Course «Information theory». Very brief lecture notes.

10.09.2019. Lecture 1.

1. The game guess a number : one players chooses an integer number between 1 and n, another player should find this number by asking questions with answers *yes* or *no*. There is a simple strategy that allows to find the chosen number in $\lceil \log n \rceil$ questions (bisection). Moreover, there is a non-adaptive strategy with the same number of questions (the second player asks bits of the binary expansion of the chosen number).

These strategies are optimal : no strategy helps to reveal the chosen number in less than $\lceil \log n \rceil$ questions (in the worst case). The lower bound can be explained in terms of an adversarial argument or in terms of the "missing information" (the binary tree representing a valid strategy must contain at least n leaves ; therefore, the height of this tree must be at least log of the number of possible answers).

2. Sorting algorithms. We are given n objects ("stones") and balance scales ; in one operation we can compare weights of two stones. Claim 1 : to find the heaviest stone, we must do $(n-1)$ weighings. Claim 2 : There is a strategy to find the *heaviest* and the *second heaviest* stones with $n + \lceil \log n \rceil - 2$ weighings. Claim 3 : to sort n stones by their weights, we have to do in the worst case $\log(n!)$ pairwise comparisons (no algorithm can guarantee the right answer with less than $\log(n!)$ weighings).

Exercise 1.1 (Sorting algorithms)**.**
(a) [optional] Prove that an array of n elements can be sorted with $O(n \log n)$ comparisons (in the worst case). *Reminder : In the class we proved that no algorithm can do this faster than in* $\log(n!) = \Omega(n \log n)$ *comparisons.*

Find the number of comparisons needed in the worst case
(b) to sort an array of size 4 ;
(c) to sort an array of size 5.

Exercise 1.2. We are given a heap of n stones, and we can use balance scales to compare weights of any two stones. In the class we discussed an algorithm that permits to find the heaviest and the second heaviest stone in $n + \lceil \log_2 n \rceil - 2$ weighing. Prove that this algorithm is optimal (no algorithm can do better than this in the worst case).
Hint : Apply the "adversarial argument."

3. Fake coins problems. *(i) A light fake coin.* We are given $n = 25$ coins. One of them is fake, which is lighter than all other (identical) coins. We can use balance scales to compare weights of any two *groups* of coins. How many operations do we need to find the fake coin ? We proved that an optimal strategy requires 3 weighings. In general, to find a light fake coin in a heap of n coins we need $\lceil \log_3 n \rceil$ weighings.

(ii) A fake coin with unknown weight. We are given $n = 12$ coins. One of them is fake, and can be lighter or heavier than all other (identical) coins. Again,

we can use balance scales to compare weights of any two groups of coins. How many operations do we need to find the fake coin ? We proved that an optimal strategy requires 3 weighings. For the same question with $n = 15$ we proved that 3 weighings are not enough.

Exercise 1.3. We are given n coins, and one of them is fake. All genuine coins have the same weights, the fake one is heavier or lighter. We can use balance scales to compare weights of any two groups of coins. How many weighings do we need to find the fake coin for (a) $n = 13$ and (b) $n = 14$.
Reminder : We discussed in the class that 3 *weighings are needed for* $n = 12$ *and* 4 *weighings are needed for* $n = 15$.

4. Combinatorial definition of the information quantity. Following Ralph Hartley, we say that the *quantity of information* in a *finite set* X is defined as

$$\mathrm{Inf}(X) := \log_2 |A|$$

(roughly speaking, $\mathrm{Inf}(X)$ is the number of *binary digits* needed to give a unique name to each element in A).

For a set $X \subset \mathbb{Z}^2$ we can define the "quantity of information" in X as well as in its projections on the first and the second coordinates (denoted $\pi_1[X]$ and $\pi_2[X]$ respectively). We have the following simple property :

$$\mathrm{Inf}(X) \le \mathrm{Inf}(\pi_1[X]) + \mathrm{Inf}(\pi_2[X]),$$

the quantity of information in X is not greater than the sum of the information quantities in two its components. This statement is equivalent to the obvious inequality

$$|X| \le |\pi_1[X]| \times |\pi_2[X]|.$$

Similarly, for a finite set $X \subset \mathbb{Z}^3$ we have

$$\mathrm{Inf}(X) \le \mathrm{Inf}(\pi_1[X]) + \mathrm{Inf}(\pi_2[X]) + \mathrm{Inf}(\pi_3[X]).$$

Exercise 1.4 (optional and difficult)**.** In what follows $\mathrm{Inf}(X)$ stands for Hartley's combinatorial information in a set X, i.e.,

$$\mathrm{Inf}(X) := \log_2 |X|.$$

Let A be a finite set in $\mathbb{Z}^3$. We denote by $\pi_{ij}[A]$ the projection of A onto the coordinates i and j (e.g., π_{13} applied to the point (x, y, z) gives (x, z)). Prove that

$$2 \cdot \mathrm{Inf}(A) \le \mathrm{Inf}(\pi_{12}[A]) + \mathrm{Inf}(\pi_{13}[A]) + \mathrm{Inf}(\pi_{23}[A]).$$

17.09.2019. Lecture 2.

1. Discussion of the homework : We discussed an algorithm for sorting an array of n elements and proved by induction that it runs time $O(n \log n)$ in the worst case. We proved (with the adversarial argument) that the search of the maximal and the second maximal element in an array of size n requires in the worst case $n + \lceil \log n \rceil - 2$ comparisons.

Exercise 2.1. We are given a heap of n stones, and we can use balance scales to compare weights of any two stones. We want to find in the heap two stones : the one with the maximal weight and the one with the minimal weight. How many weighing do we need (in the worst case) ? Propose a search algorithm and prove that it is optimal.

2. The game guess a number. We discussed another version of the game "guess a number," where the first player choses at random an integer number between 1 and n with (known in advance) probabilities $p_1, \ldots, p_n$, and the second player should reveal this number by asking questions with answers *yes* or *no*, with the minimal *on average* number of questions. We discussed several specific examples and suggested a general scheme of "modified dichotomy." In this strategy, on each step the second player divides all numbers (that have not been excluded earlier) into two groups with balanced measures, i.e., the sums of probabilities in both groups must be as close to each other as possible. We discovered a plausible approximation : the average number of steps in this strategy is close to

$$\sum_{i=1}^{n} p_i \log \frac{1}{p_i},$$

though we did not prove it formally.

3. Shannon's entropy. For a random variable α with n possible values $a_1, \ldots, a_n$ such that $\mathrm{Prob}[\alpha = a_i] = p_i$, we define its Shannon's entropy as

$$H(\alpha) := \sum_{i=1}^{n} p_i \log \frac{1}{p_i}$$

(with the usual convention $0 \cdot \log \frac{1}{0} = 0$). We proved several properties of Shannon's entropy.

Proposition 2.1. *For every random variable α distributed on a set of n values*

$$0 \leq H(\alpha) \leq \log n.$$

Moreover, $H(\alpha) = 0$ if and only if the distribution is concentrated at one point (one probability p_i is equal to 1, and the other p_j for $j \neq i$ are equal to 0), and $H(\alpha) = \log n$ if and only if the distribution is uniform ($p_1 = \ldots = p_n = \frac{1}{n}$).

Sketch of proof : We use the concavity of the function $\log x$ and Jensen's inequality for the concave functions.

Exercise 2.2. Prove that for every non-negative real number h there exists a random variable (distribution) α such that Shannon's entropy of α is equal to h.

Proposition 2.2. *For every random variable α and for every (deterministic) function F, Shannon's entropy of the random variable $\beta = F(\alpha)$ is not greater than Shannon's entropy of α.*

Sketch of proof : First of all, we observed that $H(\alpha) = H(\beta)$, if F is a bijection. Then, we proved that the entropy of a distribution decreases, when we merge together two points in this distribution ; in other words, $H(\alpha) \geq H(F(\alpha))$, if F merges together two points from the range of α and leaves distinct the other values of α. By iterating the basic "merging" operations, we prove the inequality $H(\alpha) \geq H(F(\alpha))$ for an arbitrary function F.

Given a pair of jointly distributed random variables (α, β) we can apply the definition of Shannon's entropy three times, with three protentially different distributions : we have Shannon's entropy of the entire distribution (denoted $H(\alpha, \beta)$) and the entropies of two marginals, $H(\alpha)$ and $H(\beta)$.

Proposition 2.3. *For every pair of jointly distributed random variables α and β*

$$H(\alpha, \beta) \leq H(\alpha) + H(\beta).$$

Sketch of proof : We used again the concavity of the function of logarithm and Jensen's inequality.

Exercise 2.3. Let α and β be two jointly distributed random variables. Prove that

$$H(\alpha, \beta) = H(\alpha) + H(\beta)$$

if and only if α and β are independent, i.e., for all i, j

$$\text{Prob}[\alpha = a_i \text{ and } \beta = b_j] = \text{Prob}[\alpha = a_i] \cdot \text{Prob}[\beta = b_j].$$

Exercise 2.4. Let us have a rooted binary tree with n leaves. Let l_i denote the length of the path from the root to the i-th leaf, $i = 1, \ldots, n$. Prove that

$$\sum_{i=1}^{n} \frac{1}{2^{l_i}} \leq 1.$$

17.09.2019. Lecture 3.

1. Discussion of the homework :

— Exercise 2.2 : *Every non-negative real number h there exists a random variable (distribution) α such that Shannon's entropy of α.* We proved it by sandwiching the real number h between $\log n$ and $\log n + 1$ (for an integer n) and by organizing a continuous transformation from a uniform distribution on n variables to a uniform distribution on $(n+1)$ variables. During this continuous transformation, we obtain distributions with all values of Shannon's entropy intermediate between $\log n$ and $\log(n+1)$.

— Exercise 2.3 : We proved that

$$H(\alpha, \beta) = H(\alpha) + H(\beta)$$

if and *only if* α and β are independent.

— Exercise 2.3 : *Let us have a rooted binary tree with n leaves, and let l_i denote the length of the path from the root to the i-th leaf, $i = 1, \dots, n$. Then* $\sum\limits_{i=1}^{n} \frac{1}{2^{l_i}} \leq 1$.

We discussed two proofs of this fact : (i) the induction by the hight of the tree and (ii) a direct proof with extending the given tree to an infinite complete binary tree and counting of the fractions of branches extending each of the leaves in the original tree.

2. Upper and lower bounds for the average number of questions in the game "guess a number". In this section we use Shannon's entropy to estimate the average number of questions needed in the randomized version of the game "guess a number" (with a probability distribution on the set of possible integers).

Lemma 3.1. For integer numbers $l_1, \dots, l_n$ such that

$$\sum_{i=1}^{n} \frac{1}{2^{l_i}} \leq 1$$

there exists a binary tree with n leaves such that the length of the path from the root to the i-th leaf is equal to l_i.

Theorem 3.1. In the game "guess a number" (with *yes or no* questions) with probabilities $p_1, \dots, p_n$, every strategy uses on average $\geq \sum\limits_{i=1}^{n} p_i \log \frac{1}{p_i}$ questions.

Sketch of proof : Denote by l_i the number of questions in the branch of the strategy the ends up with the answer i. The theorem claims that

$$\sum_{i=1}^{n} p_i l_i = \sum_{i=1}^{n} p_i \log 2^{l_i} \geq \sum_{i=1}^{n} p_i \log \frac{1}{p_i}.$$

To prove this bound, we apply Jensen's inequality (logarithm is concave) and use Exercise 2.3.

Theorem 3.2. In the game "guess a number" (with *yes or no* questions) where the number $i = 1, \ldots, n$ is chosen with probabilities $p_1, \ldots, p_n$, there exists a strategy that uses on average less than $\sum_{i=1}^{n} p_i \log \frac{1}{p_i} + 1$ questions.

Sketch of proof : We define $l_i := \lceil \log \frac{1}{p_i} \rceil$, notice that $\sum 2^{-l_i} \leq 1$, and use Lemma 3.1 to construct a strategy where each i-th leaf is on the distance l_i form the root.

3. Kraft's inequality. A *prefix code* is a set of strings $\{c_1, \ldots, c_n\}$ where no codeword c_i is a prefix of any other code word c_j in this set. A *uniquely decodable code* is a set of strings $\{c_1, \ldots, c_n\}$ such that for every string x there exists at most one representation

$$x = c_{i_1} \circ c_{i_2} \circ \ldots \circ c_{i_k}$$

(where $\circ$ denotes concatenation). Every prefix code is uniquely decodable, but not vice-versa. In what follows we assume that all codes contain only binary codewords (words in the alphabet of two letters).

Theorem 3.3. [Kraft's inequality] For every uniquely decodable code $\{c_1, \ldots, c_n\}$ we have

$$\sum_{i=1}^{n} 2^{-|c_i|} \leq 1.$$

Theorem 3.4. For every *uniquely decodable* code $\{c_1, \ldots, c_n\}$ there exists a *prefix code* $\{c'_1, \ldots, c'_n\}$ with the same lengths of the codewords, i.e., $|c'_i| = |c_i|$ for each i.

We mentioned the correspondence between strategies for the game *guess a number* and prefix codes. We also discussed Shannon–Fano encoding (a classical coding method which works pretty well in practice but is not necessary optimal).

Exercise 3.1. We are given a heap of n stones (for simplicity we assume that n is even), and we can use balance scales to compare weights of any two stones. We want to find in the heap two stones : the one with the *maximal weight* and the one with the *minimal weight*. Prove that these two stones can be found in at most $\frac{3n}{2} - 2$ weighings. Prove that this number of operations is optimal : any algorithm that solves this problem needs at least $\frac{3n}{2} - 2$ operations in the worst case.

Hint : At each stage of the search, we define the following four sets :

- `Potential-Max-or-Min` : the stones that have not yet participated in any comparison.
- `Potential-Max-not-Min` : the stones that have won at least one comparison but have not lost any comparison.
- `Potential-Min-not-Max` : the stones that have lost at least one comparison but have not won any comparison.

— `Not-Max-nor-Min` : the stones that have won at least one comparison and have lost at least one comparison.

Describe these sets at the beginning of the process (before the very first weighing) and at the very end of the process (when the maximal and the minimal stones are already found). Analyze how the stones travel between these sets after each comparison. Then propose an "adversarial strategy" that maximize the number of operations in the search.

Exercise 3.2 (optional ; no need to bring the solution for correction)**.** Write a program that computes Shannon's entropy for a distribution with given probabilities $(p_1, \dots, p_n)$. [You will need this program for the next homework.]

Exercise 3.3. (a) Let $S \subset \{a, b, c\}^n$ be the set of all strings with 50% of letters a, 25% of letters b, and 25% of letters c. Prove that there exists an injective mapping (a "text compressor")

$$\mathrm{Comp} : S \to \{0, 1\}^{3n/2}$$

that assigns to each word from S a string of $3n/2$ bits.

(b) Prove that there is no injective mapping

$$\mathrm{Comp}_U : \{a, b, c\}^n \to \{0, 1\}^{3n/2}$$

(a "text compressor" with this property does not exist).

01.10.2019. Lecture 4.

1. Discussion of the homework : solution of Exercise 3.3

2. Stirling's approximation. We proves a simplified version of Stirling's approximation for the factorial :

$$\log(N!) = N \log\left(\frac{N}{e}\right) + O(\log N).$$

A more precise approximation (not proven in the class) is

$$N! = \sqrt{2\pi N}\left(\frac{N}{e}\right)^N \cdot (1 + o(1)),$$

as $n \to \infty$.

3. Compressing binary strings with unbalances frequencies of bits. For every real number $p \in (0, 1)$ we denote by $B_{n,p}$ the set of all binary strings of length n with pn zeros and $(1 - p)n$ ones.

Proposition 4.1. If pn is an integer number then

$$|B_{n,p}| = 2^{h(p)n + O(\log n)},$$

where $h(p) := p \log \frac{1}{p} + (1-p)\frac{1}{1-p}$.
Sketch of the proof : The number of elements in $B_{n,p}$ is equal to

$$\binom{n}{pn} = \frac{n!}{(pn)!((1-p)n)!}.$$

Applying Stirling's approximation for the factorials, we obtain the requires estimation for the cardinality of $B_{n,p}$.

4. Expectation and variance of random variables (reminder). Let α be random variables distribute on $\mathbb{R}$. In what follows we assume that the distribution is concentrated on a finite set of real numbers. Let $\text{Prob}[\alpha = r_i] = p_i$ for $i = 1, \ldots, n$, and $\sum\limits_{i=1}^{n} p_i = 1$.

Definition. Expectation of α is defined as

$$E(\alpha) := \sum_{i=1}^{n} p_i r_i.$$

Simple properties of the expectation :

- $E(\alpha + c) = E(\alpha) + c$ for every constant c;
- $E(c \cdot \alpha) = c \cdot E(\alpha)$ for every constant c;
- $E(\alpha + \beta) = E(\alpha) + E(\beta)$ for every pair of jointly distributed α and β;
- $E(\alpha \cdot \beta) = E(\alpha) \cdot E(\beta)$ for all *independent* α and β;

Proposition 4.2. [Markov inequality] If the distribution of α is concentrated on only non-negative real numbers, then for every real number T

$$\text{Prob}[\alpha \geq T] \leq \frac{E(\alpha)}{T}.$$

Definition. Variance of α is defined as $\text{var}(\alpha) := E\left((\alpha - E(\alpha))^2\right)$.

Simple properties of the variance :

- $\text{var}(\alpha + c) = \text{var}(\alpha)$ for every constant c;
- $\text{var}(c \cdot \alpha) = c^2 \cdot \text{var}(\alpha)$ for every constant c;
- $\text{var}(\alpha + \beta) = \text{var}(\alpha) + \text{var}(\beta)$ for every pair of *independent* α and β.

Proposition 4.3. [the Chebyshev inequality] For every real number T

$$\text{Prob}[|\alpha - E(\alpha)| \geq T] \leq \frac{\text{var}(\alpha)}{T^2}.$$

Example 1. Let α be a random variable such that $\text{Prob}[\alpha = 1] = p$ and $\text{Prob}[\alpha = 0] = 1 - p$. Then $E(\alpha) = p$ and $\text{var}(\alpha) = p(1-p)$.

Example 2. Let α_i, $i = 1, \ldots, n$ be a sequence of independent identically distributed random variable such that $\text{Prob}[\alpha_i = 1] = p$ and $\text{Prob}[\alpha_i = 0] = 1-p$ for every i. Then

$$E(\alpha_1 + \ldots + \alpha_n) = pn$$

and

$$\mathrm{var}(\alpha_1 + \ldots + \alpha_n) = p(1-p)n.$$

From the Chebyshev inequality we obtain

$$\mathrm{Prob}\left[\left|\frac{\alpha_1 + \ldots + \alpha_n}{n} - p\right| \geq \delta\right] \leq \frac{p(1-p)}{\delta^2 n}$$

5. Shannon's coding theorem for block coding.

Theorem 4.1. Let $\alpha_1, \ldots, \alpha_n$ be a sequence of independent identically distributed random variables, and let $\rho > H(\alpha_i)$. Denote A the range (the alphabet) of all α_i and $k(n) := \lceil \rho n \rceil$. Then there exists a sequence of functions (encoding and decoding)

$$\begin{array}{rcccc} C_n & : & A^n & \to & \{0,1\}^{k(n)}, \\ D_n & : & \{0,1\}^{k(n)} & \to & A^n \end{array}$$

such that probability of the decoding error

$$\epsilon_n := \mathrm{Prob}_{(\alpha_1 \ldots \alpha_n)}\left[D_n(C_n(a_{i_1} \ldots a_{i_n})) = a_{i_1} \ldots a_{i_n}\right]$$

tends to 0 as $n \to \infty$. (Here $a_{i_1} \ldots a_{i_n}$ is a randomly chosen sequence of values for $(\alpha_1, \ldots, \alpha_n)$. In other words, each letter a_{i_s} is chosen with the distribution α_i, independently of other letters.)

Theorem 4.2. Let $\alpha_1, \ldots, \alpha_n$ be a sequence of independent identically distributed random variables, and let $\rho < H(\alpha_i)$. Denote A the range (the alphabet) of all α_i and $k(n) := \lceil \rho n \rceil$. Then for any sequence of functions

$$\begin{array}{rcccc} C_n & : & A^n & \to & \{0,1\}^{k(n)}, \\ D_n & : & \{0,1\}^{k(n)} & \to & A^n \end{array}$$

probability of the decoding error

$$\epsilon_n := \mathrm{Prob}_{(\alpha_1 \ldots \alpha_n)}\left[D_n(C_n(a_{i_1} \ldots a_{i_n})) = a_{i_1} \ldots a_{i_n}\right]$$

does *not* tend to 0 as $n \to \infty$.

Remark : Theorem 4.2 can be made even stronger : in fact, for $\rho < H(\alpha_i)$ probability of the decoding error ϵ_n tends to 1 as $n \to \infty$. We did not prove Theorem 4.2 in the class. However, we suggest to think of the proof of this theorem (of its the weak version, as stated above).

Exercise 4.1. Construct prefix code with minimal average length for the following probability distributions :
(a) 0.6, 0.4
(b) 0.4, 0.3, 0.3
(c) 0.4, 0.3, 0.2, 0.1

Exercise 4.2. Show that for every triple of non-negative real numbers h_1, h_2, h_3 there exists a pair of jointly distributed random variables (α, β) such that

$$\begin{array}{rcl} H(\alpha) & = & h_1 + h_2 \\ H(\beta) & = & h_1 + h_3 \\ H(\alpha, \beta) & = & h_1 + h_2 + h_3 \end{array}$$

Exercise 4.3. We are given $n = 13$ coins, and one of them is fake. All genuine coins have the same weight, the fake one can be heavier or lighter. The position of the fake coin (between 1 and 13) and its relative weight (whether it is heavier or lighter than the genuine coins) are chosen at random, and all $2 \times 13 = 26$ variants have the same probability $1/26$. We can use balance scales to compare weights of any two groups of coins.

(i) Find a strategy that discovers the fake coin with minimal *on average* number of weighings.

(ii) Compute Shannon's entropy of each weighings that can be used in an optimal strategy (at least for the 1st and or the 2nd weighing in each branch of the strategy).

Hint 1 : There are several different optimal strategies, but each of them uses 3 operation.

Hint 2 : It is helpful to start the solution with question (ii), and construct an optimal strategy by choosing on each stage the weighing that brings the maximal possible value of entropy.

08.10.2019. Lecture 5.

1. Discussion of the homework : solution of Exercise 4.1 Huffman's algorithm that constructs an optimal prefix codes for a given distribution.

A brief discussion of Exercise 4.2. Venn-like diagrams for entropies of pairs ad triples of random variables. An example of a distribution (α, β, γ) such that

$$\begin{array}{l} H(\alpha) = H(\beta) = H(\gamma) = 1, \\ H(\alpha, \beta) = H(\beta, \gamma) = H(\alpha, \gamma) = 2, \\ H(\alpha, \beta, \gamma) = 2. \end{array}$$

Discussion of a heuristic algorithm (the *"greedy" entropic* search) for Exercise 4.3. Solution of Exercise 1.3 (b).

2. Conditional entropy and mutual information.

Definition 1. Let (α, β) be jointly distributed random variables, with $p_{ij} = \mathrm{Prob}[\alpha = 1_i \& \beta = b_j]$. For each value b_j we have a conditional distribution on the values of α with probabilities

$$p'_i = \mathrm{Prob}[\alpha = 1_i \,|\, \beta = b_j] = \frac{\mathrm{Prob}[\alpha = 1_i \& \beta = b_j]}{\mathrm{Prob}[\beta = b_j]}.$$

This conditional distribution has Shannon's entropy ; we denote it $H(\alpha \,|\, \beta = b_j)$.

Definition 2. We define the entropy of α conditional on β as the average

$$H(\alpha \,|\, \beta) := \sum_j \mathrm{Prob}[\beta = b_j] \cdot H(\alpha \,|\, \beta = b_j).$$

In the class we proved several properties of *conditional entropy* :

- $H(\alpha, \beta) = H(\alpha \,|\, \beta) + H(\beta)$
- $H(\alpha \,|\, \beta) \leq H(\alpha)$
- $H(\alpha \,|\, \beta) = H(\alpha)$ if and only if α and β are independent

Definition 3. We define the information in α on β as

$$I(\alpha : \beta) := H(\beta) - H(\alpha \,|\, \beta).$$

In the class we proved several properties of the *mutual information* :

- $I(\alpha : \beta) = I(\beta : \alpha) = H(\alpha) + H(\beta) - H(\alpha, \beta)$
- $I(\alpha : \beta) \geq 0$
- $I(\alpha : \beta) = 0$ if and only if α and β are independent
- $I(\alpha : \beta) \leq H(\alpha)$
- $I(\alpha : \beta) \leq H(\beta)$

Definition 4. We define the information in α on β conditional on γ as

$$I(\alpha : \beta \,|\, \gamma) := H(\beta \,|\, \gamma) - H(\alpha \,|\, \beta, \gamma).$$

3. Linear inequalities for Shannon's entropy. We proved several inequalities for Shannon's entropy.

Proposition 5.1. The inequality

$$H(\gamma) \leq H(\gamma \,|\, \alpha) + H(\gamma \,|\, \beta) + I(\alpha : \beta)$$

is true for all jointly distributed α, β, γ.

Proposition 5.2. The inequality

$$2H(\alpha, \beta, \gamma) \leq H(\alpha, \beta) + H(\alpha, \gamma) + H(\beta, \gamma)$$

is true for all jointly distributed α, β, γ.

We discussed how Proposition 5.2. helps to solve Exercise 1.4.

4. Elements of information-theoretic cryptography. We discussed the *one-time pad* symmetric encryption scheme (the Vernam cipher) and proved that it is optimal in the following sense :

Theorem 5.1 [Shannon] Let random variables m, e, k denote the original *message*, the *encrypted* message, and the secret *key*. A symmetric encryption scheme is *perfect*, if $H(m\,|\,, e, k) = 0$ (the initial message can be uniquely reconstructed given the encrypted message and the secret key) and $I(e : m) = 0$ (the encrypted message contains no information on the initial message). Then

$$H(k) \geq H(m),$$

i.e., the size (entropy) of the secret key must be at least as long as the size (entropy) of the message.

Secret sharing. We discussed in the class the notion of a *perfect secret sharing*, with simple classical examples. In this setting, a *secret* is a random variable S_0 (usually a uniform distribution on some finite set), which is understood as a distribution on possible values of a secret keys. We want to "distribute" this secret among n participants of the project so that (i) every "authorized" group of participants could reconstruct uniquely the value of S_0, and (ii) every "non-authorized" group of participants gets no information about the secret. Technically, this means that we include the random variable S_0 in a joint distribution $(S_0, S_1, \ldots, S_n)$ (where S_0 is the secret and $S_1 \ldots S_n$ are *shares* assigned to each participant) so that the conditions (i) and (ii) are satisfied.

Example 1. Let us require that only all n participants now the secret S_0, and every group of less than n participants gets no information on S_0. In case when S_0 is a uniform distribution on $\{0,1\}^k$, there is a simple scheme satisfying the conditions (i) and (ii) : the "shares" of the secret $S_1, \ldots, S_n$ are independent and uniform distribution on $\{0,1\}^k$, and S_0 is the bitwise XOR of them,

$$S_0 = S_1 \oplus \ldots \oplus S_n.$$

Then it is easy to verify that

$$(i)\ H(S_0\,|\,S_1, \ldots, S_n) = 0$$

and

$$(ii)\ H(S_0\,|\,S_1, \ldots, S_{i-1}, S_{i+1}, \ldots, S_n) = H(S_0).$$

Example 2. Let us choose a parameter (threshold) t between 1 and n and require that (i) every group of at east t participants knows the secret, and (ii) every group of less than t participants gets no information about the secret. For simplicity, we assume that S_0 is a uniform distribution on $\mathbb{Z}/p\mathbb{Z}$ for a prime number p (in what follows we assume that $n < p$).

In this setting the secret sharing can be implemented in Shamir's scheme : we fix some elements $x_0, x_1, \ldots, x_n \in \mathbb{Z}/p\mathbb{Z}$ and define the jount distribution $(S_0, S_1, \ldots, S_n)$ as follows : let $a_0, \ldots a_{t-1}$ be independent uniformly chosen elements in $\mathbb{Z}/p\mathbb{Z}$, and respectively

$$Q(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1}$$

be a randomly chosen polynomial of degree less than t (again, over the field $\mathbb{Z}/p\mathbb{Z}$), and

$$S_i := Q(x_i) \text{ for } i = 0, 1, \ldots, n.$$

Then it can be shown that

$$(i)\ H(S_0 \,|\, S_{i_1}, \ldots, S_{i_t}) = 0$$

for all $1 \leq i_1 < \ldots i_t \leq n$ (given t different points (x_i, S_i) on the graph of the polynomial $Q(x)$, we can reconstruct the coefficients of $Q(x)$ and therefore compute $S_0 = Q(x_0)$) and

$$(ii)\ H(S_0 \,|\, S_{i_1}, \ldots, S_{i_{t-1}}) = \log p = H(S_0)$$

(if we know only $t-1$ points $(x_i, Q(x_i))$ on the graph of the polynomial, than *all* values of $Q(x_0)$ are possible and equiprobable).