Санкт-Петербургский Государственный Университет

Факультет математики и компьютерных наук

Теория информации

Конспект основан на лекциях
Дмитрия Олеговича Соколова

7 февраля 2024 г.

Санкт-Петербургский государственный университет

Факультет математики и компьютерных наук СПбГУ

Конспект основан на лекциях по теории информации, прочитанных Дмитрием Олеговичем Соколовым студентам Факультета математики и компьютерных наук Санкт-Петербургского государственного университета в весеннем семестре 2019–2020 учебного года.

В конспекте содержится часть материала 4-ого семестра курса теоретической информатики.

Авторы:

Михаил Опанасенко
Сергей Лучинин
Дмитрий Соколов

Последняя версия и исходный код:

https://www.overleaf.com/read/mxgvmtxjzfpg

Сайт СПБГУ: https://spbu.ru.
Сайт факультета МКН: https://math-cs.spbu.ru.

Оглавление

Теория информации **1**

1 Меры информации . . . 1
1.1 Информация по Хартли . . . 1
1.2 Информация по Шеннону . . . 4
1.3 Применение энтропии . . . 7
2 Кодирование . . . 9
2.1 Однозначно декодируемые и префиксные коды . . . 10
2.2 «Почти оптимальные» коды . . . 11
2.3 Кодирование с ошибками . . . 13
3 Криптография . . . 15
3.1 Шифрование и схемы разделения секрета . . . 15
3.2 Схема Шамира . . . 17
3.3 Идеальные схемы разделения секрета . . . 17
4 Коммуникационная сложность . . . 18
4.1 Прямоугольники и метод *Fooling Set* . . . 19
4.2 Метод ранга . . . 20
4.3 Балансировка протоколов . . . 21
4.4 Теорема Карчмера–Вигдерсона . . . 22
4.5 Сложные функции . . . 23
4.6 Вероятностные коммуникационные протоколы . . . 25
4.7 Вычисление функции индексирования . . . 26
4.8 Внутреннее и внешнее информационное разглашение . . . 28
5 Колмогоровская сложность . . . 29
5.1 Способы описания и их базовые свойства . . . 29
5.2 Условная колмогоровская сложность . . . 32
5.3 Применение колмогоровской сложности . . . 34
5.4 Случайные строки по Мартин-Лёфу . . . 35

Теория информации

1 Меры информации

Пусть дано некоторое множество объектов. Мы хотим ввести некоторую *меру информации*, то есть хотим понять, сколько информации мы узнаём, получая некоторый элемент данного множества. Одна из общепринятых мер информации — количество бит. Её недостаток — целые значения.

1.1 Информация по Хартли

Определение. Пусть A — некоторое конечное множество. *Информация в множестве A* — это число

$$\chi(A) = \log|A|.$$

Оно соответствует интуитивному представлению о том, что необходимо $\log|A|$ бит для выделения некоторого элемента множества. Отметим, что число $\chi(A)$ может быть нецелым.

Замечание. В отличие от курса анализа, под log мы везде понимаем логарифм по основанию 2.

Утверждение 1.1. Пусть $A \subset X \times Y$ — конечное двумерное множество, A_X — его проекция на X, A_Y — на Y. Тогда выполнены следующие свойства:

(1) $\chi(A) \geqslant 0$;

(2) $\chi(A_X) \leqslant \chi(A)$, $\chi(A_Y) \leqslant \chi(A)$;

(3) $\chi(A) \leqslant \chi(A_X) + \chi(A_Y)$.

Доказательство. Очевидно. ■

Утверждение 1.2. Пусть $A \subset X_1 \times X_2 \times X_3$. Докажите неравенство

$$2\chi(A) \leqslant \chi(A_{12}) + \chi(A_{23}) + \chi(A_{13}),$$

где A_{ij} — проекция A на $X_i \times X_j$.

Доказательство. Неравенство из условия равносильно тому, что

$$|A|^2 \leqslant |A_{12}| \cdot |A_{23}| \cdot |A_{13}|,$$

так как log — возрастающая функция. Пусть

$$A_{12} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}.$$

Тогда A состоит из множеств

$$B_i = \{(x_i, y_i, a_1), \ldots, (x_i, y_i, a_{j_i})\},$$

и $|A| = j_1 + j_2 + \cdots + j_n$. Рассмотрим теперь множество $A_{13} \cdot A_{23}$, состоящее из точек (x_i, a_j, y_s, a_k). Тогда понятно, что все точки вида (x_i, a_k, y_i, a_s), где $k, s \leqslant j_i$, лежат в нашем множестве, а таких точек $j_1^2 + j_2^2 + \cdots + j_n^2$. Значит,

$$|A_{13}| \cdot |A_{23}| \geqslant |A_{23} \cdot A_{13}| \geqslant j_1^2 + j_2^2 + \ldots + j_n^2.$$

Таким образом, $|A_{12}| \cdot |A_{23}| \cdot |A_{13}| \geqslant n(j_1^2 + j_2^2 + \cdots + j_n^2) \geqslant (j_1 + j_2 + \cdots + j_n)^2 = |A|^2$. ■

Определение. Пусть A — двумерное множество с проекциями X и Y. Количество информации, содержащееся в A, если мы уже знаем вторую координату, определяется следующим образом:

$$\chi_{Y|X}(A) = \max_x \big(\log |A_x| \big),$$

где A_x — сечение A по координате x.

Интуитивно — это достаточное количество бит, нужное для кодирования элемента, зная его первую проекцию. Но это определение «плохое», так как разным элементам могут соответствовать сечения разных размеров.

Нетрудно проверить, что при таком определении выполнено неравенство

$$\chi(A) \leqslant \chi(A_Y) + \chi_{X|Y}(A).$$

В дальнейшем иногда мы будем обозначать множество $\{1, 2, \ldots, n\}$ через $[n]$.

Задача 1.1. Есть два игрока, первый загадывает число от 1 до n. Сколько вопросов с ответом «да/нет» необходимо задать второму игроку, чтобы угадать число? При этом у задачи есть два варианта: с *неадаптивной* стратегией, когда второй игрок пишет все вопросы заданы заранее, и *адаптивной* стратегией, когда второй игрок задаёт очередной вопрос, зная ответы на все предыдущие.

Решение. Второй игрок может спросить каждый бит числа n в двоичной записи; поэтому количество запросов не превосходит $h = \lceil \log n \rceil$. Покажем, что это и есть оптимальное количество вопросов.

Пусть Q_i — ответ на i-ый вопрос (один бит), N — искомое число,

$$B := Q_1 \times Q_2 \times \cdots \times Q_h.$$

Посмотрим на множество пар (N, B) по всем возможным N и B. Корректность протокола означает, что если мы знаем все Q_i, то можем определить число, то есть

$\chi_B([n]) = 0$. Легко заметить, что $\chi(Q_i) \leqslant 1$. Тогда

$$\log n \leqslant \chi(N, B) \leqslant \sum_{i=1}^{h} \chi(Q_i) + \chi_B([n]) = \sum_{i=1}^{h} \chi(Q_i) \leqslant h.$$

Таким образом, $h \geqslant \log n$, доказана нижняя оценка.

Ту же оценку можно было легко получить и другими, более простыми способами, но метод выше обобщается на гораздо более сложные ситуации. ■

Задача 1.2. Найдите оптимальную по цене стратегию угадывания монетки, если за каждый ответ «да» второй игрок платит 1 монету, а каждый ответ «нет» — 2 монеты.

Решение. Интуитивно понятно, что в оптимальной стратегии должно быть выполнено соотношение $2\chi_{Q_i=1}([n]) = \chi_{Q_i=0}([n])$ — иначе можно будет найти такое число N, что второму игроку придётся потратить больше монет. Формализуем эту стратегию.

Пусть Q_i — ответ на вопрос «верно ли, что загаданное число N лежит в множестве T_i?» Пусть X_i — множество элементов, в котором может лежать N после первых i вопросов. Тогда на каждом шаге мы хотим добиться следующего равенства:

$$2(\log|X_i| - \log|X_i \cap T_i|) = \log|X_i| - \log|X_i \setminus T_i| \iff$$
$$\log|X_i| = 2\log|X_i \cap T_i| - \log|X_i \setminus T_i| \iff |X_i| = \frac{|X_i \cap T_i|^2}{|X_i \setminus T_i|}.$$

Обозначая $|X_i| = k$, $|T_i| = t$, получаем:

$$k = t^2/(k-t) \iff t^2 = k(k-t) = k^2 - kt \iff$$
$$t^2 + kt - k^2 = 0 \iff t = \frac{-k \pm \sqrt{k^2 + 4k^2}}{2} = k\left(\frac{-1 \pm \sqrt{5}}{2}\right).$$

Таким образом, на каждом шагу нужно выбирать такое T_i, что $\varphi|T_i| = |X_i|$, где φ — золотое сечение. ■

Размер дерева расщепления для данной задачи позволяет доказывать нижние оценки на различные алгоритмы для задачи выполнимости булевых формул. А для доказательства нижней оценки на размер дерева используется подобная игра с монетками («A lower bound for the pigeonhole principle in tree-like Resolution by asymmetric Prover–Delayer games»).

***Пример 1.1.** Подобная стратегия применяется и в некоторых более современных задачах. Пусть есть $n+1$ голубь и n клеток. По принципу Дирихле нельзя посадить голубей в клетки таким образом, чтобы каждый сидел в клетке, и в одной клетке было бы не более одного голубя. Пусть $x_{ij} = 1$, если i-ый голубь сидит в j-ой клетке, и $x_{ij} = 0$, если это не так. Тогда эти условия можно записать следующим образом:

(1) $\prod_{j=1}^{n}(1 - x_{ij}) = 0$;

(2) $x_{ij} \cdot x_{i'j} = 0$ для всех i, i', j, где $i \neq i'$.

Для того, чтобы по данной рассадке определить, какое из условий нарушено, нужна как раз игра с монетками.

Задача 1.3. Рассмотрим следующую задачу: даны n монеток, из которых одна фальшивая и имеет другой вес, и рычажные весы. Вопрос — можно ли за m взвешиваний определить фальшивую монету? Решите задачу в следующих вариантах:

(1) $n = 30$, $m = 3$.

(2) $n = 15$, $m = 3$.

(3) $n = 14$, $m = 3$.

Решение. В отличие от предыдущих задач, каждое взвешивание приносит больше информации: $\chi(Q_i) \leqslant \log 3$, так как возможны 3 ответа на каждый вопрос.

(1) При правильном протоколе должно быть выполнено неравенство

$$\log 30 = \chi([30]) \leqslant \sum_{i=1}^{3} \chi(Q_i) + \chi_B([30]) = \chi(Q_1) + \chi(Q_2) + \chi(Q_3) \leqslant \log 27,$$

что неверно. Значит, ответ — нет.

(2) В случае $n = 15$ оценка выше не даёт требуемого результата. Если добавить также условие, что надо определить, какая монета тяжелее, то надо рассматривать множество $[15] \times \{0, 1\}$, где 0 означает, что монета фальшива; и тогда верхняя оценка сработает.

Пусть надо только определить фальшивую монету. Заметим, что если хотя бы при одном взвешивании не было достигнуто равновесие, то мы можем определить не только фальшивую монету, но и то, тяжелее она или легче обычных. Пусть монетка, получающаяся как ответ при трёх равновесиях, имеет номер k. Тогда реально мы определяем информацию множества

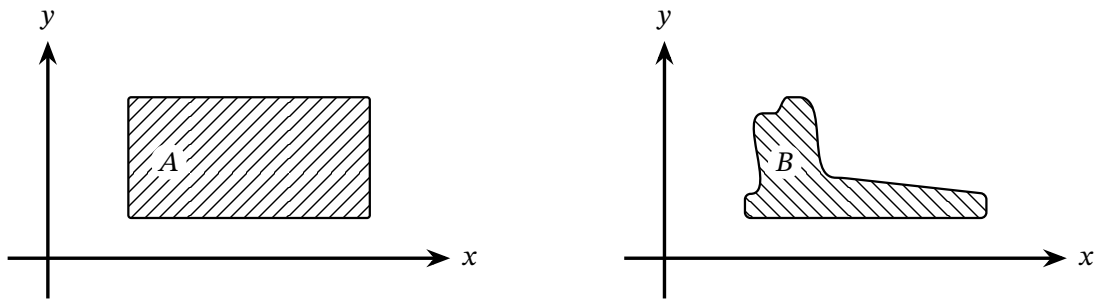
$$([15] \setminus \{k\}) \times \{0, 1\} \cup \{k\}$$

порядка 29. Поскольку $29 > 27$, ответ по-прежнему нет.

(3) Поскольку $2 \cdot 13 + 1 = 27$, предыдущее рассуждение не работает. Однако ответ всё ещё нет, но для доказательства нам понадобится некоторая более продвинутая теория. ■

1.2 Информация по Шеннону

В прошлом параграфе мы увидели ряд проблем, возникающих при работе с информацией по Хартли. С одной стороны, у нас есть задача про 27 монет и 3 взвешивания, которую мы не понимаем как решать. С другой — определение «условной информации» плохо описывает наше множество. Например, для следующих множеств



(a) (b)

выполнено равенство $\chi_{Y|X}(A) = \chi_{Y|X}(B)$, хотя сами множества ничем не похожи друг на друга (даже с точки знания количества объектов в них).

Попробуем обобщить понятие информации для решения данных проблем. Введём новую меру информации μ, согласованную с определением по Хартли. Раньше мы предполагали, что все элементы в множестве A одинаковы. Теперь предположим, что каждый элемент появляется с некоторой вероятностью p_n; то есть μ будет задаваться уже не на множестве, а на распределении. В этих терминах свойство согласованности можно выразить следующим образом:

(1) $\mu(U_n) = \log n$, где U_n — равномерное распределение на n объектах (это и есть согласованность с предыдущим определением);

(2) $\mu(p) \geqslant 0$, где p — любое распределение;

(3) $\mu(p, q) = \mu(p) + \mu(q)$, где p, q — независимые распределения.

Мы можем дополнить этот набор аксиом свойством «непрерывности», а также утверждением «согласованности» с определением условной вероятности. Тогда полный набор аксиом можно переписать в следующем виде:

(1) *монотонность*: если M, M' — равномерные распределения на $m \geqslant m'$ объектах соответственно, то $\mu(M) \geqslant \mu(M')$.

(2) *аддитивность*: $\mu(p, q) = \mu(p) + \mu(q)$, где p, q — независимые распределения;

(3) *непрерывность*: мера $\mu(B_p)$ непрерывна по p, где B_p — распределение нечестной монетки, которая выпадает решкой с вероятностью p, и орлом с вероятностью $1 - p$;

(4) выполнено равенство

$$\mu(B, X) = \mu(B) + \Pr[B = 0] \cdot \mu(X \mid B = 0) + \Pr[B = 1] \cdot \mu(X \mid B = 1),$$

где B — распределение нечестной монетки, X — произвольное распределение, Pr — вероятность, а $\mu(\cdot \mid \cdot)$ — условная вероятность.

Тогда можно доказать (мы этого делать не будем), что мера μ с точностью до мультипликативной константы определяется по формуле

$$h(X) = \sum p_i \log \frac{1}{p_i}.$$

Эта мера информации называется *энтропией*. Иногда она обозначается как $H(X)$.

Примеры.

1. Равномерное распределение: вероятность выпадения каждого элемента равна $\frac{1}{n}$. Поэтому
$$h(U_n) = \sum_{k=1}^{n} \frac{1}{n} \log n = \log n.$$
2. Нечестная монетка:
$$h(B_p) = p \log \frac{1}{p} + (1-p) \log \frac{1}{1-p}$$
— *бинарная энтропия*. Её часто обозначают просто как $H(p)$.

Поскольку теорему Шеннона мы оставили без доказательства, то нужно проверить, что энтропия удовлетворяет нашим аксиомам.

Утверждение 1.3. Энтропия $h(X)$ обладает следующими свойствами:

1. $h(X) \leqslant \log |X|$.
2. $h(X,Y) \leqslant h(X) + h(Y)$, где X, Y — произвольные распределения.

Доказательство.

1. Это просто неравенство Йенсена:
$$h(X) = \sum_{i=1}^{n} p_i \log \frac{1}{p_i} \leqslant \log \left(\sum_{i=1}^{n} p_i \frac{1}{p_i} \right) = \log n = \log |X|.$$
2. Положим $p_{ij} = \Pr[X = i, Y = j]$, $p_{i*} = \Pr[X = i]$, $p_{*j} = \Pr[Y = j]$. Заметим, что
$$p_{i*} = \sum_j p_{ij}, \qquad p_{*j} = \sum_i p_{ij},$$
— вероятность того, что выпал элемент i, равна вероятности того, что выпал элемент i и какой-то элемент j в Y. Тогда
$$h(X,Y) = \sum_{i,j} p_{ij} \cdot \log \frac{1}{p_{ij}},$$
$$h(X) + h(Y) = \sum_i p_{i*} \cdot \log \frac{1}{p_{i*}} + \sum_j p_{*j} \log \frac{1}{p_{*j}} = \sum_{ij} \left(p_{ij} \cdot \log \frac{1}{p_{i*}} + p_{ij} \cdot \log \frac{1}{p_{*j}} \right).$$
Тогда по неравенству Йенсена
$$h(X,Y) - h(X) - h(Y) = \sum_{ij} p_{ij} \log \frac{p_{i*} p_{*j}}{p_{ij}} \leqslant \log \left(\sum_{i,j} p_{i*} p_{*j} \right) = \log 1 = 0.$$

Отметим, что если X и Y независимы, то $p_{i*}p_{*j} = p_{ij}$, и мы получаем равенство $h(X,Y) = h(X) + h(Y)$. ■

Определение. *Условная энтропия* определяется как энтропия распределения на прямой $Y = b$:

$$h(X \mid Y = b) = \sum_i \Pr[X = i \mid Y = b] \cdot \log \frac{1}{\Pr[X = i \mid Y = b]}$$

Тогда $h(X \mid Y)$ — то, сколько находится в среднем информации по X, определяется как математическое ожидание условной энтропии:

$$h(X \mid Y) = \sum_j h(X \mid Y = j) \cdot \Pr[Y = j].$$

Утверждение 1.4.

1. $h(X \mid Y) \geqslant 0$, $h(f(Y) \mid Y) = 0$.
2. $h(X,Y) = h(X) + h(Y \mid X)$.
3. $h(X) \geqslant h(X \mid Y)$.

Доказательство.

1. Упражнение.
2. Упражнение.
3. По неравенству Йенсена для логарифма:

$$h(X \mid Y) - h(X) = \sum_{i,j} \left(p_{ij} \log \frac{1}{p_{i|j}} - p_{ij} \log \frac{1}{p_{i*}} \right) = \sum_{i,j} p_{ij} \cdot \log \frac{p_{i*}}{p_{i|j}} \leqslant 0,$$

где $p_{i|j} = \Pr[X = i \mid Y = j]$. ■

Упражнение. Докажите, что $h(X \mid Y) \geqslant h(X \mid Y, Z)$.

1.3 Применение энтропии

Теперь мы готовы решить пункт 3 задачи 1.3 про 14 монеток и 3 взвешивания.

Решение. Предположим, что в стратегии при трёх равенствах мы получаем монету с номером i. Как мы помним, нельзя определить, тяжелее она или легче других монет, в то время как для всех остальных монет это узнать можно. Это значит, что в дереве решения (в нём 27 листьев), число i встречается среди листьев только один раз, а остальные индексы — два раза, причём один раз в ветке «меньше», а в другой раз в ветке «больше». Зададим следующее распределение на монетках:

$$p_k = \begin{cases} \frac{1}{27}, & \text{если } k = i, \\ \frac{2}{27}, & \text{если } k \neq i, \end{cases}$$

и распределение на парах из монетки и больше/меньше: $p_{(k,<)} = p_{(k,>)} = p_k/2$.

С другой стороны, нетрудно проверить, что это распределение соответствует равновероятному выбору листа. Энтропия этого распределения (раз оно равномерно) равно логарифму количества листьев, то есть $h(p) = \log 27 = 3\log 3$.

Поймём теперь, что такое $h(p)$. Есть 3 взвешивания, назовём их q_1, q_2, q_3. Пусть $p(q_1, q_2, q_3)$ — это вершина, в которую мы пришли по какой-то ветке. Тогда

$$h(p \mid q_1, q_2, q_3) = 0,$$

так как p — функция от q_1, q_2, q_3. Значит,

$$h(p) \leqslant h(q_1, q_2, q_3) + h(p \mid q_1, q_2, q_3) = h(q_1, q_2, q_3) \leqslant h(q_1) + h(q_2) + h(q_3).$$

Однако $h(q_j) \leqslant \log 3$, так как мы выбираем из трёх вариантов, а энтропия не превосходит логарифма. Таким образом, единственная возможность, когда неравенство выполнено, это когда все три исхода равновероятны. А это значит, что на первом шаге мы с равной вероятностью идём по трём разным веткам от вершины дерева.

Покажем, что для первого шага это невозможно. Пусть взвешивается по k монет на каждой чаше. Вероятность того, что правая чаша перевесила, равна $\frac{2k}{27}$, так как либо фальшивая монета легче и лежит слева, либо она тяжелее и лежит справа, а вероятности этих событий совпадают и равны $k/27$.

Чтобы это число равнялось одной трети, нужно взять $k = 4.5$, что невозможно. Противоречие. ■

Утверждение 1.5 (оценка на биноминальные коэффициенты). Для произвольного n и $k \leqslant n/2$ выполнено неравенство

$$C = \sum_{i=0}^{k} \binom{n}{i} \leqslant 2^{nh(\frac{k}{n})}.$$

Доказательство. Дано n объектов, из них выбирается не более, чем k штук. Смотрим на равномерное распределение X по таким множествам (состоящих из $\leqslant k$ элементов). Тогда $h(X) = \log C$. При этом

$$h(X) \leqslant \sum h(X_i),$$

где X_i — вероятность того, что мы выбрали i-ый элемент. Все эти распределения одинаковы, то есть $h(X) \leqslant nh(X_1)$. Наконец, вероятность, с которой мы можем выбрать первый элемент, не превосходит $\frac{k}{n}$, то есть $h(X_1) = h(\frac{k}{n})$ (так как мы берём распределение на множествах мощности не более k), откуда легко следует искомое неравенство. ■

Пусть дан ориентированный граф без кратных рёбер и петель. Упорядоченная тройка (x, y, z) вместе в рёбрами из x в y, из y в z, и из z в x, называется *треугольником*. *Угол* — упорядоченная тройка вершин (x, y, z) вместе с рёбрами из x в y и в z. В частности, любое ребро является углом (можно взять $y = z$).

Теорема 1.6. В любом графе число треугольников не превосходит числа углов.

Доказательство. Пусть X, Y, Z — случайные величины, соответствующие первой, второй и третьей вершине треугольника в равномерном распределении на треугольниках соответственно. Тогда $H(X,Y,Z) = \log|\triangle|$. С другой стороны,

$$H(X,Y,Z) = H(X) + H(Y,Z \mid X) = H(X) + H(Y \mid X) + H(Z \mid X,Y).$$

Заметим, что если убрать X из $H(Z \mid X,Y)$, то энтропия только возрастёт. Значит,

$$H(X,Y,Z) \leqslant H(X) + H(Y \mid X) + H(Z \mid Y).$$

Картинка симметрична (можно получить одно распределение из другого циклическим сдвигом), а потому $H(Y \mid X) = H(Z \mid Y)$;

$$H(X,Y,Z) \leqslant H(X) + 2H(Y \mid X).$$

Определим теперь распределение $(\widetilde{X}, \widetilde{Y}, \widetilde{Z})$ на углах. Выберем вершину с той же вероятностью, с которой она является первой вершиной некоторого треугольника, обозначим её через x. Выберем равновероятно какой-то треугольник с вершиной x, проведём ребро и обозначим вторую вершину через y. Потом ещё раз независимо от значения y выберем треугольник и проведём ребро в z. Посчитаем энтропию:

$$H(\widetilde{X}, \widetilde{Y}, \widetilde{Z}) = H(\widetilde{X}) + H(\widetilde{Y} \mid \widetilde{X}) + H(\widetilde{Z} \mid \widetilde{X}, \widetilde{Y}).$$

По определению распределений, при известном $\widetilde{X}$ величины $\widetilde{Y}$ и $\widetilde{Z}$ независимы, то есть $H(\widetilde{Z} \mid \widetilde{X}, \widetilde{Y}) = H(\widetilde{Z} \mid \widetilde{X})$. Поскольку $\widetilde{Y}$ и $\widetilde{Z}$ выбираются одинаковым образом, $H(\widetilde{Y} \mid \widetilde{X}) = H(\widetilde{Z} \mid \widetilde{X})$. Таким образом,

$$H(\widetilde{X}, \widetilde{Y}, \widetilde{Z}) = H(\widetilde{X}) + 2H(\widetilde{Y} \mid \widetilde{X}).$$

Осталось заметить, что $H(X) = H(\widetilde{X})$, так как первая вершина в обоих распределениях выбирается одинаковым образом, и $H(Y \mid X) = H(\widetilde{Y} \mid \widetilde{X})$ по аналогичным причинам. Значит, энтропия некоторого распределения на углах $\geqslant \log|\triangle|$, то есть углов не меньше, чем треугольников. ■

2 Кодирование

Определение. Будем называть *кодом* функцию $C\colon \{a_1, \ldots, a_n\} \to \{0,1\}^*$, сопоставляющую буквам некоторого алфавита кодовые слова. Если любое сообщение, которое получено применением кода C, декодируется однозначно (то есть единственным образом разрезается на образы C), то такой код будем называть *однозначно декодируемым*.

2.1 Однозначно декодируемые и префиксные коды

Довольно удобно иметь более сильное свойство кода, чем однозначная декодируемость, которое позволяет декодировать сообщение отдельно по буквам.

Определение. Код называется *префиксным* (*беспрефиксным*, *prefix-free*), если никакое кодовое слово не является префиксом другого кодового слова.

Давайте попробуем понять, что любой однозначно декодируемый код можно переделать в беспрефиксный. Для этого мы попробуем описать критерии существования кодов.

Теорема 2.1. Пусть набор целых чисел $l_1, \ldots, l_n$ удовлетворяет неравенству

$$\sum_{i=1}^{n} 2^{-l_i} \leqslant 1.$$

Тогда существует префиксный код с кодовыми словами $c_1, \ldots, c_n$, где $|c_i| = l_i$.

Доказательство. Докажем эту теорему индукцией по n. Для одного слова утверждение задачи очевидно. Пусть даны числа $l_1, l_2, \ldots, l_{n+1}$.

Предположим, что среди этих чисел существуют два равных, скажем, $l_1 = l_2$. Тогда заменим числа l_1 и l_2 на одно число $l = l_1 - 1$. Мы получим n чисел: $l, l_3, l_4, \ldots, l_{n+1}$, для которых можно применить предположение индукции и найти префиксный код со словами $c, c_3, c_4, \ldots, c_{n+1}$. Осталось заметить, что если мы сопоставим числам l_1 и l_2 слова $c0$ и $c1$, то мы получим префиксный код, и условие теоремы будет выполнено.

Если же все числа различны, то можно выбрать слова длины $l_1, l_2, \ldots, l_n$ из слов вида $0, 10, 110, \ldots$ Понятно, что при этом мы получим префиксный код. ■

Если мы покажем, что для любого однозначно декодируемого кода выполнено неравенство (2.1), то вместе с теоремой 2.1 это позволит переделывать однозначно декодируемые коды в префиксные.

Утверждение 2.2 (неравенство Крафта–Макмиллана). Для любого однозначно декодируемого кода с кодовыми словами $c_1, c_2, \ldots, c_n$ выполнено неравенство

$$\sum_{i=1}^{n} 2^{-|c_i|} \leqslant 1. \tag{2.1}$$

Доказательство. Доказательство этой теоремы должно использовать однозначную декодируемость «в полном объёме», то есть для любой длины декодируемых сообщений. Формально заменим в каждом c_i нули на буквы x, а единицы на буквы y, где x и y не коммутируют. Пусть $p_i(x, y)$ — моном, соответствующий c_i, (например, коду $c_i = 010$ соответствует моном $p_i = xyx$); L — большое натуральное число. Рассмотрим

$$P^L(x, y) = \left(\sum_i p_i(x, y) \right)^L \leqslant \sum_{i=L}^{L \cdot \max(|c_i|)} M_i(x, y),$$

где M_i — это сумма всевозможных мономов степени i. Неравенство выполнено, так как код однозначно декодируемый, то есть каждый моном в левой части есть и в правой. Полагая $x = y = \frac{1}{2}$, получаем, что

$$P^L\left(\frac{1}{2}, \frac{1}{2}\right) \leqslant \sum_{i=L}^{L\cdot\max(|c_i|)} (2^i \cdot 2^{-i}) \leqslant O(L).$$

Предположим, что неравенство Крафта – Макмиллана не выполнено для данного кода. Тогда

$$\sum_i p_i\left(\frac{1}{2}, \frac{1}{2}\right) = \sum 2^{-|c_i|} = 1 + \varepsilon > 1.$$

Значит, $P^L = (1+\varepsilon)^L > O(L)$, что противоречит предыдущему рассуждению о линейности роста. ■

Теорема 2.3 (Шеннон). Для любого распределения p и однозначно декодируемого кода выполнено неравенство

$$\sum_i p_i|c_i| \geqslant h(p),$$

где p_i — частота, с которой встречается буква i, а c_i — её код.

Доказательство. По неравенству Йенсена и неравенству Крафта–Макмиллана,

$$h(p) - \sum_i p_i|c_i| = \sum_i p_i \cdot \log\frac{2^{-|c_i|}}{p_i} \leqslant \log\left(\sum_i p_i \cdot \frac{2^{-|c_i|}}{p_i}\right) \leqslant 0,$$

что и требовалось. ■

2.2 «Почти оптимальные» коды

Теорема 2.4. Для любого распределения p существует такой префиксный код, что

$$\sum_i p_i|c_i| \leqslant h(p) + 1.$$

Доказательство. Пусть $|c_i| = \lceil \log\frac{1}{p_i} \rceil$. Очевидно, что выполнено неравенство из условия, так как $p_i|c_i| \leqslant p_i \log\frac{1}{p_i} + p_i$, и $\sum p_i = 1$. Кроме того,

$$\sum_i 2^{-|c_i|} = \sum_i 2^{-\lceil \log\frac{1}{p_i} \rceil} \leqslant \sum_i p_i = 1.$$

По теореме 2.1 существует префиксный код, удовлетворяющий этому неравенству. Этот код и будет удовлетворять условию теоремы. ■

Примеры.

1. *Код Шеннона–Фано*. Отсортируем вероятности, $p_1 \geqslant p_2 \geqslant \ldots \geqslant p_n$. «Уложим»

вероятности p_i в отрезок $[0, 1]$, получая таким образом точки

$$0 \leqslant p_1 < p_1 + p_2 < \cdots < p_1 + p_2 + \cdots + p_n \leqslant 1.$$

Разобьём интервал пополам, и скажем, что все коды, отвечающие точкам слева от разреза, начинаются с нуля, а точкам справа — с единицы. Если отрезок пересекает разрез, и он самый левый (первый), то соответствующий код начинается с 0; если отрезок пересекает разрез и он самый правый (последний), то код начинается с 1. Иначе выбираем ноль или единицу произвольным образом. Продолжаем рекурсивно этот процесс, пока в интервале не останется ровно один отрезок.

2. *Код Хаффмана*. Код Хаффмана строится индуктивно. При $n = 2$ кодовые слова — $c_1 = 0, c_2 = 1$. При $n > 2$ рассмотрим два символа a и b с минимальными вероятностями p_{n-1} и p_n. Заменим указанные символы на новый символ σ, и дадим ему вероятность $p = p_{n-1} + p_n$. Построим код Хаффмана для $n - 1$ символа, и обозначим код символа σ за c, после чего скажем, что код символа a — это $c0$, а код символа b — $c1$.

Теорема 2.5 (Хаффман). Для кода Хаффмана выполнено неравенство

$$\sum_i p_i |c_i| \leqslant h(p) + 1, \tag{2.2}$$

и для любого другого однозначно декодируемого кода c'_i выполнено неравенство

$$\sum p_i |c'_i| \geqslant \sum p_i |c_i|. \tag{2.3}$$

Доказательство. Докажем неравенство (2.3), тогда из него и теоремы 2.4 будет следовать неравенство (2.2).

Предположим, что есть некоторый префиксный код, для которого (2.3) нарушено. Рассмотрим такой код с минимальным числом символов и минимальной средней длиной. Посмотрим на два символа с самым длинным кодом. Утверждается, что они имеют самые маленькие вероятности. Действительно, если бы это было не так, то коды символа с большей вероятностью и меньшей можно было бы поменять местами, при этом средняя длина кода от этого бы только уменьшилась. Можно считать, что коды этих двух символов равны $v0$ и $v1$ соответственно, иначе можно перекодировать слова не увеличивая среднюю длину так, чтобы это было верно. Тогда можно «склеить» эти два символа, как в коде Хаффмана. Получившийся код мы можем переделать в код Хаффмана так, чтобы средняя длина не увеличилась. Это можно сделать, так как мы брали код с минимальным числом символов, для которого нарушается неравенство, а символов стало меньше. Осталось заметить, что если «расклеить» символы, то мы получим в точности код Хаффмана, так как в нём мы делали то же самое первое действие (склеивали вершины с минимальной вероятностью). Отсюда следует, что наше предположение неверно, а значит неравенство (2.3) выполнено всегда. ■

3. *Арифметическое кодирование.* Назовём стандартным интервалом интервал вида $[0.\upsilon 0, 0.\upsilon 1)$, где υ — некоторая последовательность битов. Уложим вероятности p_i в отрезок $[0,1]$, получатся точки

$$0 \leqslant p_1 < p_1 + p_2 < \cdots < p_1 + p_2 + \cdots + p_n \leqslant 1.$$

Пусть $(0.\upsilon_i 0, 0.\upsilon_i 1)$ — максимальный стандартный интервал в отрезке

$$[p_1 + p_2 + \cdots + p_{i-1}, p_1 + p_2 + \cdots + p_i].$$

Тогда сопоставим i-ой букве код $\upsilon_i 0$. Легко заметить, что код получился префиксным, так как если υ_i является префиксом υ_j, то интервал $(0.\upsilon_j 0, 0.\upsilon_j 1)$ вложен в интервал $(0.\upsilon_i 0, 0.\upsilon_i 1)$, а такого при построении υ_i не может произойти.

Лемма 2.6. В отрезке $[a,b]$ длина наибольшего стандартного интервала не меньше, чем $\frac{b-a}{8}$.

Доказательство. Пусть $2^{-k-1} < \frac{b-a}{8} < 2^{-k}$. Рассмотрим все стандартные интервалы длины 2^{-k} . Заметим, что соседние интервалы находятся на расстоянии 2^{-k+1}. Предположим, что ни один стандартный интервал не попал полностью в отрезок $[a,b]$. Тогда длина отрезка $[a,b]$ не более, чем сумма длин двух стандартных отрезков плюс расстояние между ними, то есть $2^{-k} + 2^{-k} + 2^{-k+1} = 2^{-k+2}$. Но $2^{-k+2} < b - a$, противоречие. Значит, в отрезке $[a,b]$ найдётся стандартный интервал длины 2^{-k}, что доказывает лемму. ■

Утверждение 2.7. Для арифметического кода выполняется неравенство

$$\sum_i p_i |\upsilon_i| \leqslant h(p) + 2.$$

Доказательство. Из леммы 2.6 следует, что если $|\upsilon_i| = k$, то

$$0.\upsilon_i 1 - 0.\upsilon_i 0 = 2^{-k-1} \geqslant \frac{p_i}{8}.$$

Отсюда следует, что $k + 1 \leqslant \log \frac{8}{p_i}$, а значит $|\upsilon_i| = k \leqslant \log \frac{1}{p_i} + 2$, и

$$\sum_i p_i |\upsilon_i| \leqslant h(p) + 2(p_1 + p_2 + \cdots + p_n) \leqslant h(p) + 2.$$ ■

2.3 Кодирование с ошибками

Пусть $p_1, \ldots, p_k$ — вероятности, с которыми встречаются буквы в алфавите. Будем рассматривать слова фиксированной длины n, которые будут кодироваться в слова заданной длины L_n. Пусть нам даны функции кодирования и декодирования:

$$E\colon [k]^n \to \{0,1\}^{L_n}, \qquad D\colon \{0,1\}^{L_n} \to [k]^n.$$

При этом мы отказываемся от условия, что код декодируется однозначно, но требуем, чтобы вероятность $\varepsilon_n = \Pr[D(E(w)) \neq w]$ стремилась к нулю при $n \to \infty$.

Теорема 2.8 (Шеннон).

1. Если $L_n = \lceil h \cdot n \rceil$, где $h > h(p)$, то существуют такие функции E, D, что $\varepsilon_n \to 0$.
2. Если $L_n = \lceil h \cdot n \rceil$, где $h < h(p)$, то для любых E, D последовательность ε_n стремится к единице.

Доказательство. Пусть w — слово длины n. Будем говорить, что буква i *δ-типична*, если $|n_i/n - p_i| \leqslant \delta$ где n_i — количество букв i в w. Соответственно, w называется *δ-типичным*, если это неравенство выполняется для всех букв i. Зафиксируем $\delta := n^{-0.49}$ и рассмотрим случайную величину X_{ij} — характеристическую функцию того, что в позиции j находится буква i. Тогда для случайной величины $X_i := \sum_j X_{ij}$ мы можем написать неравенство Чебышёва:

$$\Pr[|X_i - \mu| \geqslant \delta n] \leqslant \frac{\mathbb{D}X_i}{(\delta n)^2} = \frac{np_i(1-p_i)}{(\delta n)^2} = O(n^{-0.02}),$$

где $\mu := \mathbb{E}[X_i] = np_i$. Таким образом, доля слов, в которых буква i нетипична стремится к нулю, а поскольку число букв фиксировано, мы можем заключить, что и в целом доля нетипичных слов стремится к нулю.

Число слов с заданным количеством вхождений каждой буквы равно

$$N = \frac{n!}{n_1!n_2!\dots n_k!} = h(p) \cdot n + o(n),$$

где $n_i = p_i n$.[1]

Известно, что $n! = \mathrm{poly}(n) \cdot (n/e)^n$. Тогда

$$\log N = \log\left(\left(\frac{n}{n_1}\right)^{n_1}\left(\frac{n}{n_2}\right)^{n_2}\dots\left(\frac{n}{n_k}\right)^{n_k}\right).$$

Так как это типичное слово, то $|n_i - np_i| \leqslant \delta n$. Тогда

$$\log\left(\left(\frac{n}{n_1}\right)^{n_1}\left(\frac{n}{n_2}\right)^{n_2}\dots\left(\frac{n}{n_k}\right)^{n_k}\right) \leqslant \sum n(p_i+\delta_i)\log\frac{1}{p_i+\delta_i} + O(\log n)$$
$$\leqslant nh(p) + O(\delta_i)n < h \cdot n,$$

где $n_i = (p_i + \delta_i)n$. Значит, количество типичных слов не превосходит

$$2^{nh(p)+O(\delta)n} \cdot 2^{O(h(\delta)n\cdot k)} < 2^{h\cdot n}.$$

А так как доля нетипичных слов стремится к нулю, то можно кодировать только типичные слова. Откуда следует, что если $h > h(p)$, то можно закодировать слова с ошибкой, стремящейся к нулю, при n стремящемся, к бесконечности.

Теперь перейдем к доказательству второго случая: $h < h(p)$. Пусть ε'_n вероятность ошибки при декодировании δ-типичных слов. Нам достаточно показать, что $\varepsilon'_n \to 1$,

[1] Это тождество было на практике.

поскольку $|\varepsilon'_n - \varepsilon|$ не превосходит вероятности того, что слово нетипично, т.е. не более $O(n^{-0.02})$.

Давайте рассмотрим конкретное δ-типичное слово w. Оценим вероятность появления w:

$$\Pr[w] = p_1^{n_1} \cdot \dots \cdot p_k^{n_k} = 2^{-\sum n_i \log \frac{1}{p_i}} = 2^{-\sum (p_i+\delta_i) \log \frac{1}{p_i} n}.$$

Всего мы можем корректно закодировать не более 2^{L_n} различных δ-типичных слов, то есть вероятность корректно декодировать δ-типичное слово

$$1 - \varepsilon'_n \leqslant 2^{L_n} 2^{-h(\alpha)n + O(\delta n)} \leqslant 2^{hn - h(\alpha)n + O(\delta n)} \to 0. \qquad \blacksquare$$

3 Криптография

3.1 Шифрование и схемы разделения секрета

Определение. *Взаимной информацией* между случайными величинами α и β будем называть число

$$I(\alpha : \beta) := h(\alpha) - h(\alpha \mid \beta).$$

Также определим взаимную информацию в α и β при условии γ:

$$I(\alpha : \beta \mid \gamma) := h(\alpha \mid \gamma) - h(\alpha \mid \beta, \gamma).$$

Лемма 3.1. Взаимная информация обладает следующими свойствами:

(1) $I(\alpha : \beta) = I(\beta : \alpha)$;

(2) α и β независимы тогда и только тогда, когда $I(\alpha : \beta) = 0$;

(3) $I(f(\alpha) : \beta) \leqslant I(\alpha : \beta)$ для любой функции f.

(4) выполнены равенства

$$\begin{aligned} I(\alpha : \beta) &= h(\alpha) - h(\alpha \mid \beta) \\ &= h(\beta) - h(\beta \mid \alpha) \\ &= h(\alpha) + h(\beta) - h(\alpha, \beta) \\ &= h(\alpha, \beta) - h(\alpha \mid \beta) - h(\beta \mid \alpha). \end{aligned}$$

Доказательство. Упражнение. $\blacksquare$

Рассмотрим следующую схему передачи слова от Алисы к Бобу: Алиса получает на вход слово w. С помощью ключа k она кодирует слово w и отправляет получившееся слово $c = E(w, k)$ Бобу. Боб декодирует полученное слово с помощью ключа k и получает первоначальное слово $D(c, k) = w$. Может случиться так, что злоумышленник перехватил сообщение и получил слово c. Чтобы он не смог расшифровать его,

должны выполняться следующие условия:

$$\begin{cases} H(c \mid w, k) = 0, \\ H(w \mid c, k) = 0, \\ I(c : w) = 0. \end{cases}$$

Действительно: зная w и k мы можем узнать c, зная c и k мы можем вычислить w, а также злоумышленник не должен ничего понять по полученному слову c. Если выполнены все три условия, то назовём эту схему *идеальной*.

Рассмотрим пример идеальной схемы. Пусть $E(w, k) = w \oplus k$, где слово w выбирается из множества слов длины n, k — это ключ, известный заранее, $|k| = n$. Очевидно, что для этого кодирования выполнены первые 2 условия. Также несложно проверить третье условие.

Теорема 3.2 (Шеннон). Для идеальной схемы выполнены следующие условия:

$$H(w) = H(w \mid c), \qquad H(k) \geqslant H(w).$$

Доказательство. Первое условие выполнено, так как

$$0 = I(c : w) = I(w : c) = H(w) - H(w \mid c).$$

Второе условие следует из неравенств:

$$H(w) = H(w \mid c) \leqslant H(w, k \mid c) = H(k \mid c) + H(w \mid c, k) = H(k \mid c) \leqslant H(k). \qquad \blacksquare$$

Определение. Пусть $(S_0, S_1, \ldots, S_n)$ — набор булевых слов. Будем называть его *совершенной схемой разделения секрета*, если:

(1) $H(S_0 \mid S_1, \ldots, S_n) = 0$;

(2) $H(S_0 \mid S_{i_1}, \ldots, S_{i_{n-1}}) = H(S_0)$ для любого набора индексов $i_1 < i_2 < \cdots < i_{n-1}$.

Можно понимать это так: есть n игроков, и каждый из них получил некоторое булево слово S_i. Собравшись вместе, они могут отгадать секрет S_0, но если хотя бы одного игрока не будет, то секрет отгадать не получится.

Пример 3.1. Пусть $S_0 = \bigoplus_{i=1}^{n} S_i$, где S_i — случайное бинарное слово. Очевидно, что S_0 определяется по остальным S_i, однако без любого S_i определить его будет уже невозможно.

Пример 3.2. Пусть $S_0 \in \{0,1\}^l$, $S_1, \ldots, S_{n-1}$ — это случайные величины, которые равномерно распределены на $\{0,1\}^l$, и $S_n = \bigoplus_{i=0}^{n-1} S_i$. Заметим, что S_0 определяется однозначно по остальным S_i, однако для любой перестановки σ на $[n]$ элементах

$$\Pr[S_0 = a \mid S_{\sigma(1)}, \ldots, S_{\sigma(n-1)}] = \Pr[S_0 = a],$$

и, следовательно, $H(S_0 \mid S_{\sigma(1)}, \ldots, S_{\sigma(n-1)}) = H(S_0)$.

Теорема 3.3. Если участник i является «существенным» в структуре доступа Γ (то есть существует такое $s \in \Gamma$, что $s \setminus \{i\} \in \Gamma$), то $H(S_i) \geqslant H(S_0)$.

Доказательство. Не было на лекциях[2]. ■

Также можно рассматривать *пороговую совершенную схему разделения секрета*:

(1) $H(S_0 \mid S_{i_1}, \ldots, S_{i_k}) = 0$;

(2) $H(S_0 \mid S_{i_1}, \ldots, S_{i_{k-1}}) = H(S_0)$;

для некоторого фиксированного k.

***Теорема 3.4.** Существует такая схема разделения секрета, что для некоторого $1 \leqslant i \leqslant n$ выполнено неравенство $H(S_i) \geqslant \frac{n}{\log(n)} H(S_0)$.

3.2 Схема Шамира

Пусть $S_0 \in \mathbb{F}_q$, $x_1, \ldots, x_n \in \mathbb{F}_q$, $x_i \neq x_j$, где $\mathbb{F}_q$ — конечное поле. Посмотрим на многочлен

$$F(x) = \sum_{i=1}^{t-1} a_i x^i + S_0.$$

Пусть i-ый игрок получает число $S_i = F(x_i)$. Тогда понятно, что любые t игроков смогут интерполировать многочлен и узнать S_0. Докажем, что $t-1$ игроков не смогут определить S_0. Пусть мы знаем

$$S_1 = F(x_1), \quad S_2 = F(x_2), \quad \ldots, \quad S_{t-1} = F(x_{t-1}).$$

Это даёт нам линейную систему уравнений на коэффициенты полинома F с $t-1$ уравнением и t неизвестными. Заметим, что

$$\Pr[S_0 = S, a_i = x_i \mid S_1, S_2, \ldots, S_{t-1}] = \Pr[a_i = x_i \mid S_1, S_2, \ldots, S_{t-1}],$$

так как зная один коэффициент многочлена степени t и его значения в $t-1$-ой точке, мы однозначно восстанавливаем многочлен. Значит, $t-1$ игроков не смогут найти S_0.

3.3 Идеальные схемы разделения секрета

Определение. *Идеальная схема разделения секрета* — это совершенная схема разделения секрета с дополнительным требованием «экономности»:

$$H(S_i) \leqslant H(S_0) \qquad \forall i : 1 \leqslant i \leqslant n.$$

Утверждение 3.5. Существуют структуры доступа, для которых не существует идеальной схемы разделения секрета.

[2] Я так и не понял, надо ли её рассказывать в билете. (ОМ)

Доказательство. Рассмотрим задачу разделения секрета для следующей структуры доступа с 4 участниками: минимальными группами участников, знающих секрет, являются три пары:

$$\{1,2\}, \quad \{2,3\}, \quad \{3,4\}.$$

Докажем, что существует такое i, что $H(S_i) \geqslant \frac{3}{2}H(S_0)$. Для доказательства нам потребуются 3 вспомогательных неравенства.

(1) $H(S_2 \mid S_1, S_3) \geqslant H(S_0)$.

Второй участник может восстановить секрет, воспользовавшись либо секретом первого, либо секретом третьего участника, то есть $I(S_2 : S_0 \mid S_1, S_3) \geqslant H(S_0)$. По условию, $H(S_0 \mid S_1, S_2, S_3) = 0$. Значит,

$$H(S_2 \mid S_1, S_3) \geqslant I(S_2 : S_0 \mid S_1, S_3) \geqslant H(S_0).$$

(2) $H(S_3 \mid S_1) \geqslant H(S_0)$.

Аналогично предыдущему пункту получаем, что $H(S_3 \mid S_1, S_4) \geqslant H(S_0)$. Тогда

$$H(S_3 \mid S_1) \geqslant H(S_3 \mid S_1, S_4) \geqslant H(S_0).$$

(3) $I(S_1 : S_3 \mid S_2) \geqslant H(S_0)$.

Заметим, что $I(S_1 : S_0 \mid S_2) = H(S_0)$ и $I(S_3 : S_0 \mid S_2) = 0$. При этом

$$I(S_1 : S_0 \mid S_2, S_3) = 0 \quad \text{и} \quad I(S_3 : S_0 \mid S_2, S_1) = 0.$$

Таким образом, $I(S_1 : S_3 : S_0 \mid S_2) = H(S_0)$, а значит $H(S_1 : S_3 \mid S_2 \geqslant H(S_0)$.

Осталось сложить эти неравенства:

$$\begin{aligned} H(S_2) + H(S_3) &\geqslant H(S_1, S_2) \\ &\geqslant H(S_2 \mid S_1, S_3) + H(S_3 \mid S_1) + I(S_1 : S_3 \mid S_2) + I(S_2 : S_1) \\ &\geqslant 3H(S_0). \end{aligned}$$

■

4 Коммуникационная сложность

Определение. *Коммуникационный протокол* для функции $f : X \times Y \to Z$ — это корневое двоичное дерево, которое описывает совместное вычисление Алисой и Бобом функции f. В этом дереве каждая внутренняя вершина v помечена меткой a или b, означающей очередь хода Алисы или Боба соответственно. Для каждой вершины, помеченной a, определена функция $g_v : X \to \{0,1\}$, которая говорит Алисе, какой бит нужно послать, если вычисление находится в этой вершине. Аналогично, для каждой вершины v с пометкой b, определена функция $h_v : Y \to \{0,1\}$, которая определяет бит, который Боб должен отослать этой вершине. Каждая внутренняя вершина имеет двух потомков, ребро к первому потомку помечено нулём, а ребро ко второму —

единицей. Каждый лист помечен значением из множества Z. Таким образом, каждая пара входов (x, y) определяет путь от корня до листа в описанном двоичном дереве естественным образом. Будем говорить, что коммуникационный протокол *вычисляет* функцию f, если для всех пар $(x, y) \in X \times Y$ этот путь заканчивается в листе с пометкой $f(x, y)$.

Коммуникационной сложностью функции f называется наименьшая глубина протокола, вычисляющего функцию f. Будем обозначать её символом $D(f)$.

Каждой функции f будем сопоставлять матрицу $X \times Y$, в которой в клетке (x_i, y_j) стоит значение $f(x_i, y_j)$.

4.1 Прямоугольники и метод *Fooling Set*

Утверждение 4.1. Рассмотрим дерево протокола со входом из множества $X \times Y$. Рассмотрим в нём произвольную вершину u. Тогда все входы, из которых можно прийти в вершину u, образуют прямоугольник $R_u = X_u \times Y_u \subseteq X \times Y$.

Доказательство. Это можно доказать двумя способами.

Первый способ: пусть на входах (x_1, y_1) и (x_2, y_2) мы приходим в вершину u. Тогда нетрудно убедиться, что на входе (x_1, y_2) Алиса и Боб будут делать те же действия, что и на входах (x_1, y_1) и (x_2, y_2) соответственно. Отсюда видно, что входы, приводящие в вершину u, образуют прямоугольник $R_u = X_u \times Y_u \subseteq X \times Y$.

Второй способ: Рассмотрим таблицу элементов $X \times Y$. После первого хода Боба табличка делится пополам горизонтальной линией, так как при одних $x \in X$ Боб посылает Алисе 1, а при других — 0. Потом Алиса посылает свой бит Бобу, и каждый из двух получившихся прямоугольников делится своей вертикальной прямой, и так далее. В итоге мы получим разбиение $X \times Y$ на непересекающиеся прямоугольники, и каждый из этих прямоугольников соответствует листу в коммуникационном протоколе. ■

Про прямоугольник R_u можно думать в следующим образом: если мы находимся в вершине протокола u, то нам необходимо решить задачу (то есть построить протокол) для всех входов из прямоугольника R_u. В частности этот подход можно рассмотреть, как комбинаторное определение протокола: бинарное дерево, в котором каждой вершине сопоставлен прямоугольник входов. И если вершины a, b являются потомками u, то $R_u \subseteq R_a \cup R_b$.

Рассмотрим величину $\chi_0(f)$, равную минимальному числу прямоугольников, которыми можно дизъюнктно покрыть нули в таблице. Аналогично определяется $\chi_1(f)$. Тогда листьев в коммуникационном протоколе будет хотя бы $\chi_0(f) + \chi_1(f)$. Эти рассуждения дают следующую оценку:

$$D(f) \geqslant \log(\chi_0(f) + \chi_1(f)).$$

Эта оценка не всегда точна. Это нам даёт понять следующий пример:

Пример 4.1. Рассмотрим такой пример разбиения таблицы $X \times Y$ на прямоугольники: в центре находится прямоугольник из 1, а вокруг него расположены 4 прямо-

угольника из 0. Покажем, что для этого разбиения не существует дерева протокола. Действительно, рассмотрим первое действие игроков. После него таблица должна поделиться на две части, но на рисунке 2 видно, что нет разреза, проходящего через всю таблицу. Мы получили, что $\chi_0(f) + \chi_1(f) = 5$, хотя коммуникационного протокола с пятью листьями не существует.

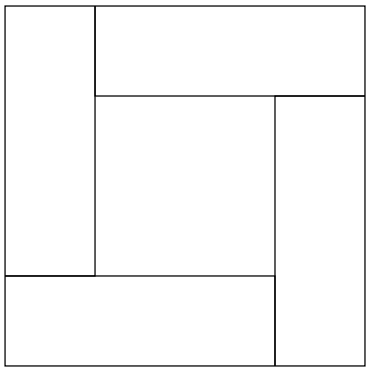


Рис. 2

Пусть дана таблица некоторой функции со значениями из $\{0, 1\}$. Выберем клетки этой таблицы $a_1, a_2, \ldots, a_n$ так, чтобы никакие две из них не могли оказаться в одном «одноцветном» прямоугольнике, состоящим только из нулей или только из единиц. Тогда для каждой клетки должен быть свой прямоугольник разбиения. Отсюда следует, что в дереве протокола должно быть хотя бы n вершин, а высота дерева хотя бы $\log n$. Этот метод называется *Fooling Set*.

Например, его можно применить для функции

$$\mathrm{EQ}(x, y) := \begin{cases} 1, & x = y, \\ 0, & x \neq y, \end{cases}$$

на строчках длины m мы получим диагональную матрицу $2^m \times 2^m$, и тогда в качестве точек a_i можно выбрать единицы на диагональной матрице. Понятно, что никакие две из них не находятся в одном одноцветном прямоугольнике. Значит, высота коммуникационного дерева $\geqslant \log(2^m) = m$.

Пример 4.2. Рассмотрим функцию $\mathrm{DISJ}(x, y)$ на подмножествах множества $[n]$, которая принимает значение 1 тогда и только тогда, когда $x \cap y = \varnothing$. Для всех множеств $S \in [2^n]$ рассмотрим ячейку матрицы $a_S := (S, \overline{S})$. Заметим, что для $S \neq S'$ ячейки a_S и $a_{S'}$ не могут лежать в одном прямоугольнике, так как иначе в этом прямоугольнике лежат ячейки $(S, \overline{S'})$, $(S', \overline{S})$, и в одной из них стоит 0. Тогда можно применить *Fooling set* для всех ячеек a_S и получить, что высота коммуникационного дерева не меньше $\log(2^n) = n$.

4.2 Метод ранга

Пусть A — таблица некоторой функции со значениями из $\{0, 1\}$. Пусть $x_1, \ldots, x_k$ — листы коммуникационного протокола для функции A, а $R_{x_1}, \ldots, R_{x_n}$ — соответ-

ствующие им прямоугольники в A. Из алгебры мы знаем, что $\operatorname{rank}(A) \leqslant \sum \operatorname{rank}(R_i)$, а $\operatorname{rank}(R_i) = 1$ для любого i. Отсюда можно сделать вывод, что количество листьев в протоколе не меньше $\operatorname{rank} A$, а коммуникационная сложность — не меньше $\log \operatorname{rank}(A)$. Этот метод называется *методом ранга*. Оценим с помощью этого метода коммуникационную сложность некоторых функций.

Утверждение 4.2. Коммуникационная сложность функции $\mathrm{EQ}(x, y)$, действующей из $\{1, 2, \ldots, 2^n\} \times \{1, 2, \ldots, 2^n\}$ в $\{0, 1\}$, равна $n + 1$.

Доказательство. Нетрудно видеть, что ранг матрицы функции EQ равен 2^n, а значит $D(\mathrm{EQ}) \geqslant \log 2^n = n$. Очевидно также, что коммуникационная сложность не превосходит $n + 1$, так как за $n + 1$ вопросов можно проверить равенство всех битов. ■

Утверждение 4.3. Коммуникационная сложность функции $\mathrm{GT}(x, y) = (x \overset{?}{>} y)$, действующей из $\{1, 2, \ldots, 2^n\} \times \{1, 2, \ldots, 2^n\}$ в $\{0, 1\}$, равна n.

Доказательство. Доказательство практически аналогично предыдущему — ранг матрицы для функции GT равен 2^n, то есть $D(\mathrm{GT}) \geqslant \log 2^n = n$, и $D(\mathrm{GT}) \leqslant n$, так как за n вопросов можно найти первый бит, в котором различаются x и y, и определить, какое из этих чисел больше. ■

4.3 Балансировка протоколов

Теорема 4.4. Пусть для функции f существует коммуникационный протокол P с l листьями. Тогда $D(f) \leqslant 3 \log l$.

Доказательство. Мы воспользуемся несложной комбинаторной леммой:

Лемма 4.5. В двоичном дереве с l листьями можно найти внутреннюю вершину u, которая разбивает граф на три части, каждая из которых содержит не более $l/2$ листьев.[3]

Доказательство. Каждой внутренней вершине u дерева соответствует поддерево с некоторым числом листьев (обозначим его через l_u). Выберем самую низкую (самую далекую от корня) вершину u, для которой выполнено неравенство $l_u > l/2$. Обозначим «сыновей» u через v и w. По выбору u поддеревья для v и w содержат не более $l/2$ листьев. При этом число листьев, не лежащих ниже u, будет меньше $l - l_u < l/2$. Таким образом, u — искомая вершина. ■

Теперь опишем новый протокол P_0 для вычисления f. Прежде всего, зафиксируем вершину u, которая делит исходный протокол на три части, в каждой из которых $\leqslant l/2$ листьев. Для определённости будем считать, что u соответствует ходу Алисы.

В новом протоколе Алиса и Боб сначала сообщают друг другу, согласованы ли их входы x и y с путём из корня в вершину u в исходном протоколе. Если оказывается,

[3] Одна из трёх частей может быть пустой — в некоторых случаях корень разбивает дерево на две части с ровно $l/2$ листьями в каждой.

что согласованы, то Алиса сообщает Бобу, следует ли при заданном (известном ей) значении x перейти к левому или к правому сыну u.

Таким образом, обменявшись 3 битами, Алиса и Боб уменьшают вдвое (по крайней мере до $l/2$) число потенциально возможных листьев в протоколе — потенциально достижимыми остаются либо все потомки левого сына u, либо потомки правого сына u, либо листья вне поддерева u.

Удалив из P все заведомо недостижимые листья, мы получаем новый протокол P_1, в котором остаётся $l_1 \leqslant l/2$ листьев. Далее можно снова повторить тот же приём: находим в P_1 вершину, которая разбивает протокол на три части с не более чем $l_1/2$ листьями, Алиса и Боб обмениваются 3 битами и выбирают из этих трёх частей одну, и т. д.

На каждой итерации данной конструкции Алиса и Боб обмениваются не более чем тремя битами, и число остающихся листьев сокращается как минимум вдвое. В итоге мы получаем новый протокол, сложность которого не больше $3 \log l$. ■

4.4 Теорема Карчмера–Вигдерсона

Рассмотрим функцию $f\colon \{0,1\}^n \to \{0,1\}$. Её можно представить в виде булевой формулы с операциями $\wedge$, $\vee$ и $\neg$. Каждую такую формулу удобно представлять в виде двоичного дерева, во внутренних вершинах которого стоят символы $\wedge$, $\vee$, в листьях — переменные x_i, а некоторые рёбра помечены символом $\neg$.

Для каждой такой функции f можно рассмотреть следующую коммуникационную задачу: Алиса получает число $x \in f^{-1}(1)$, Боб — $y \in f^{-1}(0)$, их цель — найти произвольную позицию i, в которой $x_i \neq y_i$, то есть некоторую вычислить функцию

$$\mathrm{KW}_f\colon f^{-1}(1) \times f^{-1}(0) \to [n].$$

Теорема 4.6 (Карчмер, Вигдерсон). Любую формулу для f можно переделать в коммуникационный протокол для KW_f с таким же деревом и наоборот. В частности, глубина формулы f равна её коммуникационной сложности.

Доказательство. Функцию, которую считает гейт формулы u, будем обозначать через f_u; прямоугольник, соответствующий вершине протокола v — через R_v.

(1) Построим по формуле коммуникационный протокол. Пусть Алиса получила на вход строку $x \in f^{-1}(1)$, а Боб — строку $y \in f^{-1}(0)$. Ход Алисы в протоколе будет соответствовать символу $\wedge$, ход Боба — символу $\vee$. Назовём гейт формулы u *хорошим*, если $f_u(x) \neq f_u(y)$.

Будем строить протокол, начиная с корня дерева формулы. По условию задачи, корень — это хорошая вершина, целью Алисы и Боба на каждом «раунде» будет являться поиск хорошего предка[4]. Тогда, перемещаясь на каждом раунде в такого предка, Алиса и Боб найдут хороший лист, в котором и будет тот вход формулы, на котором строки Алисы и Боба различаются. Таким образом, для

[4] Здесь подразумевается, что деревья растут снизу вверх. Если считать, что корень находится сверху, то осуществляется поиск потомков, а не предков. (ОМ)

того, чтобы завершить доказательство, нам достаточно показать, как найти хорошего предка, передав не более одного бита. Рассмотрим текущую вершину u с предками a, b (не умаляя общности можно считать, что $f_u(x) = 1$ и $f_u(y) = 0$). Далее возможны два случая.

- В вершине u написан значок $\wedge$. Тогда $f_a(x) = f_b(x) = 1$, и либо $f_a(y) = 0$, либо $f_b(y) = 0$. Таким образом, Боб может однозначно определить, какой из предков является хорошим, и сообщить это Алисе, передав один бит.
- В вершине u написан символ $\vee$. Тогда $f_a(y) = f_b(y) = 0$, и либо $f_a(x) = 1$, либо $f_b(x) = 1$. Значит, Алиса может однозначно определить, какой из предков является хорошим, и сообщить это Бобу, передав один бит.

(2) Построим по коммуникационному протоколу формулу. По индукции, начиная с листьев протокола, для каждой вершины протокола v мы предъявим формулу для такой функции f_v, что $f_v(X_v) = 1$ и $f_v(Y_v) = 0$, где $R_v := X_v \times Y_v$.

Рассмотрим лист протокола l и заметим, что так как прямоугольник $R_l := X_l \times Y_l$ одноцветный, то все строки $x \in X_l$ отличаются от всех строк $y \in Y_l$ в какой-то фиксированной позиции i_l. В частности, это означает, что выполнен один из двух следующих случаев:

- $x_{i_l} = 1$ и $y_{i_l} = 0$ для всех $x \in X_l, y \in Y_l$, и тогда мы можем определить $f_l := x_{i_l}$;
- $x_{i_l} = 0$ и $y_{i_l} = 1$ для всех $x \in X_l, y \in Y_l$, и тогда мы можем положить $f_l := \neg x_{i_l}$.

Других случаев не существует, поскольку если найдутся такие $x, x' \in X$, что $x_{i_l} \neq x'_{i_l}$, то они одновременно не могут отличаться в позиции i_l ни от какого y (случай $y_{i_l} \neq y'_{i_l}$ аналогичен).

Построим теперь формулу для функции f_v, если у нас уже есть формулы для функций f_a и f_b, где a, b — потомки вершины v. Заметим, что прямоугольники R_a и R_b получены рассечением прямоугольника R_v на две части либо вертикальным, либо горизонтальным сечением. Рассмотрим эти случай отдельно и заметим, что:

- если сечение было горизонтальным, то $Y_a = Y_b = Y_v$ и $X_v = X_a \cup X_b$, и в таком случае нам подойдет формула $f_v := f_a \vee f_b$;
- если сечение было вертикальным, то $X_a = X_b = X_v$ и $Y_v = Y_a \cup Y_b$, и в таком случае нам подойдет формула $f_v := f_a \wedge f_b$. ■

4.5 Сложные функции

Определение. *Формульной сложностью* $L(f)$ формулы f называется минимальное возможное число листьев дерева, вычисляющего эту формулу.

Теорема 4.7 (Шеннон). Существует такая функция $f\colon \{0,1\}^n \to \{0,1\}$, что

$$L(f) \geqslant \Omega\left(\frac{2^n}{n}\right).$$

Доказательство. Заметим, что количество функций $f\colon \{0,1\}^n \to \{0,1\}$ равно 2^{2^n}. Посмотрим на всевозможные деревья сложности не более, чем s. Каждое из них представляет собой двоичное дерево с не более, чем $2s-1$ вершинами, в которых написаны булевы операции, и не более, чем s листьями, в которых написаны переменные. Тогда в каждой вершине стоит один из $n+6$ символов (n переменных и 2 булевых символа с возможными отрицаниями перед ними), а количество деревьев с не более, чем $2s-1$ вершинами, по теореме Кэли не превосходит

$$(2s-1)^{2s-3}(2s-1) \leqslant (2s)^{2s} \leqslant 2^{4s\log s}.$$

Отсюда следует, что количество деревьев сложности не более, чем s, не превосходит $2^{4s\log(s)+2s\log(n+6)}$. Осталось убедиться, что при $s < \frac{2^n}{10n}$ количество деревьев сложности $\leqslant s$ строго меньше количества функций $f\colon \{0,1\}^n \to \{0,1\}$. Действительно, при $s < \frac{2^n}{10n}$ для больших n верно, что $2^{4s\log(s)+2s\log(n+6)} < 2^{2^n}$. Значит, существует формула, сложность которой не менее $\frac{2^n}{10n}$. ■

На данный момент известна «явная» функция f,[5] для которой выполнено неравенство $L(f) \geqslant n^{3-\varepsilon}$. Ничего лучше пока не придумано.

Определение. Функция $f\colon \{0,1\}^n \to \{0,1\}$ называется *монотонной*, если для любых u, v, таких, что каждый бит u не меньше соответствующего бита v, верно, что $f(u) \geqslant f(v)$.

Теорема 4.8. Существует такая монотонная функция $f\colon \{0,1\}^n \to \{0,1\}$, что

$$\log L(f) = D(\mathrm{KW}_f) \geqslant n - o(n).$$

Доказательство. Оценим количество монотонных функций. Рассмотрим такие функции g, что $g(u) = 0$, если в u меньше, чем $[n/2]$ единиц, и $g(u) = 1$, если в u больше, чем $[n/2]$ единиц, где $[x]$ — целая часть. На строках же, в которых ровно $[n/2]$ единицы, функция g определена как угодно. Нетрудно понять, что все функции такого вида монотонны, и их $2^{C_n^{[n/2]}}$. Значит, количество монотонных функций оценивается снизу числом $2^{C_n^{[n/2]}}$.

В доказательстве теоремы 4.7 показано, что количество функций сложности $\leqslant s$ не превосходит $2^{2s\log(s)+2s\log(n+6)}$. Тогда получаем неравенство:

$$2^{C_n^{[n/2]}} \leqslant 2^{2s\log(s)+2s\log(n+6)}$$

где s — максимальная сложность монотонной функции. Из формулы Стирлинга знаем, что $n! = O(\sqrt{2\pi n}(\frac{n}{e})^n)$. А значит $C_n^{[n/2]} = O(\frac{2^n}{\sqrt{n}})$. Откуда получаем, что $2s\log(s) + 2s\log(n+6) \geqslant O(2^n)$, а значит $\log s \geqslant n - o(n)$. ■

[5] то есть существует алгоритм, считающий её за полиномиальное время

4.6 Вероятностные коммуникационные протоколы

В вероятностных коммуникационных протоколах мы требуем, чтобы в начале работы протокола случайно выбиралась строка $r \in \{0,1\}^l$ фиксированной длины l, и далее на каждом шаге действия Алисы зависят от x и r, а действия Боба зависят от y и r.

Вероятностный протокол *с общими случайными битами* можно рассматривать как распределение вероятностей на детерминированных протоколах: Алиса и Боб сначала выбирают случайную строку r, а затем следуют детерминированному протоколу, соответствующему этой строке r.

Вероятность ошибки на входе (x, y) — это процент случайных слов r, при которых протокол выдаёт неправильный ответ. *Вероятностной коммуникационной сложностью $R_\varepsilon^{\text{pub}}(f)$ функции f с ошибкой ε* мы будем называть минимальную высоту коммуникационного протокола P, в котором вероятность ошибки для любого входа не превышает ε.

Утверждение 4.9. $R_\varepsilon^{\text{pub}}(\text{EQ}) = O(\log(\frac{1}{\varepsilon}))$.

Доказательство. Протокол устроен следующим образом. На входах $x, y \in \{0,1\}^n$ Алиса и Боб выбирают случайную строку $r \in \{0,1\}^n$, после чего Алиса вычисляет скалярное произведение $\langle x, r\rangle$ над полем из двух элементов и передаёт вычисленный бит Бобу. Боб вычисляет скалярное произведение $\langle y, r\rangle$, сравнивает со скалярным произведением Алисы и, если произведения равны, выдаёт 1, а иначе выдаёт 0.

Заметим, что если $x = y$, то протокол всегда выдает правильный ответ. Если же $x \neq y$, то протокол выдает 1 (то есть, ошибается), если $\langle x - y, r\rangle = 0$. Это линейное уравнение на r, и его решения образуют линейное пространство размерности $n-1$ над полем из двух элементов. Легко понять, что число векторов в таком пространстве равно 2^{n-1}, а значит ошибка происходит ровно на половине случайных наборов, то есть ошибка происходит с вероятностью $1/2$.

Если мы теперь повторим описанный протокол k раз с независимыми случайными битами и будем выдавать 0, если хотя бы один раз скалярные произведения оказались не равны, то в случае равных x и y ошибки по-прежнему не будет, а в случае различных входов ошибка происходит с вероятностью $1/2^k$. Таким образом, чтобы добиться ошибки ε, достаточно повторить протокол $\log 1/\varepsilon$ раз. ■

Теорема 4.10. $R_\varepsilon^{\text{pub}}(\text{GT}) = O(\log n \log\log n)$.

Доказательство. Оценим вероятностную коммуникационную сложность предиката GT. Для начала рассмотрим самый очевидный протокол. Чтобы выяснить, какое из чисел $x, y \in \{1, 2, \ldots, 2^n\}$ больше, Алисе и Бобу необходимо найти самый старший бит (самую левую позицию), в котором двоичные записи этих чисел различаются. Когда эта позиция найдена, Алисе и Бобу остаётся обменяться битами, которые стоят в их словах на этом месте.

Будем искать нужную позицию двоичным поиском, применяя тест на равенство (такой же, как и в предыдущей теореме) к блокам двоичных слов всё меньшего и меньшего размера. Поскольку на каждой итерации мы можем сужать область поиска вдвое, нам потребуется $\log n$ применений теста на равенство. Чтобы итоговая

вероятность ошибки не превосходила ε, каждый такой тест должен ошибаться с вероятностью не более $\delta = \varepsilon / \log n$.

Коммуникационная сложность одного теста на равенство с вероятностью ошибки δ не превосходит $O(\log 1/\delta)$. Суммируя число переданных битов в $\log n$ итерациях, получаем:

$$R_\varepsilon^{\text{pub}}(\text{GT}) = O(\log n \log\log n). \qquad \blacksquare$$

Замечание. Можно доказать, что на самом деле $R_\varepsilon^{\text{pub}}(\text{GT}) = O(\log n)$.

Указание: сначала рассмотрите протоколы с общим источником случайности. Как и в рассматриваемых ранее протоколах, примените двоичный поиск самого старшего бита, в котором двоичные записи чисел x и y различаются. В процессе двоичного поиска размер интервала, предположительно содержащего нужную нам позицию, будет постепенно сокращаться. Используйте тест на равенство с вероятностью ошибки $1/4$, который имеет коммуникационную сложность $O(1)$; периодически тестируйте корректность текущего интервала. При каждом обнаружении ошибки возвращайтесь к предыдущему значению левого края интервала поиска.

4.7 Вычисление функции индексирования

Начнём с урезанной коммуникационной модели. Определим функцию индексирования

$$\text{Ind}\colon [n] \times \{0,1\}^n \to \{0,1\}, \qquad \text{Ind}(x, y) = y_x,$$

где y_x — x-ый бит числа y. Мы можем рассмотреть задачу, в которой Алиса и Боб должны вычислить функцию Ind. Нетрудно показать, что коммуникационная сложность этой задачи равна $O(\log n)$, так как Алиса может просто послать Бобу число x, а Боб назвать бит, стоящий в x-ой позиции.

Усложним задачу. Пусть биты может посылать только Боб. Тогда коммуникационная сложность этой задачи становится $O(n)$, так как в итоге Боб должен послать информацию про все биты своего числа. Иначе у Алисы может быть бит x, про который она ничего не знает.

Будем теперь считать, что Алиса и Боб получают входы согласно равномерному распределению среди всех возможных входов. Посылать информацию может только Боб. Но теперь Бобу с Алисой разрешено делать ошибку $\varepsilon := \frac{1}{2} - \delta$, то есть можно ошибиться не более чем на $\varepsilon|X \times Y|$ входах. Докажем, что коммуникационная сложность этой задачи равна $\Omega(\delta^2 n)$.

Рассмотрим коммуникационный протокол π, решающий нашу задачу. Понятно, что $D^1(\pi) \geqslant \log|M|$, где M — множество листьев, а D^1 — коммуникационная сложность задачи, в которой только Боб может посылать биты. Рассмотрим энтропию $H(M)$ распределения на листьях, которая получается естественным способом из распределения на входах. Нетрудно получить следующие неравенства:

$$\log|M| \geqslant H(M) \geqslant I(M : y).$$

По *chain rule* мы получаем, что:

$$\begin{aligned}
I(M : y) &= \sum_i I(M : y_i \mid y_{x_{<i}}) \\
&= \sum_i H(y_i \mid y_{x_{<i}}) - H(y_i \mid M, y_{x_{<i}}) = [\text{так как все } y_i \text{ независимы}] \\
&= \sum_i H(y_i) - H(y_i \mid M, y_{x_{<i}}) \\
&\geqslant \sum_i H(y_i) - H(y_i \mid M) \\
&\geqslant \sum_i I(M : y_i).
\end{aligned}$$

Для оценки $I(M : y_i)$ нам потребуется вспомогательная величина. Пусть r_i^m — вероятность ошибки, при условии того, что Боб послал сообщение m и $x = i$. Из определения r_i^m следует, что $\mathbb{E}_{i,m}[r_i^m] \leqslant \frac{1}{2} - \delta$.

Интуиция, скрывающаяся за следующими действиями такова: если $I(M : y_i)$ — малая величина, то сообщение Боба «почти ничего» не сообщает об i-ом бите и, как следствие, если у Алисы $x = i$, мы получим ошибку с вероятностью примерно $\frac{1}{2}$. Попробуем формализовать эту стратегию.

$$\begin{aligned}
I(M : y_i) &= H(y_i) - H(y_i \mid M) \\
&= [H(y_i) = 1, \text{ поскольку распределение равномерное}] \\
&= 1 - \mathbb{E}_m[H(y_i \mid M = m)] \\
&= [y_i \text{ и } M \text{ независимы относительно } x] \\
&= 1 - \mathbb{E}_m[H(y_i \mid M = m, x = i)] \\
&= 1 - \mathbb{E}_m[H(r_i^m)].
\end{aligned}$$

Теперь попробуем оценить всю сумму:

$$\begin{aligned}
\sum_i I(M : y_i) &= \sum_i 1 - \mathbb{E}_m[H(r_i^m)] \geqslant [\text{неравенство Йенсена}] \\
&\geqslant \sum_i 1 - H(\mathbb{E}_m[r_i^m]) \\
&= n - n \sum_i \frac{1}{n} H(\mathbb{E}_m[r_i^m]) \geqslant [\text{неравенство Йенсена}] \\
&\geqslant n - nH(\mathbb{E}_{i,m}[r_i^m]) \\
&\geqslant n\left(1 - H\left(\frac{1}{2} - \delta\right)\right) \\
&= \Omega(\delta^2 n).
\end{aligned}$$

Итого, мы доказали, что $D^1(\pi) \geqslant \Omega(\delta^2 n)$.

Похожих идей будем придерживаться и в случае обычных коммуникационных протоколов.

4.8 Внутреннее и внешнее информационное разглашение

Пусть дана функция $f : X \times Y \to Z$ и соответствующий ей коммуникационный протокол π. Пусть также на $X \times Y$ задано некоторое распределение μ. Тогда на листьях протокола π естественным образом задаётся распределение: вероятность каждого листа равна сумме вероятностей входов, ведущих к этому листу. Для удобства мы часто будто говорить просто о распределение на протоколе π и обозначать его энтропию, например, через $H(\pi)$.

Определение. *Внешней взаимной информацией* называется число

$$\mathrm{IC}_\mu^{\mathrm{ext}}(\pi) = I(\pi : X, Y).$$

Внутренней взаимной информационной стоимостью называется число

$$\mathrm{IC}_\mu^{\mathrm{int}}(\pi) = I(\pi : X \mid Y) + I(\pi : Y \mid X).$$

Теорема 4.11. Пусть π — протокол некоторой детерминированной коммуникационной задачи с мерой μ на входах. Тогда

$$D(\pi) \geqslant \mathrm{IC}_\mu^{\mathrm{ext}}(\pi) \geqslant \mathrm{IC}_\mu^{\mathrm{int}}(\pi).$$

Доказательство. Очевидно, что

$$D(\pi) \geqslant \log |M| \geqslant H(\pi) \geqslant I(\pi : X, Y).$$

Таким образом, мы получили первое неравенство.

Теперь докажем второе:

$$I(\pi : X, Y) \geqslant I(\pi : X \mid Y) + I(\pi : Y \mid X).$$

По chain rule легко получить, что

$$I(\pi : X, Y) = \sum_i I(\pi_i : X, Y \mid \pi_{<i}),$$

где π_i — вероятность попасть в вершину на i-ом шаге. Также верно, что

$$\sum_i I(\pi_i : X, Y \mid \pi_{<i}) \geqslant \sum_i I(\pi_i : X \mid Y, \pi_{<i}) + I(\pi_i : Y \mid X, \pi_{<i}),$$

так как по $\pi_{<i}$ мы знаем в какой вершине протокола мы находимся после i шагов. Значит, в этой вершине либо Алиса либо Боб посылают бит, откуда следует, что одно из слагаемых $I(\pi_i : X \mid Y, \pi_{<i}), I(\pi_i : Y \mid X, \pi_{<i})$ равно нулю. Также по chain rule получаем, что

$$\sum_i I(\pi_i : X \mid Y, \pi_{<i}) + I(\pi_i : Y \mid X, \pi_{<i}) = I(\pi : X \mid Y) + I(\pi : Y \mid X),$$

откуда и следует второе неравенство из условия теоремы. ■

Эта оценка достаточно точная. Для любого протокола π существует такая мера μ, что $\log L(\pi) = \mathrm{IC}^{\mathrm{ext}}_{\mu}(\pi)$, где L — формульная сложность.

Теорема 4.12 (Храпченко). $L(\oplus_n) \geqslant \Omega(n^2)$, где функция $\oplus_n$ равна чётности количества единиц в записи числа длины n.

Доказательство. Покажем, что для любого протокола π задачи $\mathrm{KW}_{\oplus_n}$ существует такое распределение μ, что $\mathrm{IC}^{\mathrm{int}}_{\mu}(\pi) \geqslant 2\log n$. Отсюда будет следовать, что $D(\pi) \geqslant 2\log n$ и $L(\oplus_n) \geqslant n^2$. Распределение μ будет равномерным на всех парах вида $(x, x \oplus e_i)$, где $\oplus_n(x) = 0$, то есть в x чётное число единиц, а строка e_i имеет единицу в позиции i и нули во всех остальных. Таким образом, пары входов из распределения μ всегда будут отличаться только в одном бите. По определению,

$$\mathrm{IC}^{\mathrm{int}}_{\mu}(\pi) = I(\pi : X \mid Y) + I(\pi : Y \mid X).$$

Рассмотрим одно из слагаемых $I(\pi : X \mid Y)$. Имеем:

$$I(\pi : X \mid Y) = H(X \mid Y) - H(X \mid \pi, Y) = H(i \mid Y) - H(i \mid \pi, Y) = H(i) = \log n.$$

Аналогичное равенство верно и для $I(\pi : Y \mid X)$. Таким образом, $\mathrm{IC}^{\mathrm{int}}_{\mu}(\pi) \geqslant 2\log n$, что и требовалось. Верхнюю оценку на $D(\pi)$ нетрудно доказать, явно построив функцию. ■

5 Колмогоровская сложность

5.1 Способы описания и их базовые свойства

Определение. Пусть $F\colon \{0,1\}^* \to \{0,1\}^*$ — какая-то вычислимая функция. Тогда *сложностью описания x относительно F* мы будем называть число

$$K_F(x) = \min\{|p| : F(p) = x\}.$$

Если такого p не найдётся, то $K_F(x) = +\infty$.

Определение. Будем говорить, что *способ описания F не хуже способа описания G*, если существует такая константа C_{FG}, что для любого x выполнено

$$K_F(x) \leqslant K_G(x) + C_{FG}.$$

Определение. Будем называть способ описания *оптимальным*, если он не хуже любого другого способа описания.

Теорема 5.1. Существует оптимальный способ описания.

Доказательство. Пусть $H(x, y)$ — такая функция, что любая вычислимая функция $F_0(\,\cdot\,)$ представляется в виде $H(x_0, \,\cdot\,)$ для некоторого x_0.

Построим вычислимую функцию U, определённую на строках вида

$$w = a_1 a_1 a_2 a_2 \dots a_n a_n 01 y,$$

и действующую по правилу

$$U(w) = H(a_1 a_2 \dots a_n, y).$$

Докажем, что это оптимальный способ описания. Действительно, пусть F — какая-то вычислимая функция. Тогда для некоторого x выполнено $H(x, \cdot) = F(\cdot)$, и

$$K_U(y) \leqslant K_F(y) + 2|x| + 2.$$

Значит, способ описания U не хуже способа описания любой вычислимой функции, что и требовалось доказать. ■

Ясно, что если F, G — оптимальные способы описания, то $|K_F(x) - K_G(x)| \leqslant C_{FG}$ для всех x и некоторой константы C_{FG}.

Определение. Будем называть *колмогоровской сложностью* x число $K(x) := K_u(x)$, где u — какой-то оптимальный способ описания.

Мы не уточнили, какой из оптимальных способов описания рассматриваем, как следствие это означает, что мы определили колмогоровскую сложность с точностью до аддитивной константы. Поэтому говорить о сложности одной строки не имеет смысла (действительно, если взять конкретную строку w, то у нас существует алгоритм, в тексте которого уже присутствует эта строка w, и он выписывает её на пустом входе). Чтобы данное определение обрело смысл, мы будем иметь ввиду, что у нас есть семейство строк, параметризованное каким-либо параметром (чаще всего n).

Рассмотрим нетокорые простые свойства колмогоровской сложности.

Лемма 5.2.

(1) $K(x) \leqslant |x| + c$;

(2) $K(xx) \leqslant |x| + c$;

(3) если в слове x длины n не более np единиц, то

$$K(x) \leqslant h(p)n + O(1),$$

где $h(p)$ — энтропия нечестной монетки, выпадающей орлом с вероятностью p, где $0 \leqslant p \leqslant 1$.

Доказательство. Первое и второе утверждения следуют и того, что существует алгоритм, копирующий свой вход на выходную ленту, а также алгоритм, который делает это дважды.

Для доказательства третьего пункта рассмотрим множество S, состоящее их всех слов длины n, в которых не более pn единиц. Ясно, что $x \in S$, и его можно описать при

помощи номера в этом множестве (подойдёт любая нумерация элементов). Осталось вспомнить, что

$$|S| \leqslant \sum_{k=1}^{np} \binom{n}{k} \leqslant 2^{h(p)n}$$

(см. утверждение 1.5). ■

Теорема 5.3. Пусть функция $M\colon \{0,1\}^* \to \mathbb{N}$ такова, что для любого $x \in \{0,1\}^*$ верно $M(x) \leqslant K(x)$, в частности, M всюду определена; а также для любой константы C существует такое $w \in \{0,1\}^*$, что $M(w) > C$. Тогда M — невычислима.

Доказательство. Хотим провести рассуждения, аналогичные рассуждениям в *парадоксе Бэрри*: рассмотрим выражение «Наименьшее натуральное число, которое нельзя описать менее чем одиннадцатью русскими словами». Поскольку слов конечное число, существует конечное множество фраз из менее чем одиннадцати слов, и, следовательно, конечное подмножество натуральных чисел, определяемых фразой из одиннадцати слов. Однако множество натуральных чисел бесконечно, следовательно, существуют числа, которые нельзя определить фразой из менее чем одиннадцати слов. Среди них, очевидно, существует наименьшее натуральное число (наименьшее число можно выбрать из любого подмножества натуральных чисел), «не описываемое менее чем одиннадцатью словами». Но именно это число определяется приведённой выше фразой, и в ней менее одиннадцати слов, а значит, оно не может являться искомым наименьшим числом и не может описываться данной фразой. Возникает парадокс: должно существовать число, описываемое данной фразой, но поскольку выражение само себе противоречит, не может существовать числа, им описываемого.

Перейдём к доказательству теоремы. Рассмотрим первое в лексикографическом порядке слово x_c, для которого выполняется условие $M(x_c) \geqslant c$ для некоторой константы c. Определим функцию $F(\overline{c}) = x_c$, где $\overline{c}$ — битовая запись числа c. Заметим, что если M вычислима, то и функция F будет вычислимой. Но тогда $K(x_c) \leqslant \log c + c_0$, поскольку $|\overline{c}| = \log c$, и мы получаем противоречие, так как при больших c

$$M(x_c) \geqslant c > \log c + c_0 \geqslant K(x_c),$$

что неверно. Значит, функция M невычислима. ■

Следствие 5.4. Любой оптимальный способ нельзя всюду определить. ■

Теорема 5.5. 99% слов $x \in \{0,1\}^n$ длины n имеют сложность $n - O(1)$.

Доказательство. Зафиксируем какой-то оптимальный способ описания U. Тогда число описаний длины $\leqslant n - c$ не превосходит 2^{n-c+1} (это число слов длины не более, чем $n - c$). Значит, доля слов, у которых «короткое» описание, не превосходит 2^{-c+1}, а это число можно сделать сколь угодно малым, выбирая достаточно большое c. ■

5.2 Условная колмогоровская сложность

Рассмотрим вычислимую функцию $F(p, y)$. Будем говорить, что *сложность описания x относительно функции F при условии y* равна

$$K_F(x \mid y) = \min\{|p| : F(p, y) = x\}.$$

Теорема 5.6. Для любого y существует оптимальный способ описания. ■

Определение. Определим *колмогоровскую сложность x при условии y* как

$$K(x \mid y) = K_U(x \mid y)$$

для некоторого оптимального способа описания U при фиксированном y.

Нетрудно доказать несколько свойств условной колмогоровской сложности:

(a) $K(x \mid y) \leqslant K(x) + O(1)$;

(b) $K(f(x) \mid x) = O(1)$ для любой вычислимой функции f.

Определение. Определим колмогоровскую сложность пары $K(x, y) = K(\langle x, y\rangle)$, где $\langle x, y\rangle$ — какая-то вычислимая кодировка.

Для энтропии мы знаем, что $H(x, y) = H(x) + H(y \mid x)$. Хочется доказать аналогичное утверждение про колмогоровскую сложность. Равенство для колмогоровской сложности не всегда верно. Кроме того,

$$K(x, y) \not\leqslant K(x) + K(y \mid x) + O(1).$$

Это можно видеть из следующего примера:

Пример 5.1. Рассмотрим пары (x, y), для которых $|x| + |y| = n$. Таких пар $n \cdot 2^n$. Тогда очевидно, что найдётся пара сложности $\geqslant n + \log n$ (см. теорему 5.5). В то же время,

$$K(x) + K(y \mid x) + O(1) \leqslant k + (n - k) + O(1) = n + O(1).$$

При больших n последнее выражение явно меньше, чем $n + \log n$.

Теорема 5.7. На самом деле верно следующее неравенство:

$$K(x, y) \leqslant K(x) + K(y \mid x) + O(\log K(x, y)).$$

Доказательство. Пусть $U_1(p) = x$, $U_2(q, x) = y$, где U_1, U_2 — оптимальные способы описания, $|p| = n = K(x)$, $|q| = K(y \mid x)$. Запишем n в виде двоичной строки $n = n_1 n_2 \ldots n_k$. Определим функцию U на строках вида

$$n_1 n_1 n_2 n_2 \ldots n_k n_k 01 pq,$$

где p, q — описания некоторых x, y, по правилу

$$U(n_1 n_1 n_2 n_2 \dots n_k n_k 01 pq) = \langle U_1(p), U_2(q)\rangle = \langle x, y\rangle.$$

Эта функция вычислима, так как мы можем (благодаря закодированной длине p) отделить p от q, и поскольку функции U_1 и U_2 вычислимы. При этом длина описания пары $\langle x, y\rangle$ по определению не превосходит

$$K(x) + K(y \mid x) + 2\log K(x) + 2 = K(x) + K(y \mid x) + O(\log K(x)).$$

Отсюда очевидным образом следует требуемое неравенство. ■

Теорема 5.8 (Колмогоров, Левин). Имеет место равенство

$$K(x, y) = K(x) + K(y \mid x) + O(\log K(x, y)).$$

Доказательство. Неравенство в одну сторону — теорема 5.7. Докажем неравенство в другую сторону.

Зафиксируем некоторую пару $\langle x, y\rangle$ и положим $n := K(x, y)$. Пусть

$$S = \{(a, b) \mid K(a, b) \leqslant n\}.$$

Ясно, что $|S| \leqslant 2^{n+1}$. Для каждого слова v будем рассматривать сечение

$$S_v = \{w \mid \langle v, w\rangle \in S\}.$$

Пусть $m := \lfloor \log |S_x| \rfloor$, то есть m — такое целое число, что $2^m \leqslant |S_x| \leqslant 2^{m+1}$.

Заметим, что зная n, мы можем перечислять элементы S. Если к тому же мы знаем x, то можно у S оставлять только пары с первой координатой x и получить перечисление сечения S_x. Чтобы задать y, достаточно указать его номер в этом перечислении. Для этого достаточно $m + O(1)$ битов. Поскольку нужно также знать n, к этому значению прибавляется $O(\log n)$. Таким образом,

$$K(y \mid x) \leqslant m + O(\log n),$$

Оценим теперь $K(x)$. Рассмотрим множество B таких строк v, что $|S_v| \geqslant 2^m$. Ясно, что $|B| \leqslant 2^{n+1}/2^m = 2^{n-m+1}$, так как иначе значение $|S| = \sum |S_v|$ превышало бы 2^{n+1}. Для перечисления множества B достаточно знать n и m. Действительно, можно просто перечислять пары из S, и как только найдётся 2^m пар с одинаковой первой координатой, переместить эту координату в перечисление множества B. Таким образом, исходное слово x можно задать, используя $(n - m) + O(\log n)$ битов (см. оценку на мощность $|B|$). Значит,

$$K(x) \leqslant (n - m) + O(\log n).$$

Остаётся сложить два полученных неравенства:

$$K(x) + K(y \mid x) \leqslant n + O(\log n) = K(x, y) + O(\log K(x, y)),$$

что и требовалось. ■

5.3 Применение колмогоровской сложности

Колмогоровская сложность применяется в задачах, связанных с попытками доказать или опровергнуть NP-полноту задачи MCSP, в которой по таблице истинности функции $f\colon \{0,1\}^n \to \{0,1\}$ и числу $s < 2^n$ необходимо определить, существует ли булева схема размера S, которая считает данную функцию. Нетрудно понять, что эта задача принадлежит классу NP. Но не доказано, является ли эта задача NP-полной.

Рассмотрим более простую задачу. Обозначим за L_k множество языков, принимаемых детерминированными автоматами с одной лентой и k головками на ней, причём головки едут только в одну сторону.

Утверждение 5.9. $L_{k+1} \not\subseteq L_k$, то есть $L_{k+1} \setminus L_k \neq \varnothing$.

Доказательство. Рассмотрим язык L, состоящий из строк вида

$$w_1\#w_2\#\ldots\#w_{m-1}\#w_m\#w_m\#w_{m-1}\#\ldots\#w_2\#w_1,$$

где m — фиксировано, а w_i — произвольные слова. Покажем, что если $m > \frac{k(k-1)}{2}$, то язык L не распознаётся автоматом с k головками, а если это не так, то распознаётся.

В каждом слове языка L есть по два слова w_i. Будем называть их «левым» и «правым» соответственно. Пусть $m \leqslant \frac{k(k-1)}{2}$. Опишем автомат с k головками, который распознаёт язык L. Сначала все k головок расположены на началах левых слов $w_1, w_2, \ldots, w_k$. Следующим действием последняя головка, которая стоит в начале w_k, доходит до начала правого слова w_{k-1}. Далее последняя головка двигается вправо одновременно с предпоследней, и таким образом проверяется равенство левого и правого w_{k-1}. Аналогично проверяется w_{k-2}, w_{k-3} и т. д. В результате мы проверим равенство левых и правых слов w_i, где $1 \leqslant i \leqslant k-1$. При этом последнюю головку больше использовать не получится.

Тем самым, «потратив» одну головку мы проверили равенство $k-1$ слова с начала и с конца. Далее можно повторить алгоритм для $k-1$ головки и слова, в котором выкинуты первые и последние $k-1$ слов w_i. В итоге мы проверим

$$(k-1) + (k-2) + \cdots + 1 = \frac{k(k-1)}{2}$$

слов, что и требовалось.

Пусть теперь $m > \frac{k(k-1)}{2}$. Посмотрим на левый и правый блоки с номером l (то есть слово w_l). Пусть в какой-то момент головки i и j стоят на разных копиях слова w_l. Тогда эта пара головок (i, j) ни в какой другой момент времени не будет находиться в одинаковых словах w_r, где $r \neq l$, так как головки движутся только вправо. Тогда, так как $m > \frac{k(k-1)}{2}$, а это как раз число различных неупорядоченных пар (i, j), то для какого-то слова w_l верно, что в каждый момент времени только в одной копии w_l может находиться головка.

Построим протокол, который будет запоминать состояния автомата и позиции головок для каждого такого момента, что либо какая-то головка оказалась в слове

w_l, либо какая-то головка покинула слово w_l. Попробуем восстановить w_l по протоколу. Подставим x на место w_l. Если автомат выдал такой же протокол работы, то x потенциально подходит. Предположим, что нашлось 2 подходящих слова x и y, для которых протокол работает правильно. Тогда рассмотрим слово $w_{x,y}$, равное слову w, в котором левое слово w_l заменили на x, а правое слово w_l заменили на y. Заметим, что для слов w и $w_{x,y}$ алгоритм работает одинаково. Значит, $w_{x,y}$ лежит в языке L, что неверно. Таким образом, по протоколу мы однозначно восстанавливаем w_l.

Пусть длина слова w равна n, и в нём зафиксированы слова $w_i, i \neq l$. Тогда понятно, что размер протокола для w равен $O(k^3 \log n)$, где k — число головок. Осталось заметить, что символов для кодирования строки w, с фиксированными $w_i, i \neq l$, не меньше, чем $\sum_{i \neq l} |w_i| + O(k^3 \log n)$. Заметим, что существуют такие w_i, что $|w_i| = s$ для любого i и $K(w_1 \# \ldots \# w_m \# w_m \# \ldots \# w_1) \geqslant ms - c$. Но из предыдущих рассуждений мы получили, что колмогоровская сложность w равна $(m-1)s + O(\log s)$, так как $\sum_{i \neq l} |w_i| = (m-1)s$. Тогда при больших s мы получаем противоречие. А значит язык L не лежит в L_k, теорема доказана. ■

5.4 Случайные строки по Мартин-Лёфу

Рассмотрим бесконечную строчку

$$w = x_1 x_2 x_3 \ldots x_i x_{i+1} \ldots$$

Хотим узнать, случайна эта строка или получена каким-то алгоритмом. Предположим, что мы знаем биты $x_1, x_2, \ldots, x_{n-1}$. Можно ли узнать бит x_n? Если бы можно было узнать все биты x_n по предыдущим, то понятно, что строка не случайная. Но как часто можно узнавать следующий бит по предыдущим в случайной строке?

Хотелось бы сказать, что строка случайна, если существует такая константа c, что для любого n верно $K(x_{\leqslant n}) \geqslant n - c$, где $x_{\leqslant n}$ — префикс x длины n (то есть когда колмогоровская сложность достаточно большая). Однако оказывается, что не существует такой последовательности x_i, для которой это неравенство выполнено.

Теорема 5.10. Для любой последовательности $x_1 x_2 x_3 \ldots$ и числа n_0 найдётся такое $n > n_0$, что $K(x_{\leqslant n}) \leqslant n - O(\log n)$. В частности, для любой константы c найдётся такое n, что $K(x_{\leqslant n}) < n - c$.

Доказательство. Рассмотрим любую последовательность $x_1 x_2 x_3 \ldots$ и её префиксы $x_1 \ldots x_k$. Для каждого k можно найти такое m, что $1 x_1 x_2 \ldots x_k = m$, и рассмотреть префикс длины $n = k + m$. Тогда $K(x_{\leqslant n}) \leqslant m + O(1)$, так как зная последние m символов префикса длины n мы можем восстановить весь префикс. Но длина всего префикса равна $n = m + \log m$, откуда мы получаем, что

$$K(x_{\leqslant n}) \leqslant n - O(\log n).$$

Осталось заметить, что n можно выбрать сколь угодно большим. ■

Определение. Функция F называется *беспрефиксной*, если из того, что F определена на x и y следует, что x не является префиксом y, и наоборот.

Для беспрефиксных функций можно определить понятие колмогоровской сложности так же, как и для обычных функций. Мы будем обозначать её через $\mathrm{KP}(x)$.

Определение. Строка w называется *случайной по Мартин-Лёфу*, если существует такая константа c, что $\mathrm{KP}(w_{\leqslant n}) \geqslant n - c$ для всех $n \in \mathbb{N}$.

В определении колмогоровской сложности мы пользовались тем, что существует оптимальный способ описания для беспрефиксных функций. Докажем это в следующей теореме.

Теорема 5.11. Существует оптимальный беспрефиксный способ сжатия. То есть существует такой алгоритм A, что для любой вычислимой функции F выполняются условия:

- если F — беспрефиксная, то $A(F) = F$;
- если F — не беспрефиксная, то $A(F) = F'$, где F' — беспрефиксная.

Доказательство. Будем строить алгоритм A. Пусть на вход дана функция F, и мы хотим посчитать $A(F)(x)$ для каждого x. Пусть зафиксирована строка x. Алгоритм A запускает F параллельно на всех входах (диагональным способом). Тогда возможны две ситуации:

- Мы ещё не посчитали $F(x)$, но посчитали $F(y)$ и $F(z)$, причём y является префиксом z. Тогда алгоритм A зацикливается, и $A(F)(x)$ не определено.
- Мы посчитали $F(x)$ и не пришли к противоречию с беспрефиксностью. Тогда $A(F)(x) = F(x)$.

Алгоритм A — вычислим, а также любой функции сопоставляет беспрефиксную. Занумеруем все вычислимые функции. Определим наш оптимальный префиксный способ описания $U(x, y) = A(x)y$, где под x подразумевается вычислимая функция. Функция U определена на строчках вида

$$w = x_1x_1x_2x_2 \ldots x_kx_k01y,$$

и $U(w) = A(x_1x_2 \ldots x_n)y$, где $x_1x_2 \ldots x_n = x$— вычислимая функция. Нетрудно показать, что U — беспрефиксная, так как если строка $A(x)y = x_1x_1 \ldots x_kx_k01y$ является префиксом $A(x')y' = x'_1x'_1 \ldots x'_kx'_k01y'$, то $A(x) = A(x')$. Но $A(x)$ — беспрефиксная функция, а значит U не определена, на одной из этих строк. Осталось лишь показать, что U — оптимальный способ описания. Используя тот факт, что для беспрефиксных функций F имеется равенство $A(F) = F$, можно доказать оптимальность данного способа так же, как в теореме об оптимальном способе описания для всех функций. ■

Положим $U(x, y) = A(x(y))$. Тогда:

(1) $\mathrm{KP}(x) \leqslant K(x) + 2\log K(x) + O(1)$.

Чтобы доказать это неравенство, нам нужно применить кодировку

$$p \mapsto a_1 a_1 a_2 a_2 \dots a_k a_k 01 p,$$

где p — описание x, и $|p| = a_1 a_2 \dots a_k$. Тогда $k \leqslant \log K(x)$ и потому

$$\mathrm{KP}(x) \leqslant K(x) + 2\log K(x) + O(1).$$

(2) *Мера Бернулли неслучайных последовательностей равна нулю.*

Рассмотрим некоторую неслучайную последовательность

$$x_1 x_2 x_3 \dots x_i x_{i+1} \dots$$

Так как она неслучайна, то для любого c существует такая последовальность $\{n_i\}_{i \geqslant 1} \subset \mathbb{N}$, что $\mathrm{KP}(x_{\leqslant n_i}) \leqslant n_i - c$. Тогда

$$\sum_{\{x\,:\,\mathrm{KP}(x) \leqslant n_i - c,\ |x| = n_i\}} 2^{-n_i} \leqslant \sum_x 2^{-\mathrm{KP}(x) - c} \leqslant 2^{-c} \cdot \sum_x 2^{-\mathrm{KP}(x)} \leqslant 2^{-c}.$$

Последнее неравенство выполнено, так как код префиксный, то есть можно применить неравенство Крафта–Макмиллана. Если A_i — множество строк с плохими префиксами $x_1 x_2 \dots x_i$, то мера Бернулли неслучайных последовательностей

$$M\left(\bigcup_i A_i\right) \leqslant \sum_{\{x\,:\,\mathrm{KP}(x) \leqslant |x| - c\}} 2^{-|x|} < 2^{-c}.$$

Осталось устремить c к бесконечности и получить требуемое.

Теорема 5.12 (закон больших чисел в форме Харди–Литлвуда). Для почти всех последовательностей $x = x_1 x_2 \dots x_n \dots$ (то есть с вероятностью 1) выполнено:

$$\left| \frac{x_1 + x_2 + \dots + x_n}{n} - \frac{1}{2} \right| = O\left(\sqrt{\frac{\log n}{n}} \right).$$

Доказательство. Пусть в $x_1 \dots x_n$ всего $p_n \cdot n$ единиц и $(1 - p_n) \cdot n$ нулей. Тогда

$$\mathrm{KP}(x_1 \dots x_n) \leqslant K(x_1 \dots x_n) + O(\log n) \leqslant H(p_n) \cdot n + O(\log n).$$

Пусть $p_n := \frac{1}{2} + \delta_n$. Отметим, что δ_n — это в точности то число, чей модуль мы хотим оценить. Разложим $H(p_n)$ в ряд в окрестности $\frac{1}{2}$:

$$H(\tfrac{1}{2} + \delta_n) \cdot n = \left(1 - c_H \cdot \delta_n^2 + o(\delta_n^2)\right) \cdot n \leqslant (1 - c'_H \cdot \delta_n^2) \cdot n.$$

Таким образом, для случайной последовательности (то есть с вероятностью 1):

$$n - c \leqslant \mathrm{KP}(x_1 \dots x_n) \leqslant n - c'_H \cdot \delta_n^2 \cdot n + O(\log n).$$

Значит, $\delta_n^2 \leqslant O\left(\frac{\log n}{n}\right)$. ■