lim_{n\to\infty} \frac{1}{n} H(X_{1:n}) = \lim_{n\to\infty} H(X_n | X_{1:n-1}).$$

Markov source, is an information source whose underlying dynamics are given by a stationary finite Markov chain.

If the source is stationary Markovian with memory L, then the entropy rate is given by $\bar{H} = H(X_n|\Sigma_n)$
where Σ_n corresponds to the symbols $X_{n-L:n-1}$.
Σ_n is interpreted as the source state at time n.

In a stationary, but not independent, sequence the symbol Xn highly depends on Xn-1, a little less on Xn-2 until Xn-L. Xn is independent from Xn-L-1 etc

The calculation of $H(X|\Sigma)$ requires the state probability distribution, $P(\Sigma = \sigma)$ and the conditional distribution $P(X = x|\Sigma = \sigma)$. Then,

$$H(X|\Sigma) = \sum_{\sigma} P(\Sigma = \sigma) H(X|\Sigma = \sigma)$$

$$= \sum_{\sigma} P(\Sigma = \sigma) \sum_{x} P(X = x|\Sigma = \sigma) \log_2 \frac{1}{P(X = x|\Sigma = \sigma)}$$

The entropy rate is linked to the minimum number of bits per symbol required in source encoding, which is analyzed in the followng.

The entropy $H(X|\Sigma = \sigma)$ is easy to calculate if we have the conditional distribution of the source output symbol given the state.

10. Explain in detail the meaning of fixed-to-fixed and fixed-to-variable source coding.

The goal of source coding is the efficient transmission of data symbols emitted by a source by using the minimum average number of bits per source symbol.

To achieve this goal, the source symbols must not be independent and equiprobable over their own alphabet because in that case source coding would not be effective.

A key parameter for source coding is the entropy rate of the source, which, as we are going to see, represents a lower bound to the minimum average number of bits per symbol required by any source code.

A simple way to implement source coding is based on the fixed to fixed method which consists in representing al the source symbols in binary vectors of length m minimum integer greater or equal to log2(M) with M cardinality of the source symbols alphabet.
This way the number of bits per symbol is fixed and depends only on the cardinality of the source alphabet. No optimization is possible.
A disadvantage of the fixed-to-fixed source coding is that it may happen that part of the encoding potential is wasted, this happens when log2(M) is not a well rounded integer. For instance if the source symbols are 6 -> log2(6)=2.58 -> m=3 but m=3 at its full potential may encode 8 symbols log2(8)=3 -> waste.

A fixed-to-variable source code is characterized by

- a source alphabet $\mathcal{X} = \{\xi_1, \ldots, \xi_M\}$;
- an encoding function $\boldsymbol{c}$ mapping all symbols $x \in \mathcal{X}$ into bit vectors $\boldsymbol{c}(x)$ of length $\nu(x)$;
- encoding a symbol sequence $x_{1:N}$ produces a bit sequence obtained by concatenating the bit vectors $\boldsymbol{c}(x_i)$ for $i = 1, \ldots, N$ and corresponds to an average number of bits per symbol required equal to

$$\bar{n}(x_{1:N}) = \frac{1}{N} \sum_{n=1}^{N} \nu(x_n) = \sum_{i=1}^{M} \frac{N_i}{N} \nu(\xi_i)$$

where N_i counts how many times ξ_i is present in $x_{1:n}$.

Therefore besides grouping the source symbols as for fixed to fixed, we can improve coding efficiency by exploiting the source statistics but this requires a different approach to encoding: variable-length encoding.
In this case a function c maps all the source symbols in bit vectors c(x) of length v(x).
The goal of this source coding is minimizing the ensemble average number of bits per symbol:

By the Law of Large Numbers, it turns out that the ensemble average of the number of bits per symbols is obtained by replacing the frequencis f_i by the probabilities of emission of the source symbols, $p_i \triangleq P(X = \xi_i)$ for $i = 1, \ldots, M$.

$$\bar{n} \triangleq \sum_{i=1}^{M} p_i \nu(\xi_i).$$

This quantity can not be minimized arbitrarily because a source code in order to be useful, must be invertible.
The idea is that if the probability of a source symbol is high its correspondent bit vector should have the smallest length.

11. Describe in detail the classification of source codes.

Consider a source code $\mathcal{C}$ over an alphabet $\mathcal{X} = \{\xi_1, \ldots, \xi_M\}$ with an encoding function $\boldsymbol{c}$.

Definition.

A source code is nonsingular if $\boldsymbol{c}(\xi_i) \neq \boldsymbol{c}(\xi_j)$ for any $i \neq j$.

The previous condition is not sufficient to uniquely decode an encoded sequence but only to decode individual source symbols.

Definition.

A source code $\mathcal{C}$ is uniquely decodable if its complete extension $\mathcal{C}^*$ is nonsingular.

An additional requirement for source codes, besides unique decodability, is that always be possible to decode a source symbol sequence as soon as the last encoded bit is available.

Definition.

A source code $\mathcal{C}$ is called instantaneous or prefix-free if no codeword is a prefix of another codeword.

Definition.

The code extension of length n of $\mathcal{C}$ is the set of all possible source sequences of length n and is denoted by $\mathcal{C}^n$.

For example,

$$\begin{aligned} \mathcal{C}^1 &= \{\boldsymbol{c}(\xi_1), \ldots, \boldsymbol{c}(\xi_M)\} \\ \mathcal{C}^2 &= \{(\boldsymbol{c}(\xi_1), \boldsymbol{c}(\xi_1)), (\boldsymbol{c}(\xi_1), \boldsymbol{c}(\xi_2)), \ldots, (\boldsymbol{c}(\xi_M), \boldsymbol{c}(\xi_M))\} \end{aligned}$$

Definition.

The complete extension of $\mathcal{C}$ is defined as $\mathcal{C}^* \triangleq \bigcup_{n=1}^{\infty} \mathcal{C}^n$.

12. Illustrate the Kraft and McMillan inequalities with detailed proofs and applications.

Now let $n_i \triangleq \nu(\xi_i), i = 1, \ldots, M$ and $n_{\max} \triangleq \max_{1 \le i \le n} n_i$.

We can see that a codeword of length n_i (corresponding to a node at depth n_i generates $2^{n_{\max} - n_i}$ nodes at depth $n_{\max}$.

The prefix-free condition implies that the nodes at depth $n_{\max}$ generated by all codewords are all different.

Therefore, $\sum_{i=1}^{M} 2^{n_{\max} - n_i} \le 2^{n_{\max}}$

because the total number of nodes at depth $n_{\max}$ is $2^{n_{\max}}$.

Dividing both sides of the above inequality by $2^{n_{\max}}$ we obtain Kraft inequality.

Kraft inequality. A prefix-free code exists if and only if the following inequality is fulfilled: $\sum_{i=1}^{M} 2^{-n_i} \le 1.$

The formula itself is the same for both Kraft and McMillan but while the first focus on codes generated from trees, the second states the same inequality satisfy uniquely decodable codes.

McMillan inequality.
A uniquely decodable code satisfies **Kraft inequality**. $\sum_{i=1}^{M} 2^{-n_i} \leq 1.$

Proof. The code extension $\mathcal{C}^k$ (encoding symbols from $\mathcal{X}^k$) is uniquely decodable since $\mathcal{C}$ is uniquely decodable.

Its encoding function is

$$\boldsymbol{c}_k(x_{1:k}) = (\boldsymbol{c}(x_1), \ldots, \boldsymbol{c}(x_k))$$

Therefore, the codeword length is given by $\nu(x_{1:k}) = \nu(x_1) + \ldots + \nu(x_k)$

Then, consider the sum

$$\sum_{x_{1:k} \in \mathcal{X}^k} 2^{-\nu(x_{1:k})} = \sum_{x_1 \in \mathcal{X}, \ldots, x_k \in \mathcal{X}} 2^{-(\nu(x_1)+\ldots+\nu(x_k))}$$

$$= \sum_{x_1 \in \mathcal{X}} 2^{-\nu(x_1)} \cdots \sum_{x_k \in \mathcal{X}} 2^{-\nu(x_k)} = \left[\sum_{x \in \mathcal{X}} 2^{-\nu(x)} \right]^k$$

Denoting by N_m the number of codewords from $\mathcal{C}^k$ of length m, we have

$$\sum_{x_{1:k} \in \mathcal{X}^k} 2^{-\nu(x_{1:k})} = \sum_{m=k}^{kn_{\max}} N_m 2^{-m}$$

since the length of codewords from $\mathcal{C}$ is between 1 and $n_{\max}$ and then the length of codewords from $\mathcal{C}^k$ is between k and $kn_{\max}$.

Since $\mathcal{C}^k$ is uniquely decodable, codewords of given length m must be distinct and hence their number cannot exceed the maximum number of binary vectors of length m, so that $N_m \leq 2^m$.

As a result, we have $\left[\sum_{x \in \mathcal{X}} 2^{-\nu(x)} \right]^k \leq \sum_{m=k}^{kn_{\max}} 2^m 2^{-m} = k(n_{\max} - 1) + 1$

It follows that $\sum_{x \in \mathcal{X}} 2^{-\nu(x)} \leq [k(n_{\max} - 1) + 1]^{1/k}$ for every $k > 0$

By letting $k \to \infty$ we obtain

$$\begin{aligned}
\sum_{x \in \mathcal{X}} 2^{-\nu(x)} &\leq \lim_{k \to \infty} [k(n_{\max} - 1) + 1]^{1/k} \\
&= \lim_{k \to \infty} \exp[\ln(k(n_{\max} - 1) + 1)/k] \\
&= \exp[\lim_{k \to \infty} \ln(k(n_{\max} - 1) + 1)/k] \\
&= \exp[0] \\
&= 1
\end{aligned}$$

Therefore, from McMillan and Kraft inequalities, every uniquely decodable source code satisfies (14) and has an equivalent prefix-free code with the same codeword lengths and therefore the same $\bar{n}$.

(14)**=Kraft inequality.**

If the source code is uniquely decodable every code extension is non singular, if that is true every code is distinct.
The best uniquely decodable code (with minimum n bar) among all the uniquely decodable codes is based on the satisfaction of the inequalities. Therefore to search the best uniquely decodable code we don't have to search among every code but just among the prefix free codes.

13. Illustrate the Shannon Theorem for source coding and its proof.

Shannon's theorem provides an answer to the question if there exists a lower bound to the average number of bits per symbol required by a source code.

Shannon Theorem.

The average number of bits per symbol required by a uniquely decodable source code applied to a stationary independent source with entropy $H(X)$ satisfies the inequalities $H(X) \leq \bar{n} \leq H(X) + 1$

The lower bound comes from the application of the logarithmic inequality and the Kraft/McMillan inequality (14) :

$$H(X) - \bar{n} = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i} - \sum_{i=1}^{M} p_i n_i = \sum_{i=1}^{M} p_i \log_2 \frac{2^{-n_i}}{p_i}$$

$$\leq \log_2 e \sum_{i=1}^{M} p_i \left(\frac{2^{-n_i}}{p_i} - 1 \right) = \log_2 e \left[\sum_{i=1}^{M} 2^{-n_i} - 1 \right] \leq 0$$

The upper bound can be obtained by setting

$n_i = \lceil \log_2(1/p_i) \rceil < \log_2(1/p_i) + 1.$

Since $n_i = \lceil \log_2(1/p_i) \rceil \geq \log_2(1/p_i)$, $\sum_{i=1}^{M} 2^{-n_i} \leq \sum_{i=1}^{M} p_i = 1$ so that Kraft inequality is satisfied and the code is uniquely decodable.

Moreover, from the upper bound we get

$$\bar{n} = \sum_{i=1}^{M} p_i n_i \leq \sum_{i=1}^{M} p_i \left(\log_2 \frac{1}{p_i} + 1 \right) = H(X) + 1$$

It is interesting to notice that the logarithmic inequality identity conditions imply that $\bar{n} = H(X)$ only if all the source symbol probabilities are negative integer powers of 2:

$p_i = 2^{-n_i}, \quad i = 1, \ldots, M.$

Shannon theorem provides an operative meaning of entropy as the minimum number of bits per symbols required by a uniquely decodable source code.

14. Describe in detail the Huffman coding algorithm.

Huffman codes are prefix-free source codes that minimize the average number of bits per symbol for a stationary independent source.

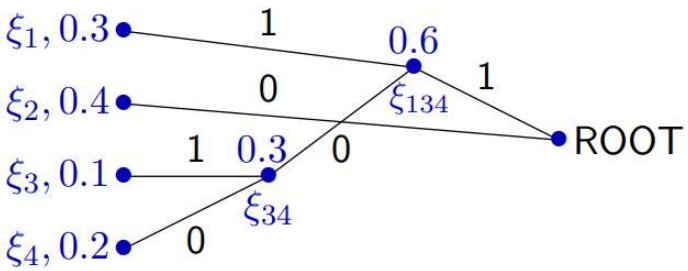
For these reason they are commonly regarded as optimal source codes.

Huffman codes are derived by construction of the code tree starting from the leaves according to the following algorithm.

Huffman algorithm. Given a stationary independent source with alphabet $\mathcal{X} = \{\xi_1, \ldots, \xi_M\}$ and probabilities $p_i = P(X_n = \xi_i)$, execute the following iterative steps:

1. Find the two lowest probabilities in the set $(p_1, \ldots, p_M)$, say p_{i_1}, p_{i_2}.
2. Set the corresponding symbols ξ_{i_1}, ξ_{i_2} as leaves of the code tree.
3. Connect the two leaves to an inner node and assign to this node a new symbol with probability $p_{i_1} + p_{i_2}$, meanwhile deleting the symbols ξ_{i_1}, ξ_{i_2}. As a result, the number of symbols to encode decreases from M to $M - 1$.
4. If the updated source alphabet contains more than 1 symbol go to 1.
5. Otherwise label the binary tree branches as 0 and 1 and find the codewords by reading the branch labels sequentially from the root (last created node) to the leaves.

Example. Construct a Huffman code for the source $\mathcal{X} = \{\xi_{1:4}\}$ with $p_1 = 0.3, p_2 = 0.4, p_3 = 0.1, p_4 = 0.2$.



Step 1: merge ξ_3, ξ_4 into ξ_{34} with probability $0.1 + 0.2 = 0.3$

Step 2: merge ξ_1, ξ_{34} into ξ_{134} with probability $0.3 + 0.3 = 0.6$

Step 2: merge ξ_2, ξ_{134} into ξ_{1234} with probability $0.4 + 0.6 = 1.0$

Resulting Huffman code:

$$\boldsymbol{c}(\xi_1) = (1, 1), \boldsymbol{c}(\xi_2) = (0), \boldsymbol{c}(\xi_3) = (1, 0, 1), \boldsymbol{c}(\xi_4) = (1, 0, 0)$$

15. Describe in detail the MAP rule.

An ideal channel reproduces the input at the output: Y = X.
A real channel output is not always equal to the input. If it is not, we have a channel error.
In order to estimate the channel input X from the channel output Y, known at the receiver, the Maximum A Posteriori (MAP) probability rule is usually implemented:

$$\widehat{X} = \arg\max_{x \in \mathcal{X}} P(X = x | Y = y).$$

$$P(X = x | Y = y) = \frac{P(X = x) P(Y = y | X = x)}{\sum_{\check{x} \in \mathcal{X}} P(X = \check{x}) P(Y = y | X = \check{x})}$$

The rule consists in calculating the conditional probabilities of the input symbol given the output one and then choosing the X that maximize this value.

The receiver operation MAP decision or detection minimizes the error probability when decisions are taken on a symbol-by-symbol basis.

Channel codes are used to minimize the error probability.
They are based on the transmission of long symbol sequences chosen with appropriate design criteria.

16. Illustrate the concept of channel codes and Shannon's Theorem for channel coding (theoretical statement and practical application).

Channel codes are used to minimize the error probability.

They are based on the transmission of long symbol sequences chosen with appropriate design criteria.

An $[M, n]$ channel code over $\mathcal{X}^n$, $\mathcal{C}$, is an invertible mapping $\mathcal{W} \to \mathcal{X}^n$ between a set of messages $\mathcal{W} = \{1, \ldots, M\}$ and the set of different n-tuples $(x_1, \ldots, x_n) \in \mathcal{X}^n$, which are called code words.

The set of all possible code words is called the codebook.

Transmitting one message from a set of M possible messages is equivalent to transmitting $\log_2 M$ bits.

For example, if $M = 4$, we can make the following associations:

$$1 \leftrightarrow (0,0) \qquad 2 \leftrightarrow (0,1) \qquad 3 \leftrightarrow (1,0) \qquad 4 \leftrightarrow (1,1)$$

We can see that the more the hamming distance between the codewords, the lower the error probability. The error is not detectable if one codeword's bit changes making that codeword become another codeword.
A codebook as the one in the example is a possible choise but it's not wise since the codebook coincides with the set of all possible combinations of two bits.

We define the code rate as:

$$R = \frac{\log_2 M}{n} \; \frac{\text{information bits}}{\text{channel symbols}}.$$

Since M cannot be larger than the number of all possible words from $\mathcal{X}^n$ (i.e., $|\mathcal{X}|^n$), we have the inequality

$$R \leq \frac{\log_2(|\mathcal{X}|^n)}{n} = \log_2 |\mathcal{X}|.$$

A binary channel code with $M = 2^k$ and $\mathcal{X} = \{0, 1\}$ is called an (n, k) code.

Its rate is $R = \log_2(2^k)/n = k/n \leq \log_2 |\mathcal{X}| = 1$.

Shannon Theorem for communication channels

- Consider a communication channel with conditional probability distribution $p_{Y|X}(y|x)$.
- Assume that sequential channel transmission is conditionally independent:

$$\begin{aligned} P(Y_1 = y_1, \ldots, Y_n = y_n \mid X_1 = x_1, \ldots, X_n = x_n) \\ = \prod_{i=1}^{n} p_{Y|X}(y_i|x_i) \end{aligned}$$

  for every $n \geq 1$.

Shannon Theorem proves that,

- given a target code rate R smaller than the channel capacity

$$\boxed{C \triangleq \max_{p_X(x)} I(X;Y)},$$

- there is a code sequence $\mathcal{C}_n = \{\boldsymbol{x}_n(m) \in \mathcal{X}^n, m \in \{1 : M_n\}\}$ with code rates

$$R_n = \frac{\log_2 M_n}{n} \to R$$

  as $n \to \infty$, such that the maximum error probability

$$\left\{ P_{n,\max}(e) \triangleq \max_{1 \leq m \leq M_n} P\Big\{ \mathcal{D}_n(\boldsymbol{y}) \neq m \mid \boldsymbol{x}_n(m) \text{ transmitted} \Big\} \right\} \to 0$$

  as $n \to \infty$ for some decoding function $\mathcal{D}_n : \mathcal{Y}^n \mapsto \{1 : M_n\}$.

According to Shannon Theorem, by using very long channel codes, we can transmit reliably (i.e., with vanishing error probability) up to a code rate equal to the channel capacity C.

One codeword of n channel symbols carries $\approx nC$ information bits.

If T is the time required to transmit one channel symbol, the maximum information bit rate is

$$\frac{\text{N. of information bits/codeword}}{\text{Time to transmit one codeword}} = \frac{nC}{nT} = \frac{C}{T}\ \frac{\text{bit}}{\text{s}}$$

Shannon Theorem has a converse, too.

The converse says that if the error probability of a code sequence $\mathcal{C}_n$ with limit rate R tends to zero as $n \to \infty$, then it must be $R \leq C$.

Equivalently, if $R > C$, there exists no code sequence such that the error probability tends to zero as $n \to \infty$.

In other words, if $R > C$, it is impossible to transmit reliably.

Therefore, the capacity is a hard limit in the sense that the achievable limiting error probability passes from 0 to 1 when the rate R crosses the capacity limit C.

Shannon Theorem outlines the role of channel capacity as the ultimate information bit rate which can be achieved over a channel with vanishing error probability.

17. Illustrate the channel capacity in general and when the channel matrix has rows permutations of each other.

$$C \triangleq \max_{p_X(x)} I(X;Y)$$

The channel capacity is the ultimate information bit rate which can be achieved over a channel with vanishing error probability.
The channel capacity C depends on M × N channel matrix

$$\boldsymbol{P} \triangleq \Big(P(Y=y_j|X=x_i)\Big)_{i,j=1}^{M,N}, \qquad M=|\mathcal{X}|, N=|\mathcal{Y}|$$

Where M are all the possible messages (cardinality of the codebook) and N..

C can be calculated analytically only in a limited number of cases, depending on the characteristics of P:
if the rows and columns of P are permutations of each other;
in some cases when only the rows of P are permutations of each other;
in the case M = N = 2.

- **Rows of $\boldsymbol{P}$ permutations of each other.**
  In this case,

$$\begin{aligned} H(Y|X) &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{XY}(x,y)\log_2 p_{Y|X}(y|x) \\ &= -\sum_{x\in\mathcal{X}} p_X(x)\sum_{y\in\mathcal{Y}} p_{Y|X}(y|x)\log_2 p_{Y|X}(y|x) \\ &= \sum_{x\in\mathcal{X}} p_X(x)H(Y|X=x) \end{aligned}$$

  where $H(Y|X=x) \triangleq -\sum_{y\in\mathcal{Y}} p_{Y|X}(y|x)\log_2 p_{Y|X}(y|x)$.

- The entropy $H(Y|X=x)$, which is the entropy function $H(\boldsymbol{\pi}_x)$, i.e., of the row of $\boldsymbol{P}$ corresponding to $X=x$.
- If the rows of $\boldsymbol{P}$ are permutations of each other, $H(Y|X=x)$ is constant for every row, say $H(\boldsymbol{\pi})$, where $\boldsymbol{\pi}$ is any of the $\boldsymbol{\pi}_x$, for example the first row of $\boldsymbol{P}$.
- Then,

$$H(Y|X) = \sum_{x \in \mathcal{X}} p_X(x) H(\boldsymbol{\pi}) = H(\boldsymbol{\pi}).$$

- As a result, since $I(X;Y) = H(Y) - H(Y|X)$,

$$C = \left\{ \max_{p_X(x)} H(Y) \right\} - H(\pi)$$

18. Illustrate and derive in detail the capacity of a strictly symmetric discrete channel and of the binary symmetric channel.

$$C \triangleq \max_{p_X(x)} I(X;Y)$$

The channel capacity is the ultimate information bit rate which can be achieved over a channel with vanishing error probability.
The channel capacity C depends on M × N channel matrix

$$\boldsymbol{P} \triangleq \Big(P(Y=y_j|X=x_i)\Big)_{i,j=1}^{M,N}, \qquad M=|\mathcal{X}|, N=|\mathcal{Y}|$$

- **Rows and columns of $\boldsymbol{P}$ permutations of each other (strictly symmetric channel).**

  In this case, we can maximize $H(Y)$ by showing that the equiprobable input distribution gives an equiprobable output distribution, maximizing $H(Y)$.
- Since the columns of $\boldsymbol{P}$ are permutations of each other, $\sum_{x\in\mathcal{X}} p_{Y|X}(y|x)$ doesn't depend on the column index $y \in \mathcal{Y}$.
- Hence, if $p_X(x) = \frac{1}{M}$,

$$p_Y(y) = \sum_{x\in\mathcal{X}} p_X(x)p_{Y|X}(y|x) = \frac{1}{M}\sum_{x\in\mathcal{X}} p_{Y|X}(y|x)$$

- As a result, since $I(X;Y) = H(Y) - H(Y|X)$,

$$C = \Big\{\max_{p_X(x)} H(Y)\Big\} - H(\pi)$$

$H(Y)$ reaches the maximum when both X and Y are equiprobable, and the maximum is $\log_2|\mathcal{Y}|$.

Thus, the capacity is $C = \boxed{\log_2|\mathcal{Y}| - H(\boldsymbol{\pi})}$ where $\boldsymbol{\pi}$ is any row of $\boldsymbol{P}$

Binary Symmetric Channel (BSC)

The channel matrix is $\boldsymbol{P} = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$

with $\mathcal{X} = \mathcal{Y} = \{0, 1\}$.

The parameter p is called error probability of the channel.

This matrix satisfies the strict symmetry condition stated above

Then, the channel capacity is

$$C = \log_2 |\mathcal{Y}| - H(1-p, p) = \boxed{1 - H_b(p)}$$

19. Derive in detail the capacity of a binary input symmetric output channel and of the binary erasure channel.

$$C \triangleq \max_{p_X(x)} I(X;Y)$$

The channel capacity is the ultimate information bit rate which can be achieved over a channel with vanishing error probability.
The channel capacity C depends on a channel matrix P.

Special case: Binary input symmetric output channel

$$\boldsymbol{P} = \begin{pmatrix} p_1 & p_2 & \cdots & p_{N-1} & p_N \\ p_N & p_{N-1} & \cdots & p_2 & p_1 \end{pmatrix}$$

We assume $p_X(\xi_1) = \alpha$ and $p_X(\xi_2) = \bar{\alpha} \triangleq 1 - \alpha$.
We can see that $p_Y(\eta_j) = \alpha p_j + \bar{\alpha} p_{N+1-j}$.
Then, applying the entropy inequality (2),

$$\begin{aligned} H(Y) &= H(\alpha p_1 + \bar{\alpha} p_N, \alpha p_2 + \bar{\alpha} p_{N-1}, \ldots, \alpha p_{N-1} + \bar{\alpha} p_2, \alpha p_N + \bar{\alpha} p_1) \\ &= H(\alpha p_1 + \bar{\alpha} p_N, \alpha p_N + \bar{\alpha} p_1, \alpha p_2 + \bar{\alpha} p_{N-1}, \alpha p_{N-1} + \bar{\alpha} p_2, \ldots) \\ &\leq H\left(\frac{p_1 + p_N}{2}, \frac{p_1 + p_N}{2}, \frac{p_2 + p_{N-1}}{2}, \frac{p_2 + p_{N-1}}{2}, \ldots\right) \end{aligned}$$

The upper bound is attained when $\alpha = \frac{1}{2}$ and is therefore the maximum $H(Y)$.
The capacity is

$$\begin{aligned} C = &H\left(\frac{p_1 + p_N}{2}, \frac{p_1 + p_N}{2}, \frac{p_2 + p_{N-1}}{2}, \frac{p_2 + p_{N-1}}{2}, \ldots\right) \\ &- H(p_1, \ldots, p_N) \end{aligned}$$

For example (Binary Erasure Channel, BEC), if

$$\boldsymbol{P} = \begin{pmatrix} 1-p & p & 0 \\ 0 & p & 1-p \end{pmatrix},$$

we have

$$\begin{aligned} C &= H\left(\frac{1-p}{2}, \frac{1-p}{2}, p\right) - H(1-p, p) \\ &= 2 \times \frac{1-p}{2} \log_2 \frac{2}{1-p} + p \log_2 \frac{1}{p} \\ &\quad - (1-p) \log_2 \frac{1}{1-p} - p \log_2 \frac{1}{p} \\ &= 1-p \end{aligned}$$

20. Derive in detail the capacity of the binary asymmetric channel and of the Z channel.

$$C \triangleq \max_{p_X(x)} I(X;Y)$$

The channel capacity is the ultimate information bit rate which can be achieved over a channel with vanishing error probability.
The channel capacity C depends on a channel matrix P.

Binary Asymmetric Channel (BAC).

$$\boldsymbol{P} = \begin{pmatrix} 1-p & p \\ q & 1-q \end{pmatrix}$$

Here, p, q are the error probabilities corresponding to the transmission of ξ_1, ξ_2, respectively.

Let $\alpha \triangleq P(X = \xi_1)$, $\bar{\alpha} \triangleq 1 - \alpha$, $\bar{p} \triangleq 1 - p$, and $\bar{q} \triangleq 1 - q$.

We express the mutual information as a function of α:

$$\begin{aligned} \varphi(\alpha) &\triangleq I(X;Y) \\ &= H(Y) - H(Y|X) \\ &= H_b(\alpha\bar{p} + \bar{\alpha}q) - \alpha H_b(p) - \bar{\alpha}H_b(q) \end{aligned}$$

Then, we calculate the derivative of $H_b(p)$:

$$\begin{aligned} H_b'(p) &= \frac{1}{\ln 2}\frac{d}{dp}\Big(-p\ln p - (1-p)\ln(1-p)\Big) \\ &= \frac{1}{\ln 2}\left(-\ln p - \frac{p}{p} + \ln(1-p) - \frac{1-p}{1-p}(-1)\right) \\ &= \log_2\frac{1-p}{p} \Longrightarrow p = \frac{1}{1+2^{H_b'(p)}} \end{aligned}$$

Then we solve $\varphi'(\alpha) = 0$ to find the maximum:

$$\varphi'(\alpha) = H_b'(\alpha\bar{p} + \bar{\alpha}q)(\bar{p} - q) - H_b(p) + H_b(q) = 0$$

The solution, for $\bar{p} \neq q$, with $\beta \triangleq 2^{(H_b(p)-H_b(q))/(\bar{p}-q)}$, is

$$\alpha\bar{p} + \bar{\alpha}q = \frac{1}{1+\beta}$$

Hence,

$$\alpha = \frac{\frac{1}{1+\beta} - q}{\bar{p} - q}, \qquad \bar{\alpha} = \frac{\bar{p} - \frac{1}{1+\beta}}{\bar{p} - q}$$

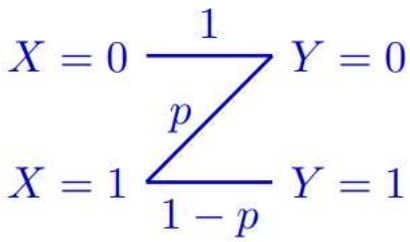
Then,

$$\begin{aligned} C &= H_b\left(\frac{1}{1+\beta}\right) - \frac{\frac{1}{1+\beta} - q}{\bar{p} - q}H_b(p) - \frac{\bar{p} - \frac{1}{1+\beta}}{\bar{p} - q}H_b(q) \\ &= H_b\left(\frac{1}{1+\beta}\right) - \frac{1}{1+\beta}\boxed{\frac{H_b(p) - H_b(q)}{\bar{p} - q}} + \frac{qH_b(p) - \bar{p}H_b(q)}{\bar{p} - q} \\ &= \frac{\log_2(1+\beta)}{1+\beta} + \frac{\beta}{1+\beta}\log_2\frac{1+\beta}{\beta} - \boxed{\frac{\log_2\beta}{1+\beta}} + \frac{qH_b(p) - \bar{p}H_b(q)}{\bar{p} - q} \\ &= \log_2(1+\beta^{-1}) + \frac{qH_b(p) - \bar{p}H_b(q)}{\bar{p} - q} \\ &= \boxed{\log_2\left(1 + 2^{(H_b(q)-H_b(p))/(\bar{p}-q)}\right) + \frac{qH_b(p) - \bar{p}H_b(q)}{\bar{p} - q}} \end{aligned}$$

Z Channel.

The probability matrix is

$$\boldsymbol{P} = \begin{pmatrix} 1 & 0 \\ p & 1-p \end{pmatrix} \quad \Longrightarrow \quad$$



We can apply the result obtained for the BAC with $\bar{p} \to 1, q \to p$:

$$\begin{aligned} C &= \log_2\left(1 + 2^{(H_b(q) - H_b(p))/(\bar{p}-q)}\right) + \frac{qH_b(p) - \bar{p}H_b(q)}{\bar{p}-q} \\ &= \log_2\left(1 + 2^{(H_b(p) - H_b(0))/(1-p)}\right) + \frac{pH_b(0) - 1 \cdot H_b(p)}{1-p} \\ &= \log_2\left(1 + 2^{H_b(p)/(1-p)}\right) - \frac{H_b(p)}{1-p} \\ &= \boxed{\log_2\left(1 + 2^{-H_b(p)/(1-p)}\right)} \end{aligned}$$

21. Illustrate the Blahut-Arimoto Theorem and prove the basic result supporting this theorem.

It is always possible to calculate numerically the capacity of a discrete channel by using the Blahut-Arimoto algorithm.
The algorithm is iterative and based on a double maximization.

Theorem.

The capacity of a discrete channel can be obtained as a double maximization:

$$C = \max_{p_X(x)} \max_{q(x|y)} \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p_{XY}(x,y) \log_2 \frac{q(x|y)}{p_X(x)}$$

where $q(x|y)$ is an arbitrary probability distribution satisfying $\sum_{x\in\mathcal{X}} q(x|y) = 1$.

PROOF

Following the definition of channel capacity, it is sufficient to prove the inner maximization result:

$$I(X;Y) = \max_{q(x|y)} \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p_{XY}(x,y) \log_2 \frac{q(x|y)}{p_X(x)}.$$

Since $I(X;Y) = H(X) - H(X|Y)$, it is sufficient to prove that the maximum is attained when $q(x|y) = p_{X|Y}(x|y)$.

To this purpose we consider the difference

$$\sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p_{XY}(x,y) \log_2 \frac{q(x|y)}{p_X(x)} - I(X;Y)$$

We have

$$\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{XY}(x,y)\log_2\frac{q(x|y)}{p_X(x)} - I(X;Y)$$

$$= \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{XY}(x,y)\log_2\frac{q(x|y)}{p_X(x)} - \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{XY}(x,y)\log_2\frac{p_{X|Y}(x|y)}{p_X(x)}$$

$$= \log_2 e \sum_{y\in\mathcal{Y}} p_Y(y) \sum_{x\in\mathcal{X}} p_{X|Y}(x|y)\ln\frac{q(x|y)}{p_{X|Y}(x|y)}$$

$$\leq \log_2 e \sum_{y\in\mathcal{Y}} p_Y(y) \sum_{x\in\mathcal{X}} p_{X|Y}(x|y)\left(\frac{q(x|y)}{p_{X|Y}(x|y)} - 1\right)$$

$$= \log_2 e \sum_{y\in\mathcal{Y}} p_Y(y) \sum_{x\in\mathcal{X}} \Big(q(x|y) - p_{X|Y}(x|y)\Big) = 0$$

Therefore, $I(X;Y) = \max\limits_{q(x|y)} \sum\limits_{x\in\mathcal{X}}\sum\limits_{y\in\mathcal{Y}} p_{XY}(x,y)\log_2\dfrac{q(x|y)}{p_X(x)}$

and the maximum is attained when $q(x|y) = p_{X|Y}(x|y)$

for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

22. Illustrate in detail the data-processing inequality.

Consider the cascade of two channels represented by $X \to Y \to Z$ forming a Markov chain, i.e., such that

$$p_{Z|XY}(z|x,y) = p_{Z|Y}(z|y).$$

The data-processing inequality is as follows:

$$I(X;Z) \leq I(X;Y).$$

Information flow progressively decrease along the channel series, as a real life analogous is a water-leaking pipeline.
Lost information can not be retrieved as shown by the inequality

We shall use the conditional mutual information

$$\begin{aligned} I(X;Y|Z) &= H(X|Z) + H(Y|Z) - H(X,Y|Z) \\ &= H(X|Z) - H(X|Y,Z) \\ &= H(Y|Z) - H(Y|X,Z). \end{aligned}$$

Proof of the Data-Processing Inequality.

We first notice from the chain rule of mutual information that

$$\begin{aligned} I(X;Y,Z) &= H(X) - H(X|Y,Z) \\ &= H(X) - H(X|Y) + H(X|Y) - H(X|Y,Z) \\ &= I(X;Y) + I(X;Z|Y) \end{aligned}$$

In a similar way, we can see that $I(X;Y,Z) = I(X;Z) + I(X;Y|Z)$

Thus, we have the identity

$$I(X;Y) + I(X;Z|Y) = I(X;Z) + I(X;Y|Z)$$

Next we show that X and Z are conditionally independent given Y, i.e.,

$$p_{XZ|Y}(x,z|y) = p_{X|Y}(x|y)p_{Z|Y}(z|y).$$

This is proved from the Markov property $p_{Z|XY}(z|x,y) = p_{Z|Y}(z|y)$:

$$\begin{aligned} p_{XZ|Y}(x,z|y) &= \frac{p_{XZY}(x,z,y)}{p_Y(y)} \\ &= \frac{p_{XZY}(x,z,y)}{p_{XY}(x,y)}\frac{p_{XY}(x,y)}{p_Y(y)} \\ &= p_{Z|XY}(z|x,y)p_{X|Y}(x|y) \\ &= p_{Z|Y}(z|y)p_{X|Y}(x|y) \end{aligned}$$

The conditional independence property implies that

$$\begin{aligned} I(X;Z|Y) &= H(X|Y) + H(Z|Y) - H(X,Z|Y) \\ &= \mathbb{E}\left[\log_2\left(\frac{p_{XZ|Y}(X,Z|Y)}{p_{X|Y}(X|Y)p_{Z|Y}(Z|Y)}\right)\right] \\ &= \mathbb{E}[\log_2 1] \\ &= 0. \end{aligned}$$

The previous result, $I(X;Z|Y) = 0$, implies that

$$I(X;Y) + 0 = I(X;Z) + I(X;Y|Z)$$

Since $I(X;Y|Z) \geq 0$, we get

$$I(X;Z) \leq I(X;Y).$$

23. Illustrate the concept of discretization for continuous random variables and its detailed application leading to the differential entropy.

Smooth = can be derived infinite times

Let X be a continuous random variable, i.e., characterized by a pdf which is a smooth function $f_X(x)$.

We can discretize X at the precision level Δ as follows.

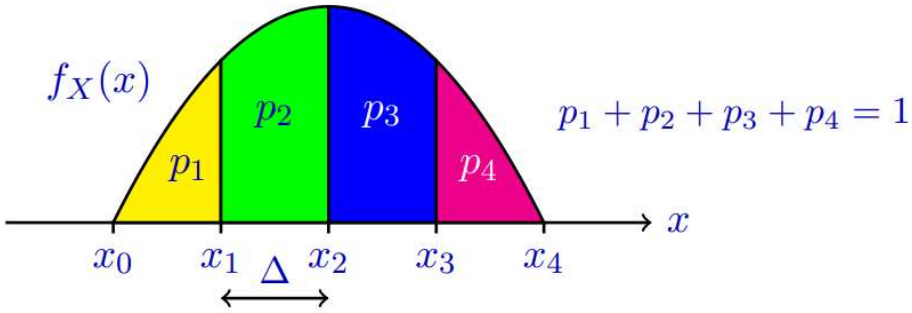
Let us assume that $P(x_0 < X < x_N) = 1$, i.e., X is included in the interval (x_0, x_N) with probability 1.

Then, we map X to a discrete random variable, X_Δ, defined by the following probability distribution:

$$p_i \triangleq P(X_\Delta = i) = P(x_i < X < x_{i+1}) = \int_{x_i}^{x_{i+1}} f_x(x)dx$$

with $\Delta \triangleq x_{i+1} - x_i$ for $i = 0, 1, \ldots, N-1$.

For example, consider the case $N = 4$ and the pdf given in the following diagram:



We can see that $p_1 = \int_{x_0}^{x_1} f_X(x)dx$ and so on, as in the definition of the previous slide.

Now, we recall the approximation of an integral as a sum:

$$\int_{x_0}^{x_N} g(x)dx \approx \Delta \times \sum_{i=1}^{N} g(\xi_i)$$

Here, ξ_i is the middle point of the interval (x_{i-1}, x_i) for $i = 1, 2, \ldots, N$.

The approximation is accurate if the function $g(x)$ is almost constant over each interval (x_{i-1}, x_i).

This condition is satisfied if the size of the intervals, Δ, is sufficiently small, that is, when $\Delta \to 0^+$.

Let us use the previous approximation to calculate the probabilities of X_Δ:

$$p_i = \int_{x_{i-1}}^{x_i} f_X(x)dx \approx \Delta f_X(\xi_i), \qquad i = 1, \ldots, N$$

Recall that ξ_i is the middle point of the interval (x_{i-1}, x_i), so that

$$\xi_i = \frac{x_{i-1} + x_i}{2}$$

Moreover, we have

$$\begin{aligned}
h(X) &\triangleq \int_{x_0}^{x_N} f_X(x) \log_2 \frac{1}{f_X(x)} dx \\
&\approx \Delta \sum_{i=1}^{N} f_X(\xi_i) \log_2 \frac{1}{f_X(\xi_i)} \\
&= \sum_{i=1}^{N} (\Delta f_X(\xi_i)) \log_2 \frac{\Delta}{\Delta f_X(\xi_i)} \\
&\approx \sum_{i=1}^{N} p_i \log_2 \Delta + \sum_{i=1}^{N} p_i \log_2 \frac{1}{p_i} \\
&= \log_2 \Delta + H(p_1, p_2, \ldots, p_N) \qquad = \log_2 \Delta + H(X_\Delta)
\end{aligned}$$

We shall call $h(X)$ the differential entropy of X:

$$h(X) \triangleq -\int f_X(x) \log_2 f_X(x) dx = \mathbb{E}[-\log_2 f_X(X)]$$

24. Illustrate in detail the properties of the mutual information between continuously-distributed random variables.

In this case, to approximate the joint entropy, we need a two-dimensional integral approximation:

$$\int_{x_0}^{x_N} \int_{y_0}^{y_N} g(x,y) dx dy \approx \Delta^2 \sum_{i=1}^{N} \sum_{j=1}^{N} g(\xi_i, \eta_j)$$

Proceding as in the one-dimensional case, we get the joint differential entropy

$$\begin{aligned} h(X,Y) &\triangleq \int_{x_0}^{x_N} \int_{y_0}^{y_N} f_{XY}(x,y) \log_2 \frac{1}{f_{XY}(x,y)} dx dy \\ &\approx \log_2(\Delta^2) + H(X_\Delta, Y_\Delta) \end{aligned}$$

Hence,

$$H(X_\Delta, Y_\Delta) \approx h(X,Y) - \log_2(\Delta^2)$$

with increasing accuracy as $\Delta \to 0^+$.

Now, we can use the one- and two-dimensional approximation and obtain

$$\begin{aligned} I(X_\Delta, Y_\Delta) &= H(X_\Delta) + H(Y_\Delta) - H(X_\Delta, Y_\Delta) \\ &\approx h(X) - \log_2 \Delta + h(Y) - \log_2 \Delta \\ &\quad - [h(X,Y) - \log_2(\Delta^2)] \\ &= h(X) + h(Y) - h(X,Y) \end{aligned}$$

since $-\log_2 \Delta - \log_2 \Delta + \log_2(\Delta^2) = 0$.

In view of the previous analysis, the mutual information between two continuous random variables X and Y is defined as

$$I(X;Y) \triangleq h(X) + h(Y) - h(X,Y)$$

25. Derive in detail the capacity of the additive Gaussian channel and the differential entropy inequality for Gaussian random variables.

The additive Gaussian channel is represented by the following channel equation: $Y = X + Z$

X is the channel input
Y is the output
Z is the additive noise
when the noise is Gaussian-distributed the channel is Gaussian
$Z \sim \mathcal{N}(0, N)$, where N is the average noise power
The channel input and the noise are always independent
The capacity is calculated under an average input power constraint:

$$\mathbb{E}[X^2] \leq S$$

To derive the capacity of the additive Gaussian channel we need some properties of the differential entropy:
For any constant a, $h(a + X) = h(X)$
For any two independent random variables X, Z, we have $h(X + Z|X) = h(X)$
If σ_X^2 is the variance of X, then $h(X) \leq h(X_G) = \frac{1}{2}\log_2(2\pi e\,\sigma_X^2)$
where X_G is a Gaussian random variable with zero mean and variance σ_X^2

Now, we apply the previous results to calculate the capacity of the additive Gaussian channel $Y = X + Z$, where $Z \sim \mathcal{N}(0, N)$, under the average input power contrraint $\mathbb{E}[X^2] \leq S$.

We can write the mutual information as $I(X;Y) = h(Y) - h(Z)$

We search the maximum $I(X;Y)$ for all possible $f_X(x)$ with the constraint $\mathbb{E}[X^2] \leq S$.

Since $Y = X + Z$ and Z is independent of X,

$$\begin{aligned} \sigma_Y^2 \leq \mathbb{E}[Y^2] &= \mathbb{E}[(X+Z)^2] \\ &= \mathbb{E}[X^2] + 2\mathbb{E}[XZ] + \mathbb{E}[Z^2] \\ &= \mathbb{E}[X^2] + 2\mathbb{E}[X]\mathbb{E}[Z] + N \\ &= \mathbb{E}[X^2] + N \\ &\leq S + N \end{aligned}$$

The upper limit is attained when $\mathbb{E}[X] = 0$ and $\mathbb{E}[X^2] = S$.
In fact, if $\mathbb{E}[X] = 0$, $\mathbb{E}[Y] = \mathbb{E}[X] + \mathbb{E}[Z] = 0$, so that $\sigma_Y^2 = \mathbb{E}[Y^2]$
Moreover, if $\mathbb{E}[X] = 0$, $\mathbb{E}[X^2] = \sigma_X^2 = S$.

Therefore we obtain $h(Y) \leq h(Y_G) = \frac{1}{2}\log_2(2\pi e(S+N))$

where the maximum is attained when $X \sim \mathcal{N}(0, S)$ since the sum of two Gaussian random variables, $Y = X + Z$, is Gaussian.

This condition, $X \sim \mathcal{N}(0, S)$, maximizes $h(Y)$ and hence $I(X;Y) = h(Y) - h(Z)$ over the possible input distributions $f_X(x)$ satisfying the power constraint $\mathbb{E}[X^2] \leq S$.

Then, the channel capacity is achieved when $X \sim \mathcal{N}(0, S)$ and its value is

$$C = \frac{1}{2}\log_2(2\pi e(S+N)) - \frac{1}{2}\log_2(2\pi eN) = \boxed{\frac{1}{2}\log_2\left(1 + \frac{S}{N}\right)}$$

$$C = \max_{f_X : \mathbb{E}[X^2] \leq S} I(X;Y) = \left\{\max_{f_X : \mathbb{E}[X^2] \leq S} h(Y)\right\} - \underbrace{h(Y|X)}_{h(Z)} =$$

$$= \frac{1}{2}\log_2\left(2\pi e(S+N)\right) - \frac{1}{2}\log_2(2\pi eN) = \frac{1}{2}\log_2\left(1 + \frac{S}{N}\right)$$

Proof of $h(X) \leq h(X_G) = \frac{1}{2}\log_2(2\pi \mathrm{e}\,\sigma_X^2)$.

Since $h(X) = h(X - \mathbb{E}[X])$ we assume (without loss of generality) that X and X_G have both zero mean.

Moreover, they have the same variance: $\sigma_X^2 = \mathbb{E}[X^2] = \mathbb{E}[X_G^2]$

Then, let $f(x)$ and $g(x)$ be the pdfs of X and X_G, so that

$$g(x) = \frac{1}{\sqrt{2\pi\sigma_X^2}} \mathrm{e}^{-x^2/(2\sigma_X^2)}$$

$$\begin{aligned}
\mathbb{E}[\log_2 g(X)] &= \int f(x) \log_2 g(x) dx \\
&= \mathbb{E}[-\log_2(\sqrt{2\pi\sigma^2}) - X^2 \log_2 \mathrm{e}\,/(2\sigma^2)] \\
&= \mathbb{E}[-\log_2(\sqrt{2\pi\sigma^2}) - X_G^2 \log_2 \mathrm{e}\,/(2\sigma^2)] \\
&= \int g(x) \log_2 g(x) dx \\
&= \mathbb{E}[\log_2 g(X_G)]
\end{aligned}$$

By using the logarithmic inequality $\ln(1+x) \leq x$ we can see that

$$\begin{aligned}
h(X) - h(X_G) &= \mathbb{E}[-\log_2 f(X)] - \mathbb{E}[-\log_2 g(X_G)] \\
&= \mathbb{E}[-\log_2 f(X)] - \mathbb{E}[-\log_2 g(X)] \\
&= \mathbb{E}\left[\log_2 \frac{g(X)}{f(X)}\right] \\
&\leq \log_2 \mathrm{e} \times \mathbb{E}\left[\frac{g(X)}{f(X)} - 1\right] \\
&= \log_2 \mathrm{e} \times \left\{\int \frac{g(x)}{f(x)} f(x) dx - 1\right\} \\
&= \log_2 \mathrm{e} \times \left\{\int g(x) dx - 1\right\} = 0
\end{aligned}$$

$$
\begin{aligned}
h(X_G) &= -\mathbb{E}[\log_2 g(X_G)] \\
&= \mathbb{E}[+\log_2(\sqrt{2\pi\sigma^2}) + X_G^2 \log_2 \mathrm{e}\,/(2\sigma^2)] \\
&= \frac{1}{2}\log_2(2\pi \mathrm{e}\,\sigma^2).
\end{aligned}
$$

26. Illustrate in detail the weighted water-filling algorithm.

Some transmission techniques correspond to the simultaneous transmission of a single data stream multiplexed over a number of independent (parallel) additive Gaussian channels (as an example, Consider a set of K independent additive Gaussian channels
$Y_k = A_k X_k + Z_k, \qquad k = 1, \ldots, K$
The transmitted symbols are independent special complex Gaussian distributed: $X_k \sim \mathcal{N}(0, P_k)$.

The noise samples are also independent special complex Gaussian distributed: $Z_k \sim \mathcal{N}(0, N_k)$.

The channel gains A_k are constant and the channels are called parallel Gaussian channels.

According to this assumptions, by direct application of the Shannon capacity formula, the achievable rate of each channel is

$R_k = \frac{1}{2}\log_2(1 + \mathrm{SNR}_k) = \frac{1}{2}\log_2(1 + \gamma_k P_k)$ where $\gamma_k \triangleq A_k^2/N_k$

In some situations, we are interested in the maximum weighted sum rate $R \triangleq \sum_{k=1}^{K} w_k \log_2(1+\gamma_k P_k)$ with $w_k > 0$ for $k = 1, \ldots, K$ under a constraint on the total power: $\sum_{k=1}^{K} P_k \leq P$

In other words, our goal is to find the power allocation $(P_k)_{k=1}^K$ which maximizes the weighted sum rate R under the total power constraint.

To solve this optimization problem we note that the transmitted powers are nonnegative numbers: $P_k \geq 0$.

The power constraint inequality can be replaced by a strict identity since the maximum weighted sum rate is surely achieved when $\sum_{k=1}^{K} P_k = P$

To implement the equality constraint we assume $P_k = P\dfrac{x_k^2}{\sum_{\ell=1}^{K} x_\ell^2}$ where $x_k \in \mathbb{R}$.

Defining the function $\varphi(x_1, \ldots, x_K) \triangleq \sum_{k=1}^{K} w_k \log_2\left(1+\gamma_k P \dfrac{x_k^2}{\sum_{\ell=1}^{K} x_\ell^2}\right)$ the optimization problem becomes $\max_{x_1,\ldots,x_K} \varphi(x_1, \ldots, x_K)$

Since $P_k = P\frac{x_k^2}{x_1^2+\cdots+x_K^2}$ we have

$$\begin{aligned}
\frac{\partial \varphi}{\partial x_k} &= \sum_{\ell=1}^{K} \frac{w_\ell \gamma_\ell}{1+\gamma_\ell P_\ell} \frac{\partial P_\ell}{\partial x_k} \log_2 \mathrm{e} \\
&= 2P \sum_{\ell=1}^{K} \frac{w_\ell \gamma_\ell}{1+\gamma_\ell P_\ell} \left(\frac{x_k \delta_{k,\ell}}{x_1^2+\cdots+x_K^2} - \frac{x_\ell^2 x_k}{(x_1^2+\cdots+x_K^2)^2} \right) \log_2 \mathrm{e} \\
&= 2P \frac{x_k}{x_1^2+\cdots+x_K^2} \sum_{\ell=1}^{K} \frac{w_\ell \gamma_\ell}{1+\gamma_\ell P_\ell} \left(\delta_{k,\ell} - \frac{x_\ell^2}{x_1^2+\cdots+x_K^2} \right) \log_2 \mathrm{e} \\
&= 2P \frac{x_k}{x_1^2+\cdots+x_K^2} \left\{ \frac{w_k \gamma_k}{1+\gamma_k P_k} - \frac{1}{P} \sum_{\ell=1}^{K} \frac{w_\ell \gamma_\ell P_\ell}{1+\gamma_\ell P_\ell} \right\} \log_2 \mathrm{e}
\end{aligned}$$

Thus, setting $\lambda \triangleq \left\{ \frac{1}{P} \sum_{\ell=1}^{K} \frac{w_\ell \gamma_\ell P_\ell}{1+\gamma_\ell P_\ell} \right\}^{-1}$

we get the following equations: $x_k \left\{ \frac{w_k \gamma_k}{1+\gamma_k P_k} - \frac{1}{\lambda} \right\} = 0$

$k = 1, \ldots, K$

These equations have two possible solutions:

1. $x_k = 0 \Longrightarrow P_k = 0$
2. $P_k = \lambda w_k - \frac{1}{\gamma_k}$

Summarizing we write the solution as

$$P_k = \left(\lambda w_k - \frac{1}{\gamma_k} \right)_+$$

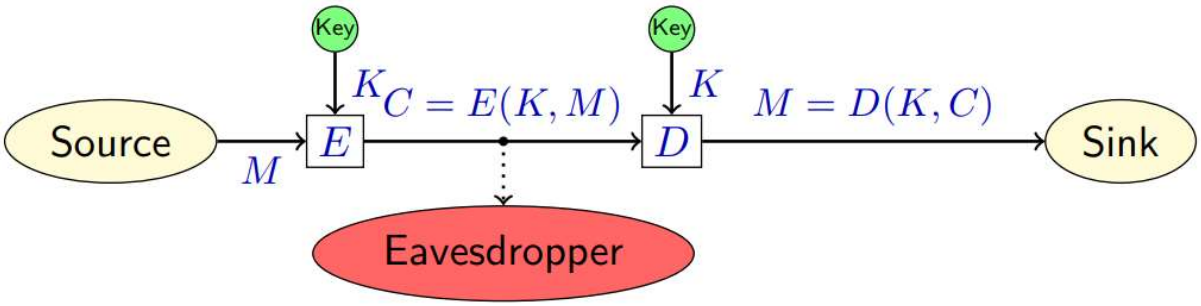
where we use the notation $(x)_+ \triangleq \max(0, x)$.

The remaining problem is determining the parameter λ.

The problem can be solved by finding the solution of the following water-filling equation:

$$\sum_{k=1}^{K} \left(\lambda w_k - \frac{1}{\gamma_k} \right)_+ = P.$$

27. Illustrate in detail the concepts of perfect secrecy and "one-time pad" for a secure communication system.



We model the plaintext message M, the key K, and the encrypted message C as random variables with alphabets $\mathcal{M}, \mathcal{K}$, and $\mathcal{C}$, respectively.

Since $M = D(K, C)$, the conditional entropy $H(M|K, C) = 0$, i.e., if we know the key and the encrypted message, decryption must be always possible.

On the contrary, if we only know C but not K, decryption must be very difficult, i.e., $H(M|C)$ must be as great as possible.

From the general entropy inequality $H(M|C) \leq H(M)$, we realize that the best conditions occur when

$$H(M|C) = H(M) \quad \Leftrightarrow \quad I(M; C) = 0$$

In this case, M and C are statistically independent and the cryptographic system has perfect secrecy.

Since

$$H(M|K,C) = 0 = H(M,K,C) - H(K,C),$$

we have

$$\begin{aligned} H(K) &\geq H(K|C) \\ &= H(K,C) - H(C) \\ &= H(M,K,C) - H(C) \\ &= H(M,K,C) - H(M,C) + H(M,C) - H(C) \\ &= H(K|M,C) + H(M|C) \\ &\geq H(M|C) \\ &= H(M) \qquad \text{(with perfect secrecy)} \end{aligned}$$

The one-time pad is a cryptographic system achieving perfect secrecy but very impractical.

It consists of using a truly random binary key sequence of the same length as the binary plaintext message without reuse.

Trusted messengers are required to carry the key from the source to the destination before its use.

One time pad is more a theoretical approach rather than a practical one since it consists in using only once as a key a truly random binary sequence of the same length as the message .

With the one-time pad, the key is a truly random bit sequence, i.e., K is a bit sequence, $K = (K_1, \ldots, K_N)$, where the K_i's are independent equiprobable binary random variables with

$$P(K_i = 0) = P(K_i = 1) = \frac{1}{2}$$

Equivalently, all the possible 2^N key sequences have the same probability, 2^{-N}.

Let the plaintext message and the encrypted message be represented by the bit sequences $(M_1, \ldots, M_N)$ and $(C_1, \ldots, C_N)$, respectively.

Encryption consists in setting $C_i = M_i \oplus K_i$ for $i = 1, \ldots, N$, where the symbol $\oplus$ denotes modulo-2 addition.

Whatever the distribution of the plaintext message $M = (M_1, \ldots, M_N)$, the encrypted message $C = (C_1, \ldots, C_N)$ is independent of M.

$$P(C = c|M = m) = P(m \oplus K = c) = P(K = m \oplus c) = 2^{-N}$$

since all the keys are equiprobable.

Also,

$$\begin{aligned} P(C = c) &= \sum_m P(M = m)P(C = c|M = m) \\ &= 2^{-N} \sum_m P(M = m) \\ &= 2^{-N} \end{aligned}$$

Therefore,

$$P(C = c|M = m) = P(C = c)$$

for every m, c, so that M and C are independent.

Therefore the one time pad starting from a truly random sequence satisfy the perfect secrecy condition

28. Illustrate the concept of "one-time pad" and the Maurer scheme along with their connection.

Se previous question for the concept of "one-time pad"

The Maurer cryptographic scheme is based on an ideal network of 2^L phones whose numbers are L-bit vectors.

The ℓ-th phone ($0 \leq \ell \leq 2^L - 1$) stores a random N-bit vector R_ℓ.

The plaintext and encrypted messages are also N-bit vectors, denoted by M and C.

The encryption key is an L-bit vector K, which is used to call the K-th phone and obtain the N-bit vector R_K in response, which is used to encrypt the plaintext message:

$$C = M \oplus R_K$$

The designed receiver knows the key K and can recover R_K and the plaintext message from the encrypted message:

$$M = C \oplus R_K$$

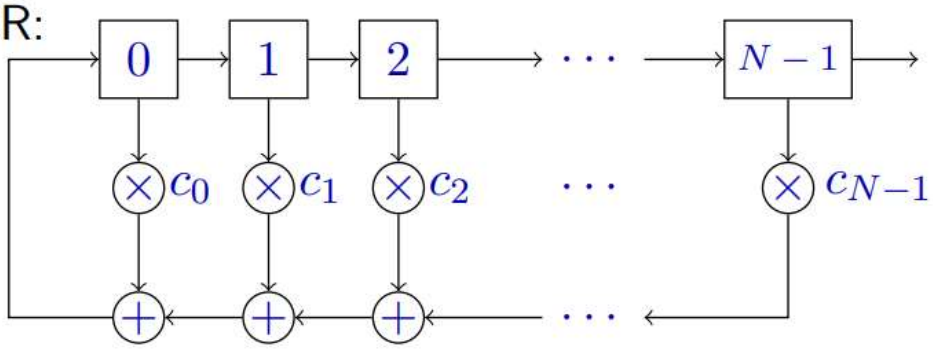
An attacker can attempt to decrypt C only by guessing the phone number.

If she makes 2^T random attempts, she can break the cipher with probability 2^{T-L}, otherwise, with probability $1 - 2^{T-L}$, the cryptographic scheme has perfect secrecy as the one-time pad.

If we set $L = 100$, $2^{50} \approx 10^{15}$ attempts would break the cipher with probability $2^{50-100} \approx 10^{-15}$.

29. Derive the output of an LFSR with N cells and connection coefficients c0, c1, . . . , cN−1 (general case).

General LFSR:



The LFSR equations are summarized by

$$R_0[n+1] = \sum_{i=0}^{N-1} c_i R_i[n]$$

We assume that $R_i[n] = 0$ for any $n < 0$.

We assume that all $R_i[0]$ are known (initial state)

We note that $R_i[n] = R_{N-1}[n+N-i-1]$ (it takes $(N-i-1)$ clock cycles to move the output of cell i to cell $N-1$).

The LFSR equations can be transformed into equations in $R_{N-1}[n]$ as follows:

$$R_{N-1}[n+N] = \sum_{i=0}^{N-1} c_i R_{N-1}[n+N-1-i]$$

Each equation is multiplied by D^{n+N} and added for $n = 0$ to ∞:

$$\sum_{n=0}^{\infty} D^{n+N} R_{N-1}[n+N] = \sum_{i=0}^{N-1} c_i \sum_{n=0}^{\infty} D^{n+N} R_{N-1}[n+N-1-i]$$

The previous equations can be rewritten in the following form:

$$\sum_{n=N}^{\infty} R_{N-1}[n] D^n = \sum_{i=0}^{N-1} c_i D^{i+1} \sum_{n=N-1-i}^{\infty} R_{N-1}[n] D^n$$

Defining $R_i(D) \triangleq \sum_{n=0}^{\infty} R_i[n]D^n$ we can write the LFSR equations in the following finite form:

$$\left(1 - \sum_{i=0}^{N-1} c_i D^{i+1}\right) R_{N-1}(D)$$

$$= \sum_{n=0}^{N-1} R_{N-1}[n]D^n - \sum_{i=0}^{N-1} c_i \sum_{n=0}^{N-2-i} R_{N-1}[n]D^{n+i+1}$$

$$= \sum_{n=0}^{N-1} R_{N-1-n}[0]D^n - \sum_{i=0}^{N-1} c_i \sum_{n=0}^{N-2-i} R_{N-1-n}[0]D^{n+i+1}$$

Therefore, $R_{N-1}(D) =$

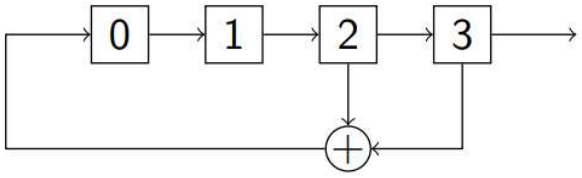
$$\frac{\sum_{n=0}^{N-1} R_{N-1-n}[0]D^n - \sum_{i=0}^{N-1} c_i \sum_{n=0}^{N-2-i} R_{N-1-n}[0]D^{n+i+1}}{1 - \sum_{i=0}^{N-1} c_i D^{i+1}}$$

the maximum period of a LFSR is $2^N - 1$ where N is the number of registers.

30. Derive the period and the output of an LFSR with N cells and given connection coefficients c0, c1, . . . , cN−1 (specific case with numerical data).

the maximum period of a LFSR is $2^N - 1$ where N is the number of registers.

Assuming $R_0[0] = 1$ and $R_i[0] = 0$ for $i = 1, 2, 3$ in our example,

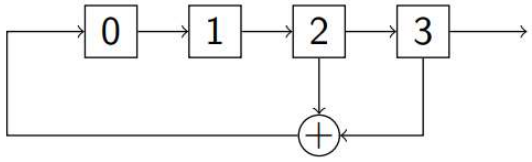


we determine the state sequence:

n	state	n	state
0	1000	8	1010
1	0100	9	1101
2	0010	10	1110
3	1001	11	1111
4	1100	12	0111
5	0110	13	0011
6	1011	14	0001
7	0101	15	1000

31. Illustrate in detail if and how an LFSR can be used as a stream cipher.

One possible way to implement a stream cipher is using a LFSR such as the following:



The boxes are registers with content $R_i[n]$ at time n, $i = 0, \ldots, 3$.

At every clock time the contents propagate through the paths indicated by the arrows, so that we have the equations

$$\begin{cases} R_0[n+1] &= R_2[n] + R_3[n] \\ R_1[n+1] &= R_0[n] \\ R_2[n+1] &= R_1[n] \\ R_3[n+1] &= R_2[n] \end{cases} \qquad n = 0, 1, 2, \ldots$$

The previous equations can be written as

$$\begin{cases} \sum_{n=0}^{\infty} D^{n+1} R_0[n+1] &= \sum_{n=0}^{\infty} D^{n+1} \{R_2[n] + R_3[n]\} \\ \sum_{n=0}^{\infty} D^{n+1} R_1[n+1] &= \sum_{n=0}^{\infty} D^{n+1} R_0[n] \\ \sum_{n=0}^{\infty} D^{n+1} R_2[n+1] &= \sum_{n=0}^{\infty} D^{n+1} R_1[n] \\ \sum_{n=0}^{\infty} D^{n+1} R_3[n+1] &= \sum_{n=0}^{\infty} D^{n+1} R_2[n] \end{cases}$$

Then, defining $R_i(D) \triangleq \sum_{n=0}^{\infty} R_i[n] D^n$, we get

$$\begin{cases} R_0(D) - R_0[0] &= D[R_2(D) + R_3(D)] \\ R_1(D) - R_1[0] &= D R_0(D) \\ R_2(D) - R_2[0] &= D R_1(D) \\ R_3(D) - R_3[0] &= D R_2(D) \end{cases}$$

The equations can be rewritten as

$$\begin{cases} D^3R_0(D) & = D^3R_0[0] + D^4[R_2(D) + R_3(D)] \\ R_1(D) & = R_1[0] + DR_0(D) \\ R_2(D) & = R_2[0] + DR_1[0] + D^2R_0(D) \\ R_3(D) & = R_3[0] + DR_2[0] + D^2R_1[0] + D^3R_0(D) \end{cases}$$

Then, after eliminating $R_0(D)$, $R_1(D)$, $R_2(D)$, we have:

$$\begin{aligned} D^3R_0(D) = R_3(D) - R_3[0] - DR_2[0] - D^2R_1[0] \\ + D^3R_0[0] + D^3\{R_3(D) - R_3[0]\} + D^4R_3(D) \end{aligned}$$

Solving for $R_0(D)$, we obtain

$$R_3(D) = \frac{R_3[0] + DR_2[0] + D^2R_1[0] + D^3(R_0[0] - R_3[0])}{1 - D^3 - D^4}$$

Typically, the LFSR registers are binary, so that modulo-2 arithmetic rules apply and we get

$$R_3(D) = \frac{R_3[0] + DR_2[0] + D^2R_1[0] + D^3(R_0[0] + R_3[0])}{1 + D^3 + D^4}$$

As we can see, the state sequence is periodic .

This is a general property of the LFSR's: the number of possible states is finite and must repeat itself as a sequence.

The all-zero state repeats itself immediately so that it is excluded.

Therefore, the maximum period of a LFSR is $2^N - 1$ where N is the number of registers.

The period reaches the maximum if the connection polynomial is primitive.

AN LFSR is not usable as a stream cipher because it is vulnerable to a known plaintext attack of length $2N$.

A known plaintext attack consists of knowing a sequence of plaintext and encrypted message bits, which is equivalent to knowing $2N$ stream cipher bits.

32. Illustrate the A5/1 algorithm and its properties.

The A5/1 bit stream is the XOR of three LFSR outputs with irregular clocking.

Each LFSR is clocked if its clocking bit (orange) agrees with the majority of the three clocking bits.

For example, if the three clocking bits are $1, 1, 0$, the majority bit is 1, so that the first two LFSR's clock while the third doesn't.

The initialization of the LFSR states is done by feeding their LSB's with the private key followed by the public session number.

Next, the LFSR's output the 114 bit stream cipher for one GSM block.

33. Illustrate the concept of unicity distance and apply it to the following encryption scheme (to be specified).

The concept arises from the fact that the longer is the encrypted message known to an attacker, the higher is the chance that the encryption key can be recovered.

Let us denote by M_L and C_L the plaintext and encrypted message of length L obtained by an encryption scheme with fixed key K.

The key uncertainty is $H(K|C_L)$. In general, the key uncertainty decreases with the message length L.

Since $C = E(K, M)$, C_L is a deterministic function of M_L given the key K. Thus,

$$H(C_L|K) = H(M_L|K) \quad \Rightarrow \quad H(K, C_L) = H(K, M_L)$$

Now, assume that the plaintext and encrypted message share the same alphabet $\mathcal{X}$.

We have

$$\begin{aligned} H(K|C_L) &= H(K, C_L) - H(C_L) \\ &= H(K, M_L) - H(C_L) \\ &= H(K) + H(M_L) - H(C_L) \\ &\geq H(K) + H(M_L) - L \log_2 |\mathcal{X}| \end{aligned}$$

since the entropy of the encrypted message cannot exceed the case of iid equiprobable symbols over $\mathcal{X}$.

On the other hand, the message is typically from a spoken language and its symbols are not iid equiprobable.

Rather, they have a limiting entropy rate $\bar{H} < \log_2 |\mathcal{X}|$.

Hence, the previous inequality becomes

$$H(K|C_L) \geq H(K) - LD_X$$

where $D_X \triangleq \log_2 |\mathcal{X}| - \bar{H}$ is called the redundancy of the source X.

Thus, in order to have the possibility of recovering the encryption key K, the lower bound must be negative or 0, which occurs when

$$L \geq L_U \triangleq \frac{H(K)}{D_X}$$

Then, L_U is called the unicity distance

As an example of application of this concept, consider the encryption scheme called substitution cipher.

The scheme corresponds to a permutation of the English alphabet, such as

$$A \to B, B \to G, C \to J, D \to B, \ldots, Z \to A$$

Since the English alphabet has 26 letters, the number of possible permutations is $26! = 4.03 \cdot 10^{26}$.

The key is the permutation and is selected randomly and equiprobably among $26!$ possibilities, so that the entropy of the key is

$$H(K) = \log_2(26!) = 88.38$$

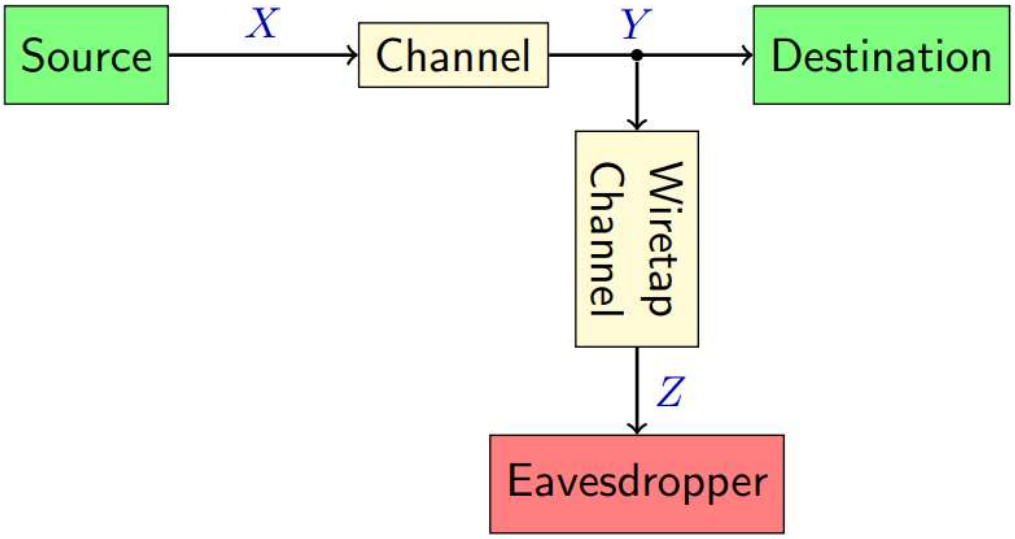
The English language has an entropy rate approximately equal to 2.62 bit/character so that the redundancy is $\log_2 26 - 2.62 = 2.08$.

As a result, the unique distance is

$$L_U = \frac{88.38}{2.08} = 42.48$$

34. Illustrate the wiretap channel and derive the secrecy capacity of a binary symmetric wiretap channel.

The block diagram of this channel is represented as follows:



The situation depicted represents that of an unauthorized eavesdropper wiretapping the transmission channel at the receiver and obtaining a degraded (more noisy) version of the received signal.
By the data-processing inequality we know that:

$$I(X;Z) \leq I(X;Y)$$

Therefore, the authorized receiver (signal Y) can achieve a better rate than the eavesdropper (signal Z). The transmitter can use a randomized encoder (known only to the authorized receiver) in order to set the leakage rate to the eavesdropper equal to zero.

The resulting achievable rate to the authorized receiver, under the above condition, is called secrecy capacity and can be evaluated by

$$C_S = \max_{p_X(x)} [I(X;Y) - I(X;Z)]$$

Example. Let the channels be BSC with error probabilities p_1 and p_E and calculate the secrecy capacity.

The cascade of two discrete channels has a probability matrix that is the product of the two channels' probability matrices:

$$\boldsymbol{P}_{1+2} = \boldsymbol{P}_1 \boldsymbol{P}_2$$

In the case of the cascade of two BSC's with error probabilities p_1 and p_E we obtain another BSC with error probability

$$p_2 = p_1(1-p_E) + (1-p_1)p_E = p_1 + p_E - 2p_1 p_E$$

We can see that if $0 \leq p_1, p_E \leq \frac{1}{2}$, then $0 \leq p_1 \leq p_2 \leq \frac{1}{2}$.

Then, we assume that the transmitted signal X has distribution given by $P(X=0) = \alpha$ and $P(X=1) = \bar{\alpha} \triangleq 1-\alpha$.

The mutual informations of the two channels are

$$I(X;Y) = H_b(\alpha\bar{p}_1 + \bar{\alpha}p_1) - H_b(p_1)$$
$$I(X;Z) = H_b(\alpha\bar{p}_2 + \bar{\alpha}p_2) - H_b(p_2)$$

Then,

$$C_S = \max_{0\leq\alpha\leq 1} \{H_b(\alpha\bar{p}_1 + \bar{\alpha}p_1) - H_b(\alpha\bar{p}_2 + \bar{\alpha}p_2)\} - H_b(p_1) + H_b(p_2)$$

Since $0 \leq p_1 \leq p_2 \leq \frac{1}{2}$, we can see that, if $0 \leq \alpha \leq \frac{1}{2}$,

$$0 \leq \alpha\bar{p}_1 + \bar{\alpha}p_1 \leq \alpha\bar{p}_2 + \bar{\alpha}p_2 \leq \frac{1}{2}$$

and, if $\frac{1}{2} \leq \alpha \leq 1$,

$$1 \geq \alpha\bar{p}_1 + \bar{\alpha}p_1 \geq \alpha\bar{p}_2 + \bar{\alpha}p_2 \geq \frac{1}{2}$$

because

$$\begin{aligned}(\alpha\bar{p}_1 + \bar{\alpha}p_1) - (\alpha\bar{p}_2 + \bar{\alpha}p_2) &= \alpha(1-2p_1) + p_1 - [\alpha(1-2p_2) + p_2] \\ &= (p_1 - p_2)(1-2\alpha)\end{aligned}$$

As a result, for any $0 \leq \alpha \leq 1$, we have

$$H_b(\alpha\bar{p}_1 + \bar{\alpha}p_1) \leq H_b(\alpha\bar{p}_2 + \bar{\alpha}p_2)$$

with equality only if $\alpha = \frac{1}{2}$.

Therefore, the secrecy capacity is given by

$$C_S = H_b(p_2) - H_b(p_1)$$

Part 2 - Prof. Garello

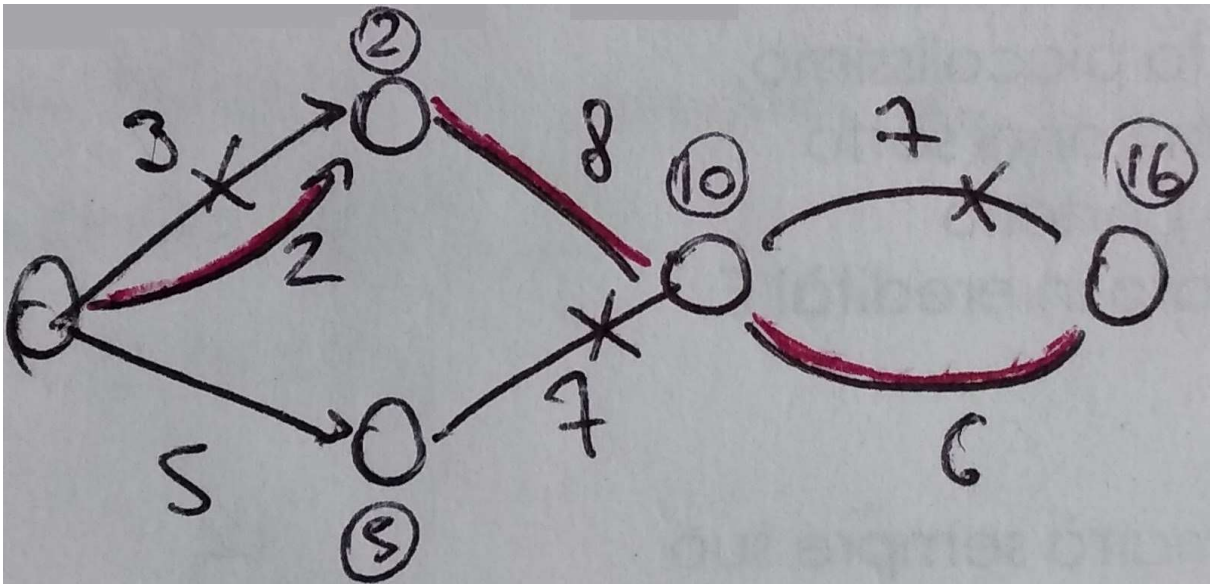
1. Present the problem of finding the minimum cost trajectory of a discrete system. Discuss the brute-force exhaustive approach. Present the Bellman theorem.

A discrete system is a system with countable number of states for any instant of time. The admissible behaviour of the system evolution identify for each state the edges connecting it to any other state.
It is called trajectory any sequence of edges from i=0 to i=L.
It is possible to define the trajectory cost once it is assigned a weight to every edge as the sum of the weights of the edges belonging to the trajectory.
We are then interested in finding the minimum cost trajectory from an initial instant of time state to a final instant state.
One possible solution can be implemented via a brute-force exhaustive approach that implies the search of the complete solution set, that is the computation of the total weight for every possible trajectory, and the selection of the minimum cost one. However this method is computationally too expensive most of the time.
To optimize the computations it is possible to implement the Viterbi algorithm which is based on the Bellman theorem that asserts: “Assuming the minimum cost trajectory between instant 0 and L is known, then any sub-trajectory of that trajectory from 0 to i (with 0<i<L) is the minimum cost trajectory between 0 and i”.

2. Present the Bellman theorem. Present the Viterbi algorithm for a generic discrete system.

To optimize the computations of the brute-force approach it is possible to implement the Viterbi algorithm which is based on the Bellman theorem that asserts: “Assuming the minimum cost trajectory between instant 0 and L is known, then any sub-trajectory of that trajectory from 0 to i (with 0<i<L) is the minimum cost trajectory between 0 and i”.

In the Viterbi algorithm the weight of every state at any given time is stored as the sum of the edges of the minimum cost trajectory reaching the state, in case two or more edges enter the same state only the one belonging to the minimum cost trajectory among them is stored, this way at the end of the sequence the optimal solution of the minimum cost trajectory is reached.



3. Present the application of the Viterbi algorithm to convolutional codes decoding.

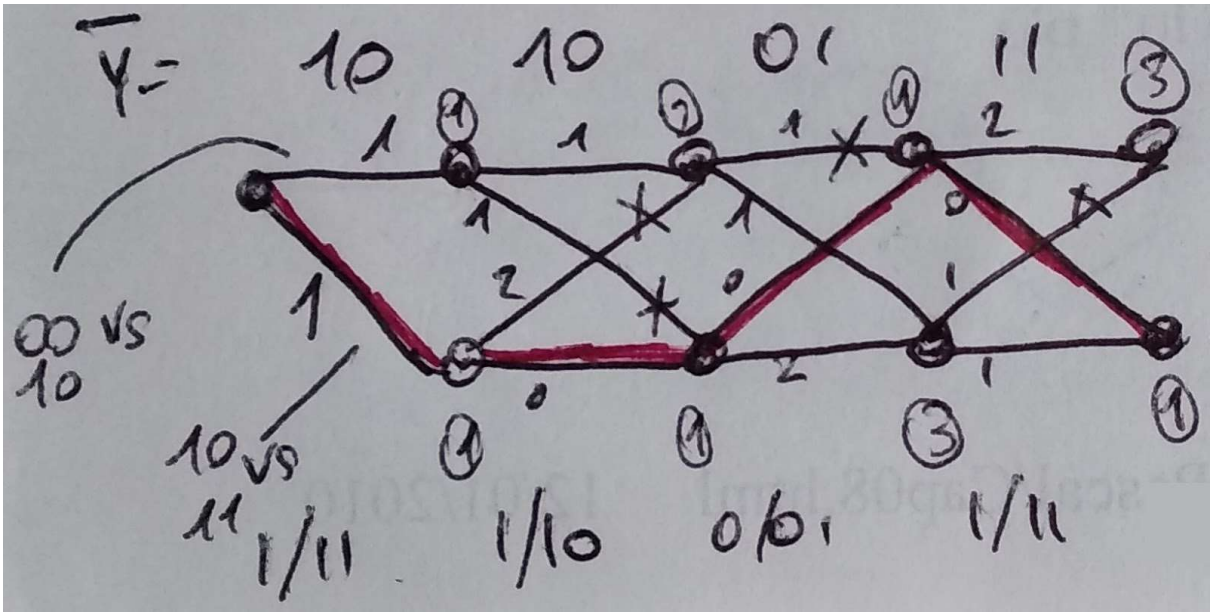
Viterbi algorithm is especially useful when it comes to convolutional codes decoding. A convolutional code is a type of error-correcting code that generates parity symbols.

The decoding phase is characterized by a state diagram called “Trellis” where every edge is associated with input bit/output bits.

In this type of system the number of trajectories is exponential with the length L of the original transmitted sequence, this is why most of the time a brute-force approach is not feasible.

In the Viterbi algorithm the weight of every state at any given time is stored as the sum of the edges of the minimum cost trajectory reaching the state, in case two or more edges enter the same state only the one belonging to the minimum cost trajectory among them is stored, this way at the end of the sequence the optimal solution of the minimum cost trajectory is reached.

The cost of the edges is assigned by mean of the computation of the Hamming distance between received bits and edge output bits.



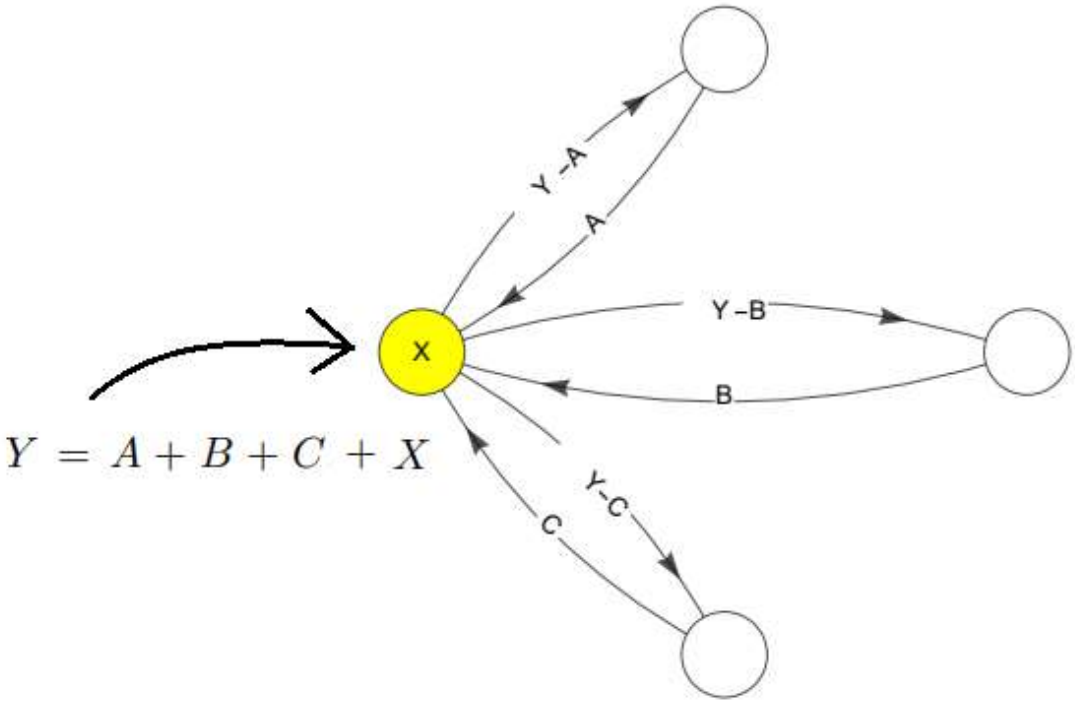
4. Present the early decision Viterbi algorithm. Discuss the advantages and the disadvantages.

In the standard implementation of the Viterbi algorithm, the optimal solution (that is the best trajectory), is only reached at the end of the computations with a latency equivalent to the length L of the sequence.
In the early decision Viterbi implementation the weight of each state is store as well as its trajectory, at each time 'i' for every state the new weight value is calculated and the bit at the time 'i - D' (with D chosen integer value) belonging to the trajectory of the state with smallest weight is released.
The early decision Viterbi implementation is therefore a trade-off between speed and accuracy, in this way the disadvantage with respect to the standard implementation is that the final solution is sub-optimal since the best trajectory at a given time might not be the final best solution.
The advantage on the other hand is that the latency is massively reduced from the sequence length L to a way smaller value D, furthermore it can be observed that with a sufficiently large value of D the error is negligible.

5. Discuss the ideas of distributed computing, message propagation and extrinsic information.

The idea of distributed computing is that a big problem can be divided into small tasks easier to solve. This can be realized by using nodes with limited computation power, each performing some observation and some simple computations and sharing the result with its neighbors by sending messages. Belief propagation is a message passing algorithm to perform distributed computing, in particular inference on graphs.
The key concept behind the algorithm is the extrinsic information, each node computes a value considering all the incoming messages, adds its own observed value and sends back to each neighbour a message containing the extrinsic information, which is equal to the total sum minus the incoming message from that particular node. It is important to note that, in this way, the information sent back to a node N is given by the information coming from all other sources but the node N itself.

6. Present an example of distributed computation of the number of nodes in a network and discuss the idea of extrinsic information.



The key concept behind the algorithm is the extrinsic information, each node computes a value considering all the incoming messages, adds its own observed value and sends back to each neighbour a message containing the extrinsic information, which is equal to the total sum minus the incoming message from that particular node. It is important to note that, in this way, the information sent back to a node N is given by the information coming from all other sources but the node N itself.

7. Provide the definition of factor graph and present an example.

The distributed algorithms based on message passing and extrinsic information idea allow to compute complex functions by working on a distributed network.

$$f : \mathbb{R}^n \to \mathbb{R}$$
$$\mathbf{x} \mapsto f(\mathbf{x})$$
$$\text{where } \mathbf{x} = (x_1, \ldots, x_j, \ldots, x_N)$$

$$f(\mathbf{x}) = \prod_{i=1}^{K} f_i(\mathbf{x}_i) \qquad \mathbf{x}_i \subseteq \mathbf{x}$$

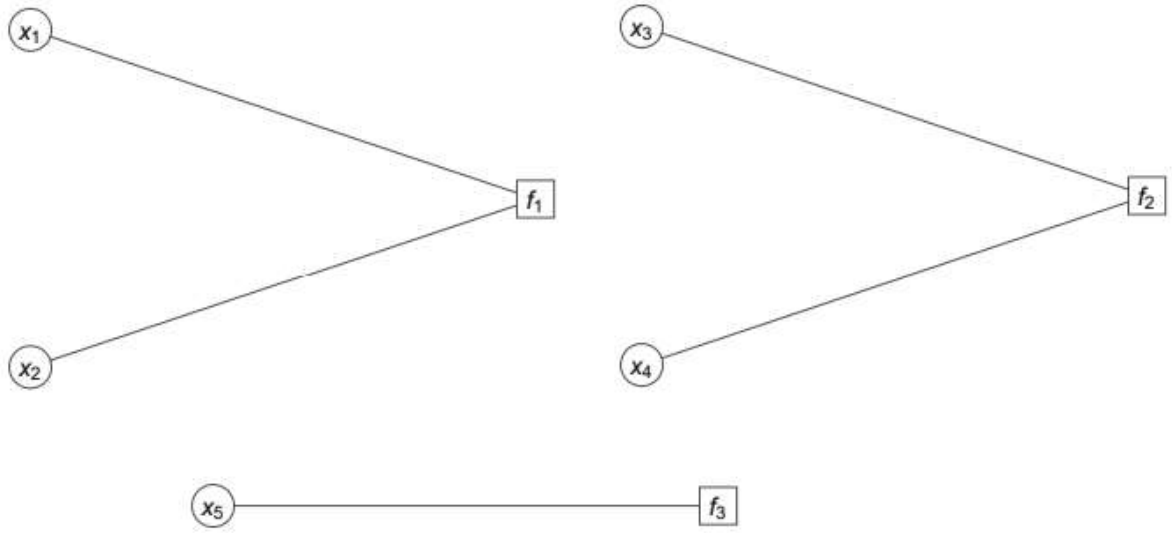
suppose that f can be expressed as the product of some elementary function, called factors, each of them involving only a small subset of the entire x, instead of computing f working on the entire vector x we can compute the elementary factors fi , where each involves only a small subset of variables xi , and then combine their results.

In this context it is useful to introduce the concept of a factor graph, a bipartite graph made by two kind of nodes (variables, denoted by circles, and factors, denoted by squares), in order to graphically display our function f.

A variable node is connected to a factor node by an edge if and only if it is one of its argument.

An example:

$$f(x_1, x_2, x_3, x_4, x_5) = f_1(x_1, x_2)\, f_2(x_3, x_4)\, f_3(x_5)$$



8. Present the marginalization problem and discuss the problems of the brute-force approach.

Given a function f representing a joint probability mass function, the marginalization problem consists in computing the marginal distribution for some variable xi.

Computing the marginal distribution consists in calculating the following formula for every possible value alpha that the given random variable can take.

$$\mathbb{P}[x_i = \alpha] = \sum_{\mathbf{x}:x_i=\alpha} f(\mathbf{x}) \qquad \forall \alpha \in \mathcal{A}$$

The problem is that the number of vectors to be considered in the sum can be huge therefore a brute-force approach is not feasible. The key idea behind the application of message passing to the marginalization problem is that if f(x) can be expressed as the product of factors, each of them involving a small subset of x, the marginalization computation becomes much simpler because each factor involves only a small set of vectors.

9. For a binary block code, provide the definitions of generator matrix, codebook, parity check matrix, indicator function.

A generator matrix is a matrix whose rows form a basis for a linear code. The codewords are all of the linear combinations of the rows of this matrix.

We consider a k-bit information vector $\mathbf{v} \in \mathcal{A}^k$. We map it into an n-bit coded vector $\mathbf{c} \in \mathcal{A}^n$ (also called codeword), given by the linear function

$$f : \mathcal{A}^k \to \mathcal{A}^N$$
$$\mathbf{v} \mapsto \mathbf{c}$$

where

$$\mathbf{c} = \mathbf{v}\,\mathbf{G}$$

The binary matrix $\mathbf{G} \in \mathcal{A}^{k \times n}$ is the **generator matrix** defined as

$$\mathbf{G} = \left[\begin{array}{c|c} \mathbf{I}_k & \mathbf{R} \end{array}\right]$$

where $\mathbf{I}_k$ is the identity matrix of order k and $\mathbf{R}$ is the redundancy matrix whose dimension is $k \times (n-k)$.

We call codebook the set of all possible codewords and we denote it by C. The cardinality of the set of all information vectors is 2k and then the cardinality of the codebook C is 2k too because f is a one to one map.

We define the **parity check matrix** $\mathbf{H} \in \mathcal{A}^{n \times n-k}$ as $\mathbf{H} = \left[\begin{array}{c} \mathbf{R} \\ \hline \mathbf{I}_{n-k} \end{array}\right]$

We note that $\mathbf{G}\,\mathbf{H} = \left[\begin{array}{c|c} \mathbf{I}_k & \mathbf{R} \end{array}\right] \left[\begin{array}{c} \mathbf{R} \\ \hline \mathbf{I}_{n-k} \end{array}\right] = \mathbf{I}_k\,\mathbf{R} + \mathbf{R}\,\mathbf{I}_{n-k} = \mathbf{R} + \mathbf{R} = \mathbf{0}$

due to the property of binary sum.

Multiplying both members of (17) by an information vector $\mathbf{v}$ leads to

$$\mathbf{v}\,\mathbf{G}\,\mathbf{H} = \mathbf{c}\,\mathbf{H} = \mathbf{0}$$

The equation $\mathbf{c}\,\mathbf{H} = \mathbf{0}$ is the fundamental property of the parity check matrix $\mathbf{H}$. We note here that $\mathbf{c} \in \mathcal{A}^n$ and $\mathbf{H} \in \mathcal{A}^{n \times n-k}$, then $\mathbf{0} \in \mathcal{A}^{n-k}$. By using $\mathbf{H}$ definition, it is easy to show that for any vector $\mathbf{y} \in \mathcal{A}^n$ which is not a codeword we have $\mathbf{y}\,\mathbf{H} \neq \mathbf{0}$, then we can use $\mathbf{H}$ to distinguish between codewords and non-codewords.

If we consider the equation $\mathbf{c}\,\mathbf{H} = \mathbf{0}$ componentwise, we obtain $n-k$ conditions upon $\mathbf{c}$ being a codeword, i.e. a way to factorize the indicator function μ. We show this with an example.

A key idea for factor graph representation is the **indicator function**

$$\mu : \mathcal{A}^N \to \{0, 1\}$$

that maps every codeword to 1 and every non codeword to 0

$$\mu(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in \mathcal{C} \\ 0 & \text{if } \mathbf{x} \notin \mathcal{C} \end{cases}$$

10. For a binary block code, consider an example and show how it is possible to factorize the indicator function by using the parity check matrix H.

CONSIDERING A BINARY BLOCK CODE WITH REDUNDANCY MATRIX $R = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$ WE THEN HAVE THE GENERATOR MATRIX $G = [I|R] = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$ AND PARITY CHECK MATRIX $H = \begin{bmatrix} R \\ I \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$

PARITY CHECK MATRIX IS DEFINED BY CONSTRUCTION WITH A PROPERTY: $G \cdot H = [I|R]\begin{bmatrix} R \\ I \end{bmatrix} = R + R = 0 \Rightarrow \bar{c} = \bar{\bar{v}}\bar{\bar{G}} \rightarrow$

$\rightarrow \bar{c}\bar{\bar{H}} = \bar{v}\bar{\bar{G}}\bar{\bar{H}} = 0$

bitwise
0+0=0
1+1=0

with c: codeword and v: vector

EVERY CODEWORD BELONGING TO THE CODEBOOK MULTIPLIED BY THE PARITY CHECK MATRIX RETURNS ZERO.
CONSIDERING A 4 BITS GENERIC CODEWORD
$\bar{x} = (x_1, x_2, x_3, x_4)$ WE DEFINE THE INDICATOR FUNCTION

$$\mu(\bar{x}) = \begin{cases} 1 & \bar{x} \in G \\ 0 & \bar{x} \notin G \end{cases}$$

with G codebook

WE CAN NOW SEE THE CONDITION A VECTOR HAS TO SATISFY IN ORDER TO BE A VALID CODEWORD

$$\bar{x}\bar{\bar{H}} = (x_1, x_2\ x_3\ x_4)\begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} x_1 + x_2 + x_3 \\ x_1 + x_4 \end{pmatrix}^T = \begin{pmatrix} 0 \\ 0 \end{pmatrix}^T$$

THE INDICATOR FUNCTION CAN BE EXPRESSED AS THE PRODUCT OF ELEMENTARY FACTORS USING THE IVERSON BRACKET

$$[P] = \begin{cases} 1 & \text{if } P \text{ IS TRUE} \\ 0 & \text{IF } P \text{ IS FALSE} \end{cases}$$

WE CAN EXPRESS THE FACTORIZATION AS

$$\mu(\bar{x}) = [x_1 + x_2 + x_3 = 0] \cdot [x_1 + x_4 = 0]$$

11. Given a factor graph, present the variable node update rule.

Given the variable node x_j, we denote with $E[x_j]$ the set of all factor nodes connected to x_j. The node x_j receives messages from all its neighbour factors. These messages represent the probability of the different values of x_j computed by that factor graph: $M_{f_i \to x_j}(x_j = \alpha_j)$ is the probability that $x_j = \alpha_j$ computed by the factor node f_i.

The variable node x_j computes the product of all incoming messages

$$\mathbb{P}[x_j = \alpha_j] = \prod_{f_i \in E[x_j]} M_{f_i \to x_j}(x_j = \alpha_j) \qquad \forall \alpha_j \in \mathcal{A}_j$$

and sends back to the node f_{i^*} the extrinsic information

$$M_{x_j \to f_{i^*}}(x_j = \alpha_j) = \prod_{\substack{f_i \in E[x_j] \\ f_i \neq f_{i^*}}} M_{f_i \to x_j}(x_j = \alpha_j) \qquad \forall \alpha_j \in \mathcal{A}_j$$

12. Discuss the ideas of statistical classification and statistical classification by decision trees.

Statistical classification is the broad supervised learning approach that trains a program to categorize new, unlabeled information based upon its relevance to known, labeled data.

The algorithms that sort unlabeled data into labeled classes, or categories of information, are called classifiers. A simple practical example are spam filters that scan incoming “raw” emails and classify them as either “spam” or “not-spam.” Classifiers are the concrete implementation of pattern recognition in any form of machine learning.

The Decision Tree is a classification algorithm based on a flowchart-like diagram that, given an object, computes an outcome from a series of decisions taken by analyzing the object’s features.

13. Present the definitions of instance, feature, class, training set and classifier.

An instance is a single object of the world from which a model will be learned, or on which a model will be used. In most machine learning work, instances are described by feature vectors where the features are quantities describing the instance.
Features have a domain defined by the feature type, which denotes the values that can be taken by that feature. Domain type can be categorical (nominal or ordinal) or continuous (subset of real numbers, where there is a measurable difference between the possible values).
Supervised learning is the machine learning task of acquiring a mathematical function from labeled training data consisting in a set of pairs of input object (typically feature vectors) and a desired output value which is called class.
That mathematical function is called classifier and it is then implemented by a classification algorithm in order to map unlabelled instances to their belonging class.

14. Present and discuss the definition of Information Gain and Information Gain Ratio.

Given a random variable X with discrete alphabet $\mathcal{X}$ and probability distribution $p_X(x)$ defined for all $x \in \mathcal{X}$, we define its entropy as:

$$H(X) \triangleq -\sum_{x \in \mathcal{X}} p_X(x) \log_2 p_X(x) \text{ bits..}$$

Entropy inequalities:

$$0 \leq H(\boldsymbol{p}) \leq \log_2 N$$

The conditional entropy of two random variables X and Y is defined as

$$H(X|Y) \triangleq -\sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} p_{XY}(x,y) \log_2 p_{X|Y}(x|y).$$

We always have: $H(A) \geq H(A|B)$

Entropy is a measure of uncertainty (information) for A.
When we know B we have some information on A, too, then its uncertainty is reduced.

$I(A;B) = H(A) - H(A|B)$

$I(A;B) =$ reduction on A uncertainty provided by B.

In classifier literature, the **mutual information** is often called the information gain and is denoted by $IG(B \longrightarrow A)$.

If we would want to compare the information gain on A given by different random variables the size may be influential on giving information on A making different random variables of different sizes not comparable therefore we use the information gain ratio defined as:

It is defined as the ratio between the mutual information (information gain) and the entropy of the conditioning variable :

$$IGR(B \longrightarrow A) = \frac{I(A;B)}{H(B)}$$

The reason to divide by the entropy $H(B)$ is the following. Suppose that A depends on two variables B_1 and B_2. If the size of B_1 is bigger than that of B_2 we can expect a bigger reduction on A uncertainty. By dividing by the entropy of B (that increases with the cardinality), we normalize the information gain and the comparison between variables B with different size is fairer.

IGR is used in the decision tree algorithm:

At each step of decision tree algorithm **use the feature that provides the largest reduction of the class uncertainty (i.e., the largest Information Gain Ratio).**

15. Describe the ID3 classifier algorithm.

Iterative Dichotomiser 3 algorithm (by **Ross Quinlan**).
Given a training set, it generates a **Decision Tree Classifier** based on this key idea:

At each step use the feature that provides the largest reduction of the class uncertainty (i.e., the largest Information Gain Ratio).

The algorithm generates a Decision Tree in this way:

- Given a set S of vectors $\underline{v} = (x_1, \cdots, x_i, \cdots, x_M)$ and their classes $c = f(\underline{v})$, compute the feature X_i that maximizes the Information Gain Ratio $IGR(C, X_i)$.
- Create a decision node for this feature X_i and split the set S into subsets according to it.
- Recurse on subsets by using the remaining features.

Stopping criterion:

- When a subset contains only vectors of the same class: we create a leaf node labeled by this class.
- When a subset contains vectors of different classes, but all the features have already been considered: we create a leaf node labeled by the most common class among the subset vectors.
- When a subset is empty: we create a leaf node labeled by the most common class of the parent's subset.

16. Describe how the C4.5 classifier algorithm deals with numerical features.

C4.5 algorithm

If X_i is a numerical variable, proceed as follows:

- Fix a threshold t.
- Split the X_i values in two sets for $X_i \leq t$ and $X_i > t$.
- Compute the information gain ratio with respect to this binary representation of X_i.
- Consider all possible threshold values t.
- Consider the threshold giving the maximum information gain ratio and use this binary partition for X_i as in the ID3 algorithm.

Each categorical feature can be considered only once, as in ID3.

Important: Numerical features can be considered again: if we select a numerical feature at a given level, we can select it again at a lower level, with a different threshold.

17. Discuss the advantages and the problem of tree classifiers.

Decision Tree = decision-making tool based on a **flowchart**-like diagram that, given an object, computes an outcome from **a series of decisions** taken by analyzing the object's features.

Each internal node represents a test on a **single feature**.

Each branch represents the test outcome.

Each leaf represents the class label.

The path from root to leaf represents the classification function g that, given a vector $\underline{v}$, analyzes its features and associates a class c.

PROS

- ➔ **Can handle both** numerical and categorical data.
- ➔ **Easy to understand and interpret**. At each node, we are able to see exactly what decision our model is making. In practice we'll be able to fully understand where our accuracies and errors are coming from, what type of data the model would do well with, and how the output is influenced by the values of the features.

CONS

- ➔ are **unstable**, meaning that a small change in the data can lead to a large change in the structure of the optimal decision tree.
- ➔ can create over-complex trees that do not generalize the data well. This is called **overfitting**.
- ➔ are vulnerable to becoming **biased** to the classes that have a majority in the dataset. It's always a good idea to do some kind of class balancing such as class weights or sampling.
- ➔ **Greedy** algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees, where the features and samples are randomly sampled with replacement.
- ➔ Decision rules yield orthogonal hyperplanes in the n-dimensional space, thus each region has the form of an hyper-rectangle. But in reality many of these decision hyperplanes are not perpendicular to the coordinates (because certain deciding factors are the results of combinations of different attributes), leading to **approximation errors**.

18. Discuss the Decision Tree Ensemble approach.

To improve the performance of tree classifiers: Tree Ensembles
Main idea: given the training data-set, instead of building a single classification tree we build many and we combine their decisions.

Bagging: starting from the training data-set T_S we build N bootstrapped data-sets made by randomly extracted vectors of T_S.

Random feature selection: For each bootstrapped data-set, we build a classification tree. When doing this, each time we process a subset to build a node, we only use $M' < M$ randomly extracted features.

Decision: given a new vector, we process it with all the N tree classifiers and we assign the most popular class among the N results.

(To decide the number M' of features, we usually start from a value $M' \simeq \sqrt{M}$. Then we test the accuracy of the forest by using the out-of-bag vectors (the vectors which are outside the different bootstrapped data-sets). We repeat the procedure by increasing and decreasing the values of M' looking for the value with best accuracy.)