ЛЕКЦИИ ШКОЛЫ
АНАЛИЗА ДАННЫХ ЯНДЕКСА

Н.К. Верещагин, Е.В. Щепин

Информация, кодирование и предсказание

Яндекс

ИЗДАНИЕ ОСУЩЕСТВЛЯЕТСЯ
ПРИ ПОДДЕРЖКЕ КОМПАНИИ «ЯНДЕКС»

Н. К. Верещагин, Е. В. Щепин

Информация, кодирование и предсказание

Москва
ФМОП
МЦНМО
2012

ББК 32.81
В31

Верещагин Н. К., Щепин Е. В.

В31 Информация, кодирование и предсказание. — М.: ФМОП, МЦНМО, 2012. — 236 с.

ISBN 978-5-904696-05-4 (ФМОП)
ISBN 978-5-94057-920-5 (МЦНМО)

Предлагаемая книга — это одновременно учебник и оригинальная монография по теории информации. Две независимые друг от друга части, составляющие книгу, написаны авторами на основе собственных лекций, читающихся в Школе анализа данных Яндекса.

Автор первой части, Е. В. Щепин, рассматривает понятия теории информации как базу для решения задач машинного обучения, и прежде всего — задач построения классификатора по эмпирическим данным. Специальное внимание автор уделяет изучению случаев многомерных ограниченных данных, когда прямые методы оценки функций распределения вероятностей неприменимы. Обсуждение этих вопросов редко встречается в работах по теории информации. В предлагаемой книге изложение доведено до описания практических методов.

Во второй части, написанной Н. К. Верещагиным, исследуются задачи о поиске на базе понятия информации по Хартли. В этой части описаны различные применения теории колмогоровской сложности (сложности описаний), даны основы логики знаний и теории коммуникационной сложности. К теоретическому материалу прилагается множество задач для самостоятельного решения.

В обеих частях отводится много места основам классической теории информации Шеннона и её применению к кодированию информации. В первой части это изложение ведется с позиций конструирования алгоритмов решения проблем, во второй части большое внимание уделено концептуальным аспектам классической теории Шеннона.

Книга завершается дополнением, взятым из выдающейся книги М. М. Бонгарда «Проблема узнавания» (1967), где с позиций теории информации изучается вопрос об оценке степени истинности описания. Эта важная тема, непосредственно примыкающая к рассматриваемым в книге проблемам, служит подтверждением перспективности теории информации для развития новых методов анализа данных.

ББК 32.81

978-5-904696-05-4 (ФМОП)
978-5-94057-920-5 (МЦНМО)

Оглавление

Часть вторая

Предисловие к книге Евгения Щепина и Николая Верещагина «Информация, кодирование и предсказание»

Интерес к применению методов теории информации к проблемам распознавания образов, и, более общо, к машинному обучению и анализу данных, возник очень давно. Норберт Винер в книге «Кибернетика» в 1947г. отмечал важность применения теоретико-информационного подхода для построения распознающей машины. Через 20 лет в 1967 Михаил Бонгард в книге «Проблема узнавания» впервые дал конкретный пример такого применения. Он показал, что с помощью такого подхода можно естественным образом количественно анализировать качество работы обученного распознающего автомата. Двумя годами позже, в 1969 году, примерно эту же проблему количественного анализа качества узнавания и догадки рассмотрел Сатоши Ватанабе в книге «Узнавать и догадываться: количественное изучение вывода и информация». В последние годы сильно возрос интерес к применению теории информации для исследования задач машинного обучения и анализа данных, особенно в связи с необходимостью решения этих задач для поиска среди очень больших массивов текстовой информации. Эта область приложений, в свою очередь, породила исследования, расширяющие базовые представления и предлагающие новые методы во внутренней структуре теории информации. Теория информации, можно сказать, переживает сейчас новый период развития. Хорошей иллюстрацией этого является предлагаемая Вам книга Евгения Щепина и Николая Верещагина. Задуманная как учебное пособие по теории информации, она параллельно с изложением фундаментальных моделей и методов традици-

онной теории информации ясно обозначает современные тенденции в ее развитии. Несмотря на небольшой объем, книга содержит модели и алгоритмы, определяющие новые направления развития теории информации, не представленные в современных публикациях по теории информации. Читатель, работающий в области анализа данных и желающий использовать в своих исследованиях теорию информации, найдет в этой книге много направляющих идей.

Книга состоит из двух частей. Первая написана Евгением Щепиным, вторая — Николаем Верещагиным. Для полноты изложения современного теоретико-информационного подхода к проблемам анализа данных, в конце книги в качестве дополнения мы приводим текст раздела из упомянутой выше монографии М. Бонгарда, одного из пионеров и классиков теории обучения машин.

Обе части книги написаны независимо и их можно читать независимо. Они обьединены тем, что обе нацелены на самые новейшие приложения теории информации, как сугубо инженерные, так и теоретические (ряд интереснейших задач, как отмечалось, в книге описан впервые). Изложение в обеих частях почти не требует вузовских знаний, однако требует определенного навыка точного мышления. Однако и при отсутствии такого навыка изучение материала книги доступно благодаря его структурной организации, позволяющей «не часто возвращаться назад» при чтении. Авторы обеспечили это подбором очень интересных вопросов-примеров, и в особенности тем, что многие из этих примеров даны в форме простых исследовательских задач, решение которых не только способствует лучшему усвоению материала, но и позволяет приобрести навыки придумывания и решения новых задач.

В первой части большое внимание уделено задачам, непосредственно примыкающим к проблемам машинного обучения, особенно связанным с построением классификатора на основе примеров. В частности, разбираются трудные задачи оценки функции плотности вероятностей в пространствах очень большой размерности по малым выборкам. Я назвал бы эти исследования Е. Щепина основами нового теоретико-информационного напрвления в построении классификаторов. Уже предложенные процедуры представляют большую практическую ценность и являются источником, вдохновляющим на новые глубокие теоретические разработки, особенно в деле поиска регуляризаторов для оценки редко наблюдаемых событий.

Одна из особенностей второй части заключается в том, что в ней

в рамках общей схемы теории информации рассматриваются задачи оценки колмогоровской сложности (в современной зарубежной литературе эти задачи часто объединяют термином «алгоритмическая теория информации»). В традиционных учебниках по классической теории информации, и в особенности в учебниках, ориентированных на инженеров, эта проблематика не рассматривается, несмотря на то, что сами задачи активно исследуются (по ним не только имеется большой поток научных статей, но и публикуются монографии). Поэтому этот относительно маленький раздел второй части представляется важным, особенно для студентов и специалистов, которые впервые изучают теорию информацию и методы ее применения. Николай Верещагин построил изложение этого раздела таким образом, что он может служить базой и для чтения других быстро развивающихся направлений в анализе данных, например, для теории машинного обучения, для которой критическими являются (1) изучение существующих оценок гарантий качества моделей, построенных на анализе примеров, и (2) разработка новых оценок. Исследования по решению проблем машинного обучения на основе моделей колмогоровской сложности — активная область, в развитии которой Н. Верещагин непосредственно участвует. Поэтому ему удалось, несмотря на краткость, настолько выразительно описать это перспективное направление, что впервые изучающий книгу читатель имеет возможность включиться в эти исследования.

В обеих частях, как уже говорилось, подробно разбираются и классические задачи теории информации. Особенно подробно разбираются задачи представления и кодирования данных для их сжатого хранения и передачи, а также задачи возможности и оценки предсказания в различных условиях.

В описании классических задач теории информации в текстах первой и второй части имеются дублирования, но они полезны, так как по-разному построены и дополняют друг друга. Особенно хорошо этот разный подход к изложению проявляется в подборе задач-упражнений. Если Е. Щепин старается показать, как сложные процедуры можно заменить эффективными аппроксимациями и на каких идеях можно строить эти хорошие аппроксимации, то Н. Верещагин ищет наиболее понятное, простое и короткое описание точных процедур.

Самое большое "дублирование"имеет место в описании базовых понятий — энтропии и информации и их свойств. В обех частях читатель найдет ясное и полное их обсуждение. И тем не менее, и даже в

этих описаниях читатель найдет полезные дополнения. В частности, интересным представляется, что Н. Верещагин наиболее детально описывает алгебраическую природу необходимых конструкций, а Е. Щепин подчеркивает их вероятностную природу. Н. Верещагин уделяет большое внимание асимптотическим свойствам методов, а Е. Щепин — свойствам, которые проявляются на конечных конструкциях. В итоге читатель получает обьемную картину всего здания теории.

Книга Е. Щепина и Н. Верещагина — не просто техническое пособие к лекциям. Получилась новая интересная монография по основам и приложениям теории информации, которая будет полезна и студентам, и ученым, работающим в различных областях, использующих модели теории информации.

И. Б. Мучник
август 2011 г.

Часть первая

Введение

Курс нацелен на изучение способов применения теории информации в практических задачах распознавания образов. Хотя все требуемое от теории информации по существу является вариацией одной и той же формулы энтропии, многие из тех, кто изучал теорию информации, часто бывают поставлены в тупик простейшими вопросами, возникающими при ее применении. Например, большинство выпускников механико-математического факультета МГУ не понимают, как посчитать количество информации, которую сумма цифр трехзначного числа несет об их произведении.

Интуитивное понятие информации, возникающее из обычного языка, хорошо согласуется с определением количества информации, с которым оперирует наука. Развитию интуитивного понятия способствует решение задач и разбор доказательств основных теорем.

Доказательства теорем в курсе не отличаются строгостью, ведь все эти теоремы и так хорошо известны, их цель — дать читателю лучше прочувствовать формулировки и показать в работе основные приемы.

Данный курс лекций может быть использован как для первичного ознакомления с теорией информации, так и для более глубокого ее изучения. В последних двух лекциях излагаются новые экспериментальные подходы к применению теории информации в распознавании образов, возникшие у автора при работе в группе разработчиков Яндекса.

Глава 1. Информационная емкость символа

Информация может быть заключена в словах, числах, цветах, звуках и многом другом, но в конечном счете всякую информацию можно выразить словами любого человеческого языка. Всякий язык имеет свой *алфавит* — множество символов (букв, цифр, знаков препинания и т. д.). Последовательности символов представляют *слова* языка. Вопрос, который мы решим сегодня — сколько информации может нести одно слово. Единицей измерения информации служит *бит*: один ответ на вопрос типа да-нет. Количество битов информации, которое содержит данное слово, означает, на сколько вопросов типа да-нет может отвечать это слово. Например, информационная емкость символа трехбуквенного алфавита (то есть однобуквенного слова в языке с алфавитом из трех символов), как мы увидим в дальнейшем, приблизительно равна 1.5849625 битам.

1.1 Двоичные коды

Алфавит языка компьютера состоит из нуля и единицы, а словами являются последовательности нулей и единиц, называемые *двоичными словами*. Число элементов двоичного слова называется его *длиной*. Память компьютера состоит из ячеек, называемых *битами*, в каждом из которых записан либо ноль, либо единица. Биты объединяются в восьмерки, называемые *байтами*. Таким образом, каждый байт памяти компьютера хранит двоичное слово длины 8. Как следует из нижеприведенной леммы, всего имеется $2^8 = 256$ различных *байтовых слов*.

Лемма 1.1. *Общее количество двоичных слов длины n равно 2^n.*

Доказательство. Доказательство проводится индукцией по длине последовательности. Для $n = 1$ утверждение верно. Пусть уже доказано,

что число двоичных слов длины n равно 2^n. Все двоичные слова длины $n+1$ делятся на два типа: начинающиеся с нуля и начинающиеся с единицы. Число слов каждого типа совпадает с числом слов длины n, то есть в силу предположения индукции равно 2^n. Тогда число последовательностей длины $n+1$ равно $2^n + 2^n = 2^{n+1}$. □

Количества байтовых слов хватает, чтобы представить все буквы русского и английского алфавитов, цифры, знаки препинания и все символы, изображенные на клавиатуре компьютера. Например, стандартным двоичным кодом русской буквы «а» является «10100000», код цифры «0» — «00110000», пробела — «00100000».

Таким образом, любое предложение русского языка может быть представлено с помощью двоичных кодов таким образом, что займет ровно столько байт, сколько символов (включая пробелы и знаки препинания) оно содержало.

1.2 Оптимальное кодирование

Одна из важных практических задач — как поместить в компьютер как можно больше информации.

Например, мы хотим внести в компьютер данные переписи населения России таким образом, чтобы они занимали как можно меньше памяти. Рассмотрим для начала только данные о поле граждан. Для кодирования пола человека достаточно одного бита — например, можно обозначать женский пол нулем, а мужской единицей. Получается очень компактный способ кодирования, более того, как мы сейчас докажем, лучшего способа кодирования пола не существует.

Двоичным кодированием множества M называется отображение c, ставящее в соответствие каждому элементу множества $x \in M$ двоичное слово $c(x)$ — его код. Кодирование называется *инъективным*, если коды различных элементов множества M различны.

Лемма 1.2. *Множество M допускает двоичное инъективное кодирование с длиной кода, равной n, в том и только том случае, когда число элементов множества M не превосходит 2^n.*

Доказательство. Так как число двоичных слов длины n равно 2^n по лемме 1.1, то теорема сводится к следующему очевидному утверждению: одно множество можно инъективно отобразить в другое в том и

только том случае, когда число элементов первого множества не превышает числа элементов второго множества. □

Доказанная лемма позволяет давать эффективные оценки минимально необходимого объема памяти компьютера для запоминания различного рода анкетных данных. Например, для запоминания информации о поле всех граждан России (население России будем считать равным 140 миллионам человек), мы должны зарезервировать не менее 140 миллионов бит памяти компьютера. Действительно, пусть d — какой-то способ кодирования полов. Предположим, что все записи упорядочены по алфавиту. Обозначим через c кодирование, ставящее 0 в соответствие женскому полу и 1 — мужскому. Результат переписи, кодированной по принципу «женский – 0, мужской – 1», представляет собой двоичную последовательность длины $140 \cdot 10^6$. Заранее мы не можем исключать никакого варианта переписи, поэтому все возможные результаты переписи должны инъективно кодироваться кодом d. Поэтому dc^{-1} должно инъективно кодировать множество всех двоичных слов длины $140 \cdot 10^6$. Так как число таких слов $2^{140000000}$ и по доказанной выше лемме их нельзя инъективно закодировать словами длины меньшей, чем $140 \cdot 10^6$, то, следовательно, нашу информацию нельзя разместить в памяти, заняв менее 140 миллионов битов.

1.3 Объединение в блоки

А теперь рассмотрим задачу кодирования результатов голосования. Имеется три варианта голосования: «за», «против» и «воздержался». Каждый результат можно было бы кодировать с помощью двухбитовых слов — «11», «00», «01». При этом одна комбинация, «10», осталась неиспользованной. При таком кодировании n результатов голосования займут объем $2n$ бит памяти. Но этот результат можно улучшить и поместить все результаты, заняв лишь $5n/3$ бит памяти, то есть потратив в среднем на один вариант голосования $5/3$ бита. Чтобы это сделать, мы будем объединять результаты голосования в тройки. Тогда число вариантов голосования для тройки бюллетеней равняется $3 \cdot 3 \cdot 3 = 27 < 2^5$ и может быть кодировано пятибитовыми словами. Так как число троек равно $n/3$, а каждая тройка занимает 5 бит памяти, то мы получаем обещанную оценку $5n/3$. Но и этот результат можно улучшить, если объединять в блоки по пять бюллетеней сразу. В этом

случае число возможных исходов голосования для блока составляет $3^5 = 243 < 256 = 2^8$ и потому результат по пятерке бюллетеней записывается в один байт. Таким образом, в среднем на бюллетень придется $8/5$ бита памяти. С другой стороны, так как $3^2 > 2^3$, результаты голосования по n бюллетеням имеют более чем $2^{3n/2}$ вариантов и поэтому не могут занимать меньше $3n/2$ бит памяти, то есть не существует способа кодирования, при котором на один бюллетень отводится менее 1.5 бит.

Рассмотрим теперь k-блочное разбиение бюллетеней. Пусть m — такое натуральное число, что $2^m < 3^k < 2^{m+1}$. Тогда, с одной стороны, мы можем кодировать результаты голосования, потратив в среднем на бюллетень не более $\frac{m+1}{k}$ бит, с другой стороны — не можем кодировать, потратив менее $\frac{m}{k}$ бит. k-блочное кодирование с ростом k дает сколь угодно близкие к оптимальному (1.5 бита на бюллетень) результаты.

1.4 Бит и трит

Одной из первых вычислительных машин в СССР была основанная на троичной системе счисления «Сетунь». Представим себе винчестер троичной машины, у которого элементарная ячейка памяти имеет не два, как у современных компьютеров, а три состояния — записанная в нем информация кодируется символами алфавита из трех символов (можно взять, например, 0, 1, 2), а информационнная емкость измеряется в *тритах*, выражающих информационную емкость символа троичного алфавита. Каково же соотношение бита и трита? Предположим, у нас есть «бинарный» винчестер и мы хотим переписать информацию с него на винчестер троичной машины. Какую емкость в тритах он должен иметь? Не заботясь о практической реализуемости алгоритма, мы можем решить вопрос следующим образом: содержимое двоичного винчестера, представляющее собой последовательность нулей и единиц, можно трактовать как натуральное число в двоичной системе счисления. Точно так же содержимое винчестера троичной машины представляется последовательностью нулей, единиц и двоек, которую можно трактовать как натуральное число, записанное в троичной системе счисления. Перекодирование двоичной последовательности в троичную можно реализовать как переход от одной системе счисления к другой — запись числа, представляющего собой содержимое двоичного винчестера, в троичной системе счисления и будет

являться перекодированием информации для троичной машины. Аналогично, содержимое винчестера троичной машины можно трактовать как натуральное число в троичной системе счисления, перевод которого в двоичную систему представляет собой обратное, декодирующее, отображение. Итак, пусть емкость двоичного винчестера равна n бит, а емкость троичного m трит. Тогда максимальное число, представленное содержимым двоичного винчестера, равно $2^n - 1$, а содержимым троичного винчестера — $3^m - 1$. Если $2^n < 3^m$, то любое содержимое двоичного винчестера перекодируется в троичное число, которое может поместиться на троичном винчестере. Поэтому в данном случае мы приходим к заключению, что n бит должно быть меньше, чем m трит. Если же $3^m < 2^n$, то содержимое троичного винчестера перекодируется в число, помещающееся на двоичном винчестере. Если n и m таковы, что выполнено неравенство $2^n < 3^m < 2^{n+1}$, то m трит заключено между n и $n+1$ битом. Логарифмирование последнего неравенства по основанию 2 дает $n < m \log_2 3 < n + 1$, откуда мы видим, что разность между тритом и $\log_2 3$ не превосходит $\frac{1}{m}$. Ввиду произвольности m отсюда следует, что трит следует считать равным $\log_2 3$.

1.5 Формула Хартли

Будем говорить, что *информативность* символа k-элементного алфавита M меньше чем a в том и только том случае, если при некотором n множество всех слов длины n из алфавита M может быть кодировано не более чем na-битовыми словами. Будем называть информативностью символа наибольшее число I, для которого информативность символа не меньше чем I.

Теорема 1.1 (Хартли)**.** *Информативность символа k-элементного алфавита равна* $\log_2 k$

Доказательство. Пусть M — k-элементный алфавит. Обозначим через M^n множество всех слов длины n в алфавите M. Мощность этого множества равна k^n. Пусть s таково, что

$$2^s < k^n \leqslant 2^{s+1} \tag{1.1}$$

Тогда, с одной стороны, можно кодировать M^n с помощью $s + 1$-битовых последовательностей, поэтому информативность элемента ал-

фавита M не превышает $\frac{s+1}{n}$, а с другой стороны, невозможно кодировать слова длины n с помощью s-битовых слов, откуда следует, что информативность символа алфавита M не меньше $\frac{s}{n}$. Логарифмируя неравенство (1.1) по основанию 2, получаем неравенство $s < n \log_2 k < s + 1$, откуда следует, что разность между информативностью символа и $\log_2 k$ не превосходит $\frac{1}{n}$ для любого n. Отсюда и следует требуемое равенство. □

1.6 Задача сжатия файла

Одной из необходимых программ на современном компьютере является программа сжатия-разжатия файлов — архиватор. Всякий архиватор выполняет две операции: во-первых, он пакует файлы, преобразуя их в файлы меньшего размера, чтобы сохранить на диске больше свободного пространства, а во-вторых, он делает обратное преобразование — распаковывает упакованные им файлы, восстанавливая их в точности такими, какими они были до архивации. Таким образом, всякий архиватор осуществляет взаимно-однозначное отображение множества всех файлов на его подмножество, состоящее из упакованных файлов.

На чем же основана возможность сжатия файлов? Всякий файл можно рассматривать как двоичное слово. Текстовые файлы более естественно рассматривать как байтовое или двухбайтовое (юникод) слово. В любом случае файл можно представлять себе как слово в некотором алфавите.

Первая возможность для сжатия файла заключается в оптимизации двоичного кодирования его алфавита. Например, если файл представляет собой последовательность десятичных цифр, закодированных стандартным образом, при этом каждой цифре соответствует байт, то мы можем на основании проведенных выше рассуждений закодировать его так, чтобы на символ в среднем приходилось примерно $\log_2 10 \approx \frac{10}{3}$ бита, то есть сжать его почти в два с половиной раза.

1.7 Равночастотное кодирование

Рассмотрим некоторое слово (файл) m-символьного алфавита. Пусть числа $n_1, \ldots, n_m$ выражают количество встретившихся в нем символов алфавита: n_i — количество вхождений в слово i-го символа алфавита.

Для каждого слова соответствующий набор чисел $n_1, \ldots, n_m$ назовем *декомпозицией* этого слова.

Количество слов m-символьного алфавита длины $N = n_1 + \ldots + n_m$ выражается числом m^N и является суммой чисел $C(n_1, \ldots, n_m)$, выражающих количество слов с декомпозицией $n_1, \ldots, n_m$. Формула

$$C(n_1, \ldots, n_m) = \frac{N!}{n_1!\, n_2! \cdot \ldots \cdot n_m!} \tag{1.2}$$

доказывается индукцией по m. Мы ограничимся доказательством формулы для $m = 2$.

Слово с декомпозицией n_1, n_2 однозначно определяется выбором подмножества мест, на которых должен стоять первый символ алфавита. Таких мест ровно n_1, и $C(n_1, n_2)$ равно количеству способов выбора в множестве из $n_1 + n_2$ элементов подмножества из n_1 элементов. Это число, как известно, равно $\frac{N!}{n_1! n_2!}$.

Если рассматриваемые числа достаточно велики, заменяя по формуле Стирлинга[1] $n!$ на $n^n e^{-n}$, получаем следующую асимптотическую оценку для числа слов с данной декомпозицией:

$$\frac{N!}{n_1! n_2! \cdot \ldots \cdot n_m!} \approx \frac{N^N}{n_1^{n_1} \cdot \ldots \cdot n_m^{n_m}}, \tag{1.3}$$

где приближенное равенство означает, что предел отношения при $N \to \infty$ логарифмов левой и правой частей равен единице.

Наибольшего значения выражение (1.2) достигает при $n_1 = \ldots = n_m = N/m$. В этом случае двоичный логарифм выражения (1.3) равен $N \log_2 m$. Таким образом, логарифм количества слов, в которых с равной частотой встречаются все символы алфавита, асимптотически эквивалентен логарифму количества произвольных слов такой же длины. Этот замечательный факт обеспечивает совместимость вероятностного (о котором пойдет речь в следующей лекции) и комбинаторного (или алгебраического) подходов к теории информации.

[1] Используемый здесь вариант формулы Стирлинга элементарно доказывается. А именно, перемножение неравенств $\frac{(n+1)^n}{n^n} < e < \frac{(n+1)^{n+1}}{n^{n+1}}$ для $n = 1, 2, \ldots N$ дает $\frac{(N+1)^N}{N!} < e^N < \frac{(N+1)^{N+1}}{N!}$. Откуда $\lim\limits_{N\to\infty} \frac{\log e^N N!}{\log N^N} = 1$.

1.8 Задача поиска

Интересующая нас в быту информация часто представляет собой ответы на вопросы типа что? где? когда? сколько? какой? Вопрос, на который существует только два варианта ответа — «да» или «нет» — назовем *двоичным*. Многие сложные вопросы можно заменить последовательностью двоичных вопросов. Например, вопрос об определении n-символьного двоичного слова может быть заменен последовательностью из n двоичных вопросов типа «на i-ой позиции слова находится единица?». Назовем двоичной сложностью вопроса минимальное число двоичных вопросов, его заменяющих.

Пусть нам дано некоторое множество, в котором каким-то образом выбран некоторый неизвестный нам элемент, и мы хотим узнать, какой именно элемент выбран. Такой вопрос называется вопросом об индивидуализации элемента множества.

Теорема 1.2. *Двоичная сложность вопроса об индивидуализации произвольного элемента n-элементного множества не меньше чем* $\log_2 n$.

Доказательство. Предположим, что мы придумали последовательность двоичных вопросов, позволяющих не более чем за m шагов определить загаданный элемент множества. Последовательность всех возможных вопросов и ответов образует двоичное дерево ответов, в узлах которого записаны вопросы, а на ребрах — ответы. В корне дерева записан первый вопрос; второй вопрос задается в зависимости от полученного ответа на первый вопрос, третий — в зависимости от ответа на второй и так далее. Высота дерева не превосходит m, поэтому количество терминальных узлов дерева не превосходит 2^m. Каждому терминальному узлу дерева соответствует элемент множества, ведь прекращаем задавать вопросы мы тогда, когда определяем загаданный элемент. Этот разгаданный элемент мы и ставим в соответствие терминальному узлу. Так как всякий элемент множества может быть загадан, то всякому элементу соответствует хотя бы один терминальный узел. Поэтому количество терминальных узлов дерева не может быть меньше количества элементов множества, откуда следует неравенство $n \leqslant 2^m$ или $m \geqslant \log_2 n$. □

С другой стороны, если занумеровать элементы множества двоичными числами, то для индивидуализации элемента достаточно узнать

его номер, а выяснение этого номера, как мы видели выше, имеет двоичную сложность, не превосходящую числа цифр в этом номере, то есть не превосходит $\log_2 n + 1$ и в точности совпадает с $\log_2 n$, если n является степенью двойки.

1.9 Количество информации по Хартли

Если принять информационную мощность двоичного вопроса за единицу (бит), то информационная мощность вопроса с n ответами естественно оценивается как $\log_2 n$ бит. Логарифмическая мера для количества информации была предложена и обоснована Д. Хартли в 1928 году.

З а д а ч и

1. Сколько информации вмещает трехзначное число?
2. Докажите, что вероятность сжатия двоичного файла (любым данным) архиватором хотя бы на один байт не превосходит 1/256.

Глава 2. Энтропия

2.1 Вывод формулы Шеннона из формулы Хартли

Согласно Хартли, количество информации, которое мы можем получить, узнав ответ на вопрос, допускающий n различных ответов, выражается величиной $\log_2 n$. Следующий шаг в развитии количественного измерения информативности дает формула Шеннона, учитывающая разные вероятности ответов.

Предположим, у нас был троичный файл F_0, в котором было поровну — по n — вхождений каждого из трех символов алфавита {0,1,2}. Предположим, с файлом случилось нечто, в результате чего вместо двоек всюду были записаны единицы, то есть второй и третий символы алфавита перестали различаться. Таким образом, у нас получился двоичный файл F_1 длины $3n$ с n нулями и $2n$ единицами. Сколько информации было потеряно?

Обозначим через $n(k)$ позицию k-го вхождения единицы в файл F_1, то есть такую позицию, на которой стоит k-я по счету единица (на предыдущих позициях стоят $k-1$ единиц и нули). Для полного восстановления утерянной информации достаточно предоставить двоичный файл F_2 длины $2n$, k-ая позиция которого дает разность между $n(k)$-ой позицией файлов F_0 и F_1. Таким образом, утерянная информация согласно Хартли равна $2n$ бит. Так как информационная емкость исходного файла оценивалась величиной $3n \log_2 3$, то логично предполагать, что информационная емкость двоичного файла, у которого $2n$ единиц и n нулей, выражается разностью $3n \log_2 3 - 2n$.

Эта оценка подтверждается асимптотическими вычислениями. А именно, количество слов с декомпозицией $n, 2n$, как было показано на предыдущей лекции, выражается числом $\frac{(3n)!}{n!(2n)!}$, двоичный логарифм которого асимптотически эквивалентен $\log_2((3n)^{3n}) - \log_2 n^n - \log_2(2n)^{2n} = 3n \log_2 3 + 3n \log_2 n - n \log_2 n - 2n - 2n \log_2 n = 3n \log_2 3 - 2n$.

2.2 Информативность двоичного слова с произвольной декомпозицией

Пусть теперь рассматриваемые двоичные слова состоят из nk нулей и mk единиц (т./,е. имеют декомпозицию nk,mk). Рассмотрим некоторый алфавит из $n+m$ символов: $a_1, \ldots, a_n, b_1, \ldots, b_m$. Произвольное слово из этого алфавита, в котором с одинаковой частотой встречаются все символы алфавита, как мы выяснили на прошлой лекции, асимптотически (при $k \to \infty$) имеет информационную емкость $(mk+nk)\log_2(m+n)$ бит. Если мы заменим первые n символов алфавита нулями, то мы потеряем $nk\log_2 n$ бит информации, а если заменим единицами последние m символов, то дополнительно потеряем $mk\log_2 m$ бит. При таком отождествлении мы получим произвольный двоичный файл с декомпозицией nk, mk Таким образом средняя информационная емкость символа двоичного слова с декомпозицией nk, mk выразится формулой:

$$\log_2(m+n) - \frac{n}{n+m}\log_2 n - \frac{m}{n+m}\log_2 m = \\ = -\left(\frac{n}{n+m}\log_2\frac{n}{m+n} + \frac{m}{n+m}\log_2\frac{m}{n+m}\right). \qquad (2.1)$$

2.3 Вывод формулы Шеннона

Пусть есть источник информации с m-символьным алфавитом и данной частотной характеристикой $f_1, f_2, \ldots f_m$, где f_i выражает частоту или *вероятность* появления i-го символа, так что все частоты неотрицательны и их сумма равна единице. Тогда информативность этого источника вычисляется по следующей формуле:

$$-f_1\log_2 f_1 - f_2\log_2 f_2 - \ldots - f_m\log_2 f_m. \qquad (2.2)$$

Эта формула — основная формула теории информации — называется *формулой Шеннона*.

Выведем эту формулу из формулы (2.1) индукцией по числу символов. При отождествлении двух последних символов алфавита мы получим распределение с частотами $f_1, \ldots, f_{m-2}, f_{m-1}+f_m$. Его инфор-

мативность по предположению индукции выражается формулой

$$-f_1\log_2 f_1 - f_2\log_2 f_2 - \ldots - f_{m-2}\log_2 f_{m-2} - \\ -(f_{m-1}+f_m)\log_2(f_{m-1}+f_m). \qquad (2.3)$$

Пусть $F_{m-1} = \frac{f_{m-1}}{f_{m-1}+f_m}$ и $F_m = \frac{f_m}{f_{m-1}+f_m}$ — относительные частоты пары отождествленных символов. По вышеприведенным рассуждениям, при этом отождествлении теряется $-F_{m-1}\log_2 F_{m-1} - F_m\log_2 F_m$ бит информации на каждый объединенный символ, а доля объединенных символов составляет $f_{m-1}+f_m$. Следовательно, в среднем на один символ теряется $-f_{m-1}\log_2 F_{m-1} - f_m\log_2 F_m$ бит. Поэтому для получения информативности m-символьного распределения нужно добавить только что полученное выражение к формуле (2.3). Как нетрудно проверить, полученная сумма дает формулу (2.2).

2.4 Асимптотические оценки

Оценим теперь среднюю информационную емкость символа слова с данной декомпозицией $n_1 + \ldots + n_m = N$ прямыми вычислениями.

Как мы выяснили на прошлой лекции, общее количество слов с данной декомпозицией выражается формулой

$$C(n_1,\ldots,n_m) = \frac{N!}{n_1!n_2!\cdot\ldots\cdot n_m!} \approx \frac{N^N}{n_1^{n_1}\cdot\ldots\cdot n_m^{n_m}}. \qquad (2.4)$$

Двоичный логарифм этого выражения, деленный на N, при $N\to\infty$ и при условии существования пределов $\frac{n_i}{N}\to p_i$, стремится к следующему выражению:

$$H(p_1,\ldots,p_n) = -p_1\log_2 p_1 - \ldots - p_m\log_2 p_m, \qquad (2.5)$$

называемому *энтропией* частотного распределения $p_1,\ldots,p_n$,

Выражение (2.5) согласуется с выведенной выше формулой (2.3).

2.5 Сжатие файла с данной декомпозицией

Пусть дан некоторый текстовый файл с алфавитом из m символов и пусть $f_1,\ldots,f_m$ — частоты вхождения символов. Как сильно можно сжать этот файл?

Теорема 2.1 (Шеннон). *Существует способ сжатия любого файла с частотными характеристиками $f_1, \ldots, f_m$, который обеспечивает сжатие, сколь угодно близкое к $|f_1 \log_2 f_1 + \ldots + f_m \log_2 f_m| = H(f_1, \ldots, f_m)$ бит на символ и не существует способа сжатия файлов с данными частотными характеристиками, обеспечивающего лучшее сжатие.*

Доказательство. Обозначим $H(f_1, \ldots, f_m) = H$. Как было посчитано выше, число файлов фиксированной длины N с данной частотной характеристикой приближенно равно 2^{NH}, что совпадает с общим количеством двоичных файлов длины NH. Поэтому не существует взаимнооднозначного отображения множества файлов длины N с данным частотным распределением в файлы длины меньшей, чем HN. Существование подобного отображения давало бы способ кодирования, сжимающий файлы так, что на символ в среднем тратится меньше H бит.

Чтобы кодировать символьные распределения с частотами $f_1, \ldots, f_m$ таким образом, чтобы на символ приходилось в среднем H бит информации, можно поступить следующим образом. Рассмотрим множество $S(N)$ всех последовательностей символов длины N с данным частотным распределением. Пусть H' — наименьшее натуральное число, для которого $|S(N)| < 2^{H'}$. На множестве $S(N)$ рассмотрим *лексикографический порядок*, то есть предполагая упорядоченным алфавит, мы считаем последовательность w_1 предшествующей последовательности w_2, если до позиции i включительно эти последовательности совпадают, а на $(i+1)$-й позиции в w_1 стоит символ алфавита, предшествующий символу, стоящему на $(i+1)$-й позиции в w_2.

В качестве кода последовательности w из $S(N)$ мы рассматриваем такое двоичное слово $c(w)$ длины H', у которого ровно столько же предшествующих ему двоичных слов длины H', сколько имеется последовательностей в $S(N)$, предшествующих w. Как мы выяснили выше, при большом N отношение H' к HN приблизительно равно единице. Поэтому среднее число бит на символ при таком кодировании с ростом N стремится к H. □

2.6 Вероятностный подход

Рассмотрим теперь слово бесконечной длины, то есть бесконечную последовательность символов алфавита, такую, что i-ый символ встре-

чается в последовательности с частотой p_i (то есть для i-го символа p_i является пределом частоты встречаемости в начальном отрезке последовательности при стремлении длины этого отрезка к бесконечности).

Теорию информации можно строить как теорию конечных или бесконечных слов. Первый подход реализует алгебраическая или комбинаторная теория информации, второй подход основан на теории вероятностей. Эти теории согласованы между собой асимптотически, то есть с ростом длины слова формулы алгебраической теории информации приближаются к формулам вероятностной теории информации. Хотя в практической жизни нам встречаются лишь конечные слова, формулы, применяемые на практике, относятся к теории бесконечных слов, потому что последние проще.

Следующая теорема позволяет доказать вероятностный аналог теоремы 2.1 для бесконечных последовательностей (формулировку и доказательство вероятностного аналога теоремы 2.1 мы оставляем читателю).

Теорема 2.2. *Пусть $M_\varepsilon(f_1, \dots, f_k)$ — множество слов длины n, частотное распределение которых отличается от распределения $f_1, \dots, f_k$ не более чем на ε. Тогда*

$$\lim_{\varepsilon \to 0, n \to \infty} \frac{\log_2 |M_\varepsilon(f_1, \dots, f_k)|}{nH(f_1, \dots, f_k)} = 1.$$

Доказательство. Обозначим через $H_\varepsilon(f_1 \dots f_k)$ максимум, а через $h_\varepsilon(f_1 \dots f_k)$ минимум энтропии среди всех распределений, ε-близких к $f_1, \dots, f_k$. Тогда, очевидно,

$$\lim_{\varepsilon \to 0} H_\varepsilon(f_1 \dots f_k) = \lim_{\varepsilon \to 0} h_\varepsilon(f_1 \dots f_k) = H(f_1 \dots f_k). \tag{2.6}$$

Количество разнообразных декомпозиций оценивается сверху величиной n^k. Поэтому логарифм количества всех слов с частотным распределением, ε-близким к данному, оценивается сверху суммой $n \log_2 H_\varepsilon(f_1, \dots, f_k) + k \log_2 n$, в которой второе слагаемое бесконечно мало по сравнению с первым при $n \to \infty$. □

2.7 Префиксный код

Рассмотрим задачу сжатия текстового файла. Различные буквы встречаются в тексте с существенно разными частотами. Поэтому следует

кодировать буквы таким образом, чтобы те, которые встречаются чаще, имели более короткие коды, чем те, которые встречаются реже. Если код неравномерный, то возникает проблема — как понять, где кончился код одного символа и начался код другого. Эта проблема может быть решена, если код обладает следующим свойством: код одного символа не может быть началом кода другого символа. Код, удовлетворяющий этому условию, называется *префиксным*. Например, пусть исходный текстовый файл содержит 64 буквы «а», 32 буквы «б», 16 букв «в», 16 букв «г». Поскольку весь алфавит файла состоит из четырех символов, эти символы можно было бы закодировать парой бит на символ и весь файл сжать до 256 бит. Если же мы будем кодировать «а» нулевым битом 0, кодировать «б» последовательностью 10 и кодировать «в» и «г» соответственно последовательностями 110 и 111, то файл сожмется до размера в $64 + 2 \cdot 32 + 3 \cdot 16 + 3 \cdot 16 = 224$ бита. Правда, нам дополнительно придется где-то поместить кодовую таблицу, указывающую, какие коды каким символам соответствуют. Код 0, 10, 110, 111 является префиксным. Если представить шифрованный файл в виде двоичного массива $b[i], 1 \leqslant i \leqslant 224$, то дешифрация выполняется следующим алгоритмом, выдающим на выходе байтовый массив $d[i]$ исходного текста:

```
i:=1;j=0; :next j:=j+1; if i<225 then begin if b[i]=0 then
        begin d[j]:="a"; i:=i+1; goto next end;
        else if b[i+1]=0  begin d[j]:="б"; i:=i+2; goto next end;
             else begin   if b[i+2]=0 then d[j]:="в";
                          else  d[j]:="г"; i:=i+3; goto next
        end;
end
```

Для четырехсимвольного текста с частотами символов $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}$ рассмотренный способ кодирования является оптимальным.

2.8 Алгоритм Хаффмана

В то же время существуют алгоритмы сжатия, которые имеют близкую к теоретически оптимальной эффективность. Одним из наиболее применимых алгоритмов такого рода является алгоритм Хаффмана. Код Хаффмана — это двоичный префиксный код.

Кодирование по Хаффману определяется рекурсивно. Если в тексте имеется всего два символа, то один из них кодируется нулем, а другой —

единицей. Предположим, что кодирование по Хаффману уже определено для n-символьных распределений и нам дано $(n+1)$-символьное. Отождествим два самых редких символа распределения и построим код Хаффмана для получившегося n-символьного распределения. Если $b_1 \dots b_k$ — код Хаффмана для склеенного символа, то, не изменяя кодов прочих символов, определим коды слившихся символов как $b_1 \dots b_k 0$ и $b_1 \dots b_k 1$. Получится префиксный код.

П р и м е р .

Пусть алфавит состоит из пяти символов a, b, c, d, e с частотами $0.37\,(a)$, $0.22\,(b)$, $0.16\,(c)$, $0.14\,(d)$, $0.11\,(e)$. Объединяя d и e, получаем частоты $0.37\,(a)$, $0.25\,(de)$, $0.22\,(b)$, $0.16(c)$. На следующем шаге объединяются символы b и c: $0.38\,(bc)$, $0.37\,(a)$, $0.25\,(de)$, и, наконец, последнее объединение дает частоты $0.62\,(ade)$, $0.38\,(bc)$. Сопоставим объединенному символу ade код 0, bc — код 1. Расщепим символ ade на a и de с кодами 00 и 01. Символ bc расщепляется на b и c с кодами 10 и 11. Наконец, расщепив de, получим для исходного алфавита коды $00\,(a)$, $10\,(b)$, $11\,(c)$, $010\,(d)$, $011\,(e)$.

Можно доказать, что алгоритм Хаффмана дает наилучший возможный двоичный префиксный код.

2.9 Свойства функции энтропии

Функция энтропии (2.5) обладает весьма замечательными свойствами. Ниже мы рассмотрим условия, при которых эта функция достигает экстремальных значений.

Теорема 2.3. *Энтропия распределения $p_1, \dots p_n$ равна нулю в том и только том случае, когда все p_i за единственным исключением равны нулю.*

Доказательство. Мы считаем, что $0 \log_2 0 = 0$. Это соглашение оправдано, например, тем соображением, что $\lim\limits_{x \to 0} x \log_2 x = 0$. Поэтому $H(1, 0, \dots, 0) = 0$. Таким образом, мы можем считать, что $-p_i \log_2 p_i \geqslant 0$ для всех i. Если же некоторое p_i отлично как от нуля, так и единицы, то $-p_i \log_2 p_i$ строго больше нуля, откуда $H(p_1, \dots, p_n) > 0$. □

Теорема 2.4. *Равномерное распределение имеет наибольшую энтропию среди всех распределений с данным числом исходов.*

Доказательство. Доказательство ведется индукцией по числу исходов k. Для $k = 2$ результат был получен выше. Пусть утверждение доказано для k. Рассмотрим формулу энтропии для $k+1$ исхода:

$$\begin{aligned} H(p, q_1, \dots q_k) &= -p\log_2 p - q_1\log_2 q_1 - \dots - q_k\log_2 q_k = \\ &= -p\log_2 p - (1-p)\frac{q_1}{1-p}\left(\log_2\frac{q_1}{1-p} + \log_2(1-p)\right) - \dots - \\ &\quad - (1-p)\frac{q_k}{1-p}\left(\log_2\frac{q_k}{1-p} + \log_2(1-p)\right) = \\ &= -p\log_2 p + (1-p)H\left(\frac{q_1}{1-p}, \dots, \frac{q_k}{1-p}\right) - \\ &\quad - (1-p)\log_2(1-p). \end{aligned}$$

Так как по предположению индукции $H\left(\frac{q_1}{1-p}, \dots, \frac{q_k}{1-p}\right) \leqslant \log_2 k$, получим следующее неравенство:

$$H(p, q_1 \dots q_k) \leqslant -p\log_2 p - (1-p)\log_2(1-p) + (1-p)\log_2 k. \qquad (2.7)$$

Найдем максимум правой части неравенства. Для этого продифференцируем ее по p. Получим:

$$-\log_2 p - \frac{p}{p\ln 2} + \log_2(1-p) + \frac{1-p}{(1-p)\ln 2} - \log_2 k = \log_2\frac{1-p}{kp}.$$

Производная равна нулю при $kp = 1-p$, то есть при $p = \frac{1}{k+1}$. При этом производная положительна при $p < \frac{1}{k+1}$ и отрицательна при $p > \frac{1}{k+1}$. Следовательно, наибольшее значение функция в правой части (2.7) принимает при $p = \frac{1}{k+1}$. И это значение, как нетрудно убедиться, равно $\log_2(k+1)$, то есть равно энтропии равномерного распределения. □

2.10 Энтропия как мера неопределенности

Если источник информации выдает нам последовательность символов с частотами $p_1, \dots, p_n$, то энтропия этого частотного распределения представляет собой количественную меру неопределенности источника.

Нулевая энтропия означает полную детерминированность источника. Чем больше вариантов представляет данное распределение, тем выше энтропия. Максимум энтропии достигается при равновероятных символах и этот максимум совпадает с информативностью символа по Хартли.

2.11 Отгадывание слов: вариант 1

Рассмотрим игру в отгадывание слов. Нам дан некоторый словарь, любое слово из которого может быть загадано. Нам известны длина и, возможно, несколько букв слова, стоящих на определенных местах. Мы можем попросить открыть любую из букв, прежде чем попытаемся угадать слово. Допустим, что словарь возможных слов загружен у нас в компьютере и мы можем получать ответы на вопросы типа «Сколько имеется в словаре слов данной длины с данными буквами на данных местах?» Какую по порядку букву нам в этом случае следует просить открыть?

Обозначим через W множество допустимых слов, то есть таких слов данного словаря, которые соответствуют известным нам параметрам (то есть имеют данную длину и у которых на определенных местах стоят известные буквы). Обозначим через A алфавит языка (например, кириллический алфавит). Для любого $s \in A$ и любого натурального k обозначим через $s_W(k)$ количество слов из W, имеющих на k-ом месте букву s. Совокупность чисел $\{s_W(k) \mid s \in A, k \in \mathbb{N}\}$ назовем *буквоместным распределением* словаря W. *Частотой* буквы s на позиции k назовем отношение $\frac{s_W(k)}{|W|}$, где $|W|$ — число слов в W. При любом натуральном k совокупность частот $\left\{\frac{s_W(k)}{|W|} \mid s \in A\right\}$ назовем частотным распределением k-ой позиции словаря W. Частотные распределения позиций — это единственная информация, на основе которой мы должны сделать выбор.

Предположим, например, что на некоторой позиции во всех словах словаря стоит одна и та же буква. Ясно, что в этом случае бессмысленно просить открыть эту позицию, ведь открывая ее мы не получаем новой информации — мы заранее знаем, что там стоит.

Если же на k-ой позиции встречаются m букв, то чем больше m — тем больше информации несет позиция. Действительно, посчитаем вероятность угадывания слова в случае, когда мы открываем k-ую позицию. Пусть на k-ой позиции встречаются буквы $s_1, \ldots, s_m$ с вероятностями $p_1, \ldots, p_m$. Если на k-ой позиции оказалась буква s_i, то у нас осталось $p_i W$ вариантов. Следовательно, при выборе наудачу наш шанс теперь угадать слово равен $1/p_i W$, а вероятность попасть в эту ситуацию равна p_i. Таким образом, суммарная вероятность угадывания слова составляет $\sum\limits_{i=1}^{m} p_i \frac{1}{p_i W} = \frac{m}{W}$. То есть больше всего шансов на вы-

игрыш нам дает открытие самой информативной по Хартли позиции, каковой является позиция, на которой встречается наибольшее число различных букв независимо от их частотного распределения.

2.12 Отгадывание слов: вариант 2

Теперь цель игры заключается в отгадывании загаданного слова за наименьшее число ходов, а один ход игры заключается в том, что нам сообщают какая буква стоит на запрошенной нами позиции. Оказывается, в этом случае оптимальным будет выбор позиции с наибольшей энтропией букво-местного распределения. Попробуйте обосновать этот выбор.

З а д а ч и

1. Какое из распределений — $\left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\}$ или $\left\{\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{2}{5}\right\}$ — имеет большую энтропию?
2. Среди распределений с четырьмя исходами, частоты которых f_{11}, f_{12}, f_{21}, f_{22} удовлетворяют соотношениям $f_{11}+f_{12}=p$, $f_{11}+f_{21}=q$, найти распределение с наибольшей энтропией.

Глава 3. Информационная зависимость

3.1 Энтропия и информация

Энтропия измеряет количество имеющихся возможностей, то есть является мерой неопределенности ситуации. Информация — это снятая неопределенность, то есть разность между той неопределенностью (энтропией), которая у нас была до получения информации и возникла после ее получения.

Количество информации, содержащееся в источнике информации, совпадает с энтропией этого источника. Поэтому в этой лекции термины энтропия и информация используются как синонимы.

3.2 Совместное распределение

Пусть имеется пара источников информации, которые согласованно выдают последовательности символов, первый — из алфавита A, второй — из алфавита B. В качестве примера согласования источников можно рассмотреть следующий: первый источник — это первая буква в слове русского языка, а второй — вторая буква. В этом случае алфавиты источников одинаковы. Согласованность этих источников заключается в том, что они выдают буквы одного и того же слова. Другой вариант согласования источников — синхронизация: оба источника выдают сообщения в одни и те же моменты времени или дают информацию об одном и том же явлении.

Всякий способ согласования позволяет рассматривать пару источников как один составной источник информации, алфавитом которого служит произведение $A \times B$. Договоримся здесь и далее обозначать i-ую букву алфавита A через $A[i]$.

Частотная характеристика составного источника называется *совместным распределением* пары источников. Мы будем изображать сов-

местное распределение в виде матрицы (двумерного массива) частот f_{ij}, где f_{ij} есть частота, с которой встречается пара $(A[i], B[j])$, состоящая из i-го символа алфавита A и j-го символа алфавита B.

3.3 Условная и взаимная информация

Пусть H_1 и H_2 представляют собой энтропии двух согласованных источников, а H_{12} — энтропия их совместного распределения. Энтропия совместного распределения, как мы покажем ниже, удовлетворяет следующим неравенствам:

$$\max\{H_1, H_2\} \leqslant H_{12} \leqslant H_1 + H_2. \tag{3.1}$$

Разность $H_{12} - H_1$ представляет собой количество дополнительной информации, которую второй источник несет по отношению к первому. Эта разность называется *условной информацией* первого источника относительно второго. Следовательно, логично ожидать, что разность $H_2 - (H_{12} - H_1) = H_1 + H_2 - H_{12}$ указывает на количество информации второго источника, которая уже содержалась в первом. Поэтому величина $H_1 + H_2 - H_{12}$ называется *взаимной информацией* пары источников. Таким образом, вся информация пары источников делится на три части: $H_{12} - H_2$ — условную информацию первого источника относительно второго, $H_{12} - H_1$ — условную информацию второго источника относительно первого, и $H_1 + H_2 - H_{12}$ — взаимную информацию первого и второго источников.

Таким образом, чтобы ответить на вопрос «Сколько информации цвет волос человека несет о цвете его глаз?», нужно посчитать энтропию цвета волос H_1, энтропию цвета глаз H_2 и энтропию пары «цвет волос–цвет глаз» H_{12}. Тогда ответ на вопрос даст величина $H_1 + H_2 - H_{12}$.

3.4 Функциональная зависимость

Если условная информация одного источника относительно другого равна нулю, то, зная содержание второго источника, можно однозначно определить содержание первого.

Будем говорить что источник информации β с алфавитом B *функционально* зависит от согласованного с ним источника информации α с алфавитом A, если имеется функция $f\colon A \to B$, такая, что всякий раз,

когда α порождает некоторый символ $a \in A$, источник β порождает $f(a) \in B$.

Лемма 3.1. *Если $p = p_1 + \ldots + p_n$, $p_i \geqslant 0$, то выполняется неравенство $p \log_2 p \geqslant p_1 \log_2 p_1 + \ldots + p_n \log_2 p_n$, причем равенство достигается только в случае, когда все p_i за единственным исключением равны нулю.*

Доказательство. Рассмотрим разность $p \log_2 p - p_1 \log_2 p_1 - \ldots - p_n \log_2 p_n$. Подставив $p_1 + \ldots + p_n$ вместо p, запишем эту разность в виде

$$\sum_{i=1}^{n} p_i (\log_2 p - \log_2 p_i).$$

$p \geqslant p_i$ при любом i, следовательно, для любого i $p_i(\log_2 p - \log_2 p_i) \geqslant 0$ и их сумма также больше нуля. Сумма неотрицательных чисел равна нулю в том и только том случае, если все слагаемые равны нулю. А равенство $p_i(\log_2 p - \log_2 p_i) = 0$ возможно лишь если $p_i = 0$ или $p_i = p$. □

Теорема 3.1. *Условная энтропия источника β относительно согласованного с ним источника α неотрицательна и равна нулю в том и только том случае, когда β функционально зависит от α*

Доказательство. Пусть $p_1, \ldots, p_m$ — частотное распределение источника α, $q_1, \ldots, q_n$ — частотное распределение β, $\{r_{ij}\}$ — их совместное распределение. Тогда $p_i = \sum_j r_{ij}$ и $q_j = \sum_i r_{ij}$ при любых i и j.

Если β функционально выражается через α, то $r_{ij} \neq 0$ в том и только том случае, когда $f(A[i]) = B[j]$ и в этом случае $p_i = r_{ij}$. Поэтому сумма, представляющая H_{12}, имеет те же ненулевые слагаемые, что и H_1. Следовательно, $H_{12} - H_1 = 0$.

Обратно, разность $H_{12} - H_1$ представляется в виде суммы

$$\sum_i \left(p_i \log_2 p_i - \sum_j r_{ij} \log_2 r_{ij} \right).$$

Все слагаемые этой суммы неотрицательны в силу леммы 3.1. Поэтому равенство нулю этой суммы влечет равенство нулю всех слагаемых. А равенство нулю слагаемого $p_i \log_2 p_i - \sum_j r_{ij} \log_2 r_{ij}$ в силу леммы

3.1 означает, что при данном i все r_{ij} за единственным исключением равны нулю. Обозначим это единственное исключение через $f(i)$. Так мы определяем функцию $f\colon A \to B$, реализующую функциональную зависимость источников. □

3.5 Критерий независимости

Будем говорить, что матрица $\{f_{ij}\}_{i\leqslant m, j\leqslant n}$ является *тензорным произведением* строк $p_1, \ldots, p_m$ и $q_1, \ldots, q_n$, если $f_{ij} = p_i q_j$ при любых i, j.

Два источника информации называются *независимыми*, если их совместное распределение является тензорным произведением их распределений.

Теорема 3.2 (Критерий независимости)**.** *Энтропия совместного распределения двух источников информации не превосходит суммы энтропий этих источников и совпадает с ней в том и только том случае, когда эти источники информации независимы.*

Другими словами, независимость источников равносильна тому, что их взаимная информация равна нулю.

Для доказательства этой теоремы нам потребуется несколько определений и лемма.

Для двумерного распределения p_{ij} определим его построчную (постолбцовую) *свертку* как одномерное распределение $P_i = \sum_j p_{ij}$ (соответственно, $P^j = \sum_i p_{ij}$). *Строчное подразделение* двумерного распределения p_{ij} определяется как распределение, у которого i-ая строка $\{p_{ij}\}_{j\leqslant n}$ заменяется на k одинаковых строк $\{p_{ij}/k\}_{j\leqslant n}$.

Лемма 3.2. *Если два двумерных распределения имеют одинаковые построчные свертки, то при построчном подразделении их энтропии изменяются на одно и то же число.*

Доказательство. Пусть первая строка подразделяется на k строк. Пусть

$$S = \sum_{j=1}^{n} \sum_{i=2}^{n} p_{ij} \log_2 p_{ij}.$$

Тогда энтропия исходного распределения есть $S + \sum_i p_{1i} \log_2 p_{1i}$, а энтропия подразделенного распределения есть $S + k \sum_i \frac{p_{1i}}{k} \log_2 \frac{p_{1i}}{k}$. Изменение энтропии равно $\sum_i p_{1i} \log_2 k$, то есть выражается через построчную свертку. □

Перейдем теперь к доказательству критерия независимости.

Пусть p_i и q_j — распределения двух источников, r_{ij} — их совместное распределение. Докажем, что энтропия совместного распределения всегда не превосходит энтропии тензорного произведения их распределений и совпадает с последней только в случае независимости. Для этого мы сделаем сначала построчные, а потом постолбцовые подразделения совместного распределения и тензорного поизведения, так, чтобы построчные и постолбцовые свертки у полученных распределений были бы равномерными распределениями. В силу доказанной леммы при этом совместное распределение и тензорное произведение изменятся на одинаковую величину. Далее тензорное произведение превратится в тензорное произведение равномерных распределений и будет равномерным распределением, поэтому его энтропия будет больше, чем энтропия подразделенного совместного распределения. □

3.6 Относительное кодирование

Доказанная на лекции 1.9 теорема дает теоретическую нижнюю оценку для сжатия файла, основанную на неравномерности символьного распределения. Эта оценка, однако, на практике легко превосходится за счет явления *зависимости* следующего символа от предыдущего. Так, например, если на какой-то позиции слова стоит гласная буква, то вероятность появления согласной буквы на следующей позиции возрастает.

Новая идея в сжатии информации заключается в следующем: мы по-прежнему работаем с префиксным кодом, но кодирование происходит по-разному в зависимости от того, какой символ был обнаружен на предыдущей позиции. То есть мы можем поступить следующим образом: для каждого символа рассматриваемого алфавита мы вычисляем распределение символов, непосредственно за ним следующих, и каждое такое распределение мы кодируем с помощью алгоритма Хаффмана. Кодирование файла после этого производится так: каждый символ последовательности, за исключением первого, кодируется кодом Хаффмана, соответствующим предыдущему символу. Как кодировать первый символ — несущественно, поскольку это мало влияет на коэффициент сжатия. Например, первый символ последовательности можно кодировать кодом, порожденным первым символом алфавита.

Вернемся к задаче из первой лекции — о файле, содержащем информацию о поле всех граждан России. Почему можно сжать этот файл?

Основная причина в том, что в русском языке по фамилии в большинстве случаев можно определить пол. Следовательно, люди, имеющие одинаковые фамилии, с большой вероятностью имеют и одинаковый пол. В списке, составленном по алфавиту, однофамильцы идут подряд. Поэтому сохранение пола при переходе к следующей фамилии гораздо более вероятно, чем изменение.

3.7 Случайные последовательности

Можно рассматривать зависимость символа не только от предыдущего, но и от пары предыдущих и т. д. Наличие таких зависимостей также позволяет сжать файл. Случайная последовательность символов — такая, которая не имеет зависимостей. Сжать её нельзя.

Рассмотренные выше понятия не применимы непосредственно к источникам с числовой информацией, потому что у таких источников бесконечен алфавит. Для того, чтобы применить развитую теорию к числовым последовательностям, можно применить прием, называемый *квантованием*.

Пусть числовая последовательность принимает значения на отрезке $[0, 1]$. Квантование отрезка $[0, 1]$ осуществляется посредством неубывающей функции $f\colon [0, 1] \to \{1, 2, \dots n\}$. (Например, можно квантовать $[0, 1]$ с помощью отображения $f(x) = [nx]$.) Любой источник числовых данных теперь может квантоваться посредством f. Квантовать числовой источник следует таким образом, чтобы проквантованная величина сохранила максимум информации об исходной последовательности. Для этого нужно выбрать квантующую функцию так, чтобы символы $0, 1, \dots (n-1)$ имели равные частоты. Такое квантование называется равномерным. Числовая последовательность называется случайной, если случайными являются все символьные последовательности, получающиеся из нее квантованием.

3.8 Суперпозиция неопределенностей

Пусть даны две случайные последовательности $\{a_1, a_2, \dots, a_k, \dots\}$ и $\{b_1, b_2, \dots, b_k, \dots\}$ символов некоторых конечных алфавитов, первая с энтропией H_1, вторая — с энтропией H_2. Определим теперь смесь этих двух последовательностей следующим образом: k-ый элемент смеси равен a_k или b_k, причем с вероятностью p выбирается a_k и с вероятностью

$1-p$ выбирается b_k. Определим энтропию H_{12} смешанной последовательности.

Пусть p_i — частота встречаемости (вероятность) i-го символа алфавита первой последовательности ($i=1,\dots,m$), а q_i — частота встречаемости i-го символа алфавита второй последовательности ($i=1,\dots,n$). Тогда $H_1 = -\sum\limits_{i=1}^{m} p_i \log_2 p_i$ и $H_2 = -\sum\limits_{i=1}^{n} q_i \log_2 q_i$. Смешанная последовательность содержит i-ый символ первого алфавита с частотой pp_i и содержит i-ый символ второго алфавита с частотой $(1-p)q_i$. Поэтому энтропия смешанной последовательности равна

$$\begin{aligned}
-H_{12} &= \sum_{i=1}^{m} pp_i \log_2(pp_i) + \sum_{i=1}^{n}(1-p)q_i \log_2((1-p)p_i) = \\
&= p\sum_{i=1}^{m} p_i(\log_2 p_i + \log_2 p) + \\
&\qquad + (1-p)\sum_{i=1}^{n} q_i(\log_2 q_i + \log_2(1-p) = \\
&= -pH_1 - (1-p)H_2 + p\log_2 p + (1-p)\log_2(1-p).
\end{aligned}$$

Таким образом, энтропия смешанной последовательности представляет собой сумму средневзвешенной энтропии $pH_1 + (1-p)H_2$ этих последовательностей и смешивающей энтропии $p\log_2 p + (1-p)\log_2(1-p)$.

Более общо: пусть у нас есть несколько независимых последовательностей (источников информации) $\{w_1\}$, $\{w_2\}$,..., $\{w_n\}$ с энтропиями $H_1,\dots,H_n$ и пусть дана независимая смешивающая последовательность $\{n_k\}$ с энтропией H_0, составленная из натуральных чисел диапазона $[1,n]$. Построим смешанную последовательность следующим образом: на k-м месте в смешанной последовательности стоит k-й элемент последовательности $\{w_{n_k}\}$. Тогда смешанная последовательность имеет энтропию

$$H_0 + \sum_{i=1}^{n} p_i H_i, \tag{3.2}$$

где p_i — частота встречаемости i в смешивающей последовательности.

Задачи

1. Рассмотрим следующий карточный фокус: колода трижды делится на три части, каждый раз указывается та часть колоды, в которой находится загаданная карта. Каково наибольшее число карт в колоде, чтобы можно было «честно» определить загаданную карту?
2. Пусть вероятность того, что следующий по алфавиту житель России имеет тот же пол, что и предыдущий, равна $\frac{3}{4}$. Насколько можно сжать файл пола населения России в этом случае?
3. Во сколько раз можно сжать файл, представляющий собой последовательность из трех символов a, b, c, если известно, что частота появления символа a равна $\frac{1}{2}$ и после a в половине случаев идет b, а в половине c?
4. Насколько можно сжать трехсимвольный файл, если в нем нет двух подряд идущих одинаковых символов?
5. В игре «Поле чудес» вам выпала возможность открыть сразу две буквы слова. Какую пару позиций следует выбрать?

Задачи к главам 2 и 3

Задача 1. Рассматриваются трехзначные числа с неубывающими цифрами.

1. Какая из цифр (первая,вторая,третья) несет больше информации о таком числе?
2. Сколько информации первая цифра такого числа несет о его второй и третьей цифрах?
3. Зависит ли первая цифра числа от двух последних?

Задача 2. Рассматриваются трехзначные числа с суммой цифр больше десяти.

1. Какая из цифр (первая, вторая или третья) несет больше информации о таком числе?
2. Сколько информации первая цифра такого числа несет о его второй и третьей цифрах?
3. Зависит ли первая цифра числа от двух последних?

Задача 3. Рассматриваются все трехзначные числа.

1. Сколько информации несет о числе сумма первой и второй, второй и третьей цифр?
2. Сколько информации сумма первой и второй цифр несет о сумме второй и третьей?
3. Зависит ли сумма первой и третьей цифр от суммы первой и второй цифр?

Задача 4. Рассматриваются трехзначные числа, у которых совпадают две цифры.

1. Сколько информации несет о таком числе разность первой и второй, второй и третьей цифр ?

2. Сколько информации разность первой и второй цифр несет о разности первой и третьей цифр?
3. Зависит ли третья цифра от разности первой и второй цифр?

Задача 5. Рассматриваются трехзначные числа с нечетными цифрами.

1. Сколько информации о числе несет сумма квадратов его цифр?
2. Сколько информации сумма цифр несет о сумме квадратов цифр?
3. Зависит ли сумма квадратов цифр от первой цифры?

Задача 6. Рассматриваются все трехзначные числа.

1. Сколько информации несет разность между наименьшей и наибольшей цифрами числа?
2. Какая цифра (первая, вторая или третья) несет больше информации о разности между наименьшей и наибольшей цифрами?
3. Зависит ли разность наименьшей и наибольшей цифр от значения наименьшей цифры?

Задача 7. Рассматриваются все трехзначные числа.

1. Сколько информации несет о числе произведение первой и второй, второй и третьей цифр?
2. Сколько информации произведение первой и второй цифр несет о сумме второй и третьей цифр?
3. Зависит ли третья цифра от произведения первых двух цифр?

Задача 8. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несут его наибольшая и наименьшая цифры?
2. Сколько информации наибольшая цифра числа содержит о его наименьшей цифре?
3. Зависит ли значение наибольшей цифры числа от разности первой и второй цифр?

Задача 9. Рассматриваются трехзначные числа, все цифры которых различны.

1. Сколько информации о числе дает знание цифр, из которых оно состоит?
2. Сколько информации о второй цифре такого числа дает знание суммы первой и третьей его цифр?
3. Зависит ли средняя по величине цифра такого числа от средней по позиции (т. е. второй) цифры?

Задача 10. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет сумма его нечетных цифр?
2. Сколько информации сумма нечетных цифр несет о сумме всех цифр?
3. Зависит ли сумма четных цифр от суммы нечетных цифр?

Задача 11. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет остаток от деления на 7?
2. Сколько информации остаток от деления на 7 всего числа несет об остатке от деления на 7 суммы его цифр?
3. Зависит ли остаток от деления на 7 от остатка от деления на 13?

Задача 12. Рассматриваются пятизначные числа, являющиеся полными квадратами.

1. Стоящая на какой позиции цифра наиболее информативна?
2. Сколько информации сумма первых двух цифр несет о сумме последних двух цифр?
3. Зависит ли третья цифра от суммы первых двух?

Задача 13. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет сумма цифр его квадрата, его куба?
2. Сколько информации о сумме цифр куба несет сумма цифр квадрата числа?
3. Зависит ли первая цифра квадрата числа от его суммы цифр?

Задача 14. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет остаток от деления на 7 суммы цифр его квадрата?
2. Сколько информации об остатке от деления на 7 суммы цифр куба числа несет остаток от деления на 7 суммы цифр его квадрата?
3. Зависит ли остаток от деления на 7 суммы цифр квадрата числа от суммы цифр куба числа?

Задача 15. Рассматриваются пятизначные числа с возрастающими цифрами.

1. Стоящая на какой позиции цифра несет больше информации о числе?
2. Сколько информации несет вторая цифра о четвертой?

3. Зависит ли разность первой и второй цифр от значения первой цифры?

Задача 16. Рассматриваются трехзначные числа, у которых ровно одна цифра четна.

1. Сколько информации о таком числе несет произведение его цифр?
2. Сколько информации произведение двух первых цифр числа несет о четной цифре?
3. Зависит ли произведение нечетных цифр от четной?

Задача 17. Рассматриваются все трехзначные числа, у которых средняя цифра является наибольшей.

1. Сколько информации о таком числе несет его первая цифра?
2. Сколько информации первая и третья цифра несут о второй цифре?
3. Зависит ли первая цифра от третьей?

Задача 18. Рассматриваются четырехзначные числа, у которых первая цифра равна последней.

1. Какая цифра несет больше информации о таком числе, первая или вторая?
2. Сколько информации сумма цифр такого числа несет о его первой цифре?
3. Зависит ли вторая цифра от первой?

Задача 19. Рассматриваются все трехзначные числа, не содержащие нулей.

1. Сколько информации несет отношение первой цифры ко второй?
2. Сколько информации отношение первой цифры ко второй несет об отношении наибольшей цифры к наименьшей?
3. Зависит ли отношение первой цифры ко второй от значения первой цифры?

Задача 20. Рассматриваются трехзначные числа с суммой квадратов цифр, не превосходящей 100.

1. Разность какой пары цифр несет больше информации о таком числе?
2. Сколько информации разность первой и второй цифр несет о сумме квадратов цифр такого числа?
3. Зависит ли третья цифра такого числа от первой?

Задача 21. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет сумма квадратов цифр?
2. Сколько информации несет сумма квадратов цифр о сумме цифр?
3. Зависит ли сумма квадратов цифр от разности первой и второй цифр?

Задача 22. Рассматриваются все четырехзначные числа, у которых сумма первых двух цифр равна сумме последних двух цифр.

1. Сумма какой пары цифр несет больше всего информации о числе?
2. Сколько информации о сумме второй и четвертой цифр несет сумма первой и третьей?
3. Зависит ли сумма первых двух цифр от произведения последних двух цифр?

Задача 23. Рассматриваются четырехзначные числа с убывающими цифрами.

1. Разность между какой парой цифр несет больше всего информации о таком числе?
2. Сколько информации о разности первой и четвертой цифр несет разность между второй и третьей?
3. Зависит ли разность первой и второй цифр от разности третьей и четвертой?

Задача 24. Рассматриваются четырехзначные числа, у которых по крайней мере две цифры совпадают.

1. Сколько информации о таком числе несет знание первых трех цифр?
2. Сколько информации о последней паре цифр несет первая пара цифр числа?
3. Зависит ли четвертая цифра такого числа от первой?

Задача 25. Рассматриваются все четырехзначные числа, у которых сумма цифр делится на 7.

1. Сколько информации о таком числе дает остаток от деления суммы цифр на 3?
2. Сколько информации об остатке от деления суммы цифр на 3 дает остаток от деления суммы цифр на 5?
3. Зависит ли остаток от деления на 7 суммы первых двух цифр числа от остатка от деления на 3 суммы последних двух цифр числа?

Задача 26. Рассматриваются десятизначные числа, состоящие из цифр 1 и 2.

1. Сколько информации о таком числе несет знание, что оно содержит последовательность цифр вида 1122?
2. Сколько информации о первой цифре дает знание количества единиц в числе?
3. Зависит ли количество единиц во второй половине числа от количества единиц в первой его половине?

Задача 27. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет знание о четности/нечетности его цифр?
2. Сколько информации сумма четных цифр несет о сумме нечетных?
3. Зависит ли четность первой цифры числа от четности суммы цифр?

Задача 28. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет знание того, что первая цифра больше второй?
2. Сколько информации соотношение (больше, меньше, равно) между первой и второй цифрами несет о соотношении между второй и третьей?
3. Зависит ли соотношение между первой и второй цифрами от значения наибольшей цифры числа?

Задача 29. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет знание того, что на первом месте стоит наибольшая цифра?
2. Сколько информации об остатке от деления на 3 наименьшей цифры числа содержит информация об остатке от деления на 7 его наибольшей цифры?
3. Зависит ли остаток от деления на 3 первой цифры от остатка от ее деления на 7?

Задача 30. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет сумма кубов его цифр?
2. Сколько информации сумма кубов цифр несет о сумме квадратов цифр?

3. Зависит ли сумма кубов цифр от произведения цифр?

Задача 31. Рассматриваются все трехзначные числа.

1. Что несет больше информации о числе: сумма цифр или квадрат суммы цифр?
2. Сколько информации о сумме квадратов цифр несет квадрат суммы цифр?
3. Зависит ли разность квадрата суммы цифр и суммы квадратов цифр от первой цифры?

Задача 32. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет сумма попарных произведений его цифр?
2. Сколько информации о сумме попарных произведений цифр несет сумма цифр числа?
3. Зависит ли сумма попарных произведений цифр числа от соотношения (больше, меньше или равно) между первой и второй цифрами?

Задача 33. Рассматриваются трехзначные числа без нулей.

1. Сколько информации о числе несет отношение наибольшей и наименьшей цифр?
2. Сколько информации об отношении первой цифры ко второй несет отношение наибольшей цифры к наименьшей?
3. Зависит ли отношение первой цифры ко второй от отношения второй к третьей?

Задача 34. Рассматриваются все трехзначные числа.

1. Что несет больше информации о числе: знание остатков всех его цифр от деления на 3 или знание первой цифры?
2. Сколько информации об остатке от деления на 5 суммы цифр числа несет знание остатков от деления на 3 всех его цифр?
3. Зависит ли остаток от деления на 5 суммы цифр от остатка от деления на 3 суммы цифр?

Задача 35. Рассматриваются все пятизначные числа, у которых по крайней мере три цифры совпадают.

1. Сколько информации несет одна цифра такого числа?
2. Сколько информации первая цифра такого числа несет о значении самой часто встречающейся цифры?

3. Зависит ли первая цифра такого числа от второй цифры?

Задача 36. Рассматриваются четырехзначные числа, у которых первая цифра больше второй, а третья больше четвертой.

1. Какая цифра несет больше всего информации о таком числе?
2. Сколько информации пара из первой и третьей цифр несет о паре из второй и четвертой цифр?
3. Зависит ли первая цифра от третьей?

Задача 37. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет сумма кубов его цифр?
2. Сколько информации сумма кубов цифр содержит о наибольшей цифре числа?
3. Зависит ли сумма квадратов цифр от суммы кубов цифр?

Задача 38. Рассматриваются все трехзначные числа. *Вариацией* числа xyz называется сумма $|x-y|+|y-z|$.

1. Сколько информации о числе несет его вариация?
2. Сколько информации вариация числа несет о сумме его цифр?
3. Зависит ли вариация числа от его первой цифры?

Задача 39. Рассматриваются все трехзначные числа.

1. Сколько информации о числе несет сумма первой, удвоенной второй и утроенной третьей цифр?
2. Сколько информации вышеописанная сумма несет о сумме квадратов цифр числа?
3. Зависит ли разность удвоенной второй цифры и первой цифры от третьей цифры числа?

Задача 40. Рассматриваются все шестизначные числа, у которых сумма первых трех цифр равна сумме последних трех.

1. Сколько информации о таком числе несет сумма первых трех цифр?
2. Сколько информации произведение первых трех цифр несет о произведении трех последних цифр?
3. Зависит ли разность первой и четвертой цифр от разности второй и пятой цифр?

Задача 41. Рассматриваются все простые четырехзначные числа.

1. Сколько информации о таком числе несет сумма цифр?

2. Сколько информации сумма двух первых цифр такого числа несет о сумме двух последних цифр?
3. Зависит ли первая цифра такого числа от трех последних?

Задача 42. Рассматриваются все трехзначные числа.

1. Сколько информации о таком числе несет количество его делителей?
2. Сколько информации количество делителей числа несет о его последней цифре?
3. Зависит ли количество делителей числа от суммы делителей?

Задача 43. Рассматриваются четырехзначные числа, у которых разница между наибольшей и наименьшей цифрами не превосходит пяти.

1. Найти энтропию первой и последней цифр таких чисел.
2. Сколько информации первые две цифры такого числа несут о последних двух цифрах?
3. Зависит ли сумма цифр такого числа от разности между наибольшей и наименьшей цифрами?

Задача 44. Рассматриваются четырехзначные числа. Назовем *сверткой числа* сумму попарных произведений его цифр.

1. Сколько информации о числе несет знание о том, что свертка его цифр меньше двадцати?
2. Сколько информации свертка цифр числа несет о наличии среди цифр нуля?
3. Зависит ли первая цифра числа от свертки?

Задача 45. Рассматриваются целые числа диапазона 0–20 и их русские и английские названия.

1. Сколько информации о числе несет первая буква его русского и английского названий?
2. Сколько информации длина русского названия числа несет о длине его английского названия?
3. Зависит ли величина числа от количества букв в его названии?

Задача 46. Рассматриваются целые числа диапазона 0–20 и их русские и английские названия.

1. Сколько информации о числе несет длина его русского и английского названий?

2. Сколько информации буква русского названия числа несет о букве его английского названия, стоящей на той же позиции?
3. Зависит ли длина русского названия числа от длины его английского названия?

Задача 47. Рассматриваются целые числа диапазона 0–20 и их русские и английские названия. Назовем *сверткой слова* сумму номеров (по алфавиту) входящих в него букв.

1. Сколько информации о числе несет несет последняя цифра свертки его русского и английского названий?
2. Сколько информации последняя цифра свертки русского названия числа несет о последней цифре свертки его английского названия?
3. Зависит ли первая буква английского названия числа от первой буквы его русского названия?

Задача 48. Рассматривается колода из 52 карт.

1. Что несет больше информации о пятерке карт: знание об отсутствии среди них какой-то одной масти или знание о присутствии там тузов?
2. Сколько информации наибольшая бубновая карта несет о наименьшей бубновой карте в тройке карт?
3. Зависит ли присутствие пик в тройке карт от присутствия там червей?

Задача 49. Рассматривается всевозможные пятерки карт из 52 картовой колоды. Покер.

1. Сколько информации о пятикартовом стрите[2] дает ответ на вопрос о наличии в нем валета?
2. Сколько информации о фуле[3] дает знание одной его карты?
3. Зависит ли значение старшей пиковой карты в пятерке карт от значения старшей червовой карты?

Задача 50. Рассматривается колода из 36 карт. Очко. Игрок руководствуется правилом: прикупать карту тогда и только тогда, когда количество очков на руках меньше пятнадцати.

[2](от англ. *straight*) пять карт, идущих по порядку, любых мастей. Туз может входить в два стрита: Т 2 3 4 5 и 10 В Д К Т.

[3](от англ. *full house*) две карты одного достоинства и три карты другого достоинства, например, 7 7 8 8 8.

1. Известно, что игрок остановился после двух прикупов. Какова энтропия количества набранных им очков?
2. Сколько информации о количестве набранных очков содержит количество набранных карт?
3. Зависит ли количество набранных очков от масти первой карты?

Задача 51. Рассматривается текст задания.

1. Сколько информации о слове данного задания несет информация о наличии или отсутствии в нем буквы «е»?
2. Сколько информации о следующей букве в данном задании содержит знание о том, гласной или согласной является предшествующая ей буква?
3. Зависит ли длина слова в данном задании от наличия в нем гласных?

Задача 52. Рассматривается текст гимна России.

1. Какая из позиций букв в слове наиболее информативна?
2. Сколько информации первая буква слова несет о последней букве того же слова?
3. Зависит ли последняя буква слова от позиции слова в строке?

Задача 53. Рассматривается текст гимна России.

1. Сколько информации о слове несет его длина?
2. Сколько информации первая гласная буква слова несет о его первой согласной букве?
3. Зависит ли количество гласных в слове от количества согласных в том же слове?

Задача 54. Рассматривается текст гимна России.

1. Какая из букв слова, первая гласная или первая согласная, несет больше информации о слове?
2. Сколько информации первая согласная буква слова несет о первой гласной букве слова?
3. Зависит ли число букв в слове от позиции слова в строке?

Глава 4. Защита от шума

4.1 Ошибки при передаче информации

Теория информации возникла из потребностей связи. К. Шенноном и его последователями в двадцатом веке было получено как теоретическое, так и практическое решение проблемы передачи информации по каналу связи с шумом.

Инженеры долго не могли поверить, что без уменьшения скорости передачи можно добиться сколь угодно высокой надежности переданной информации за счет умного кодирования.

При передаче информации даже у самой совершенной техники случаются ошибки. Представьте, что случится с текстом, сжатым алгоритмом Хаффмана, если там изменить всего один бит: весь файл после этого бита может быть декодирован неправильно. Если же файл не был сжат, то изменение одного бита изменит лишь одну букву текста и эта ошибка, скорее всего, будет легко исправима и практически не повлияет на смысл текста. Это свойство несжатого текста связано с его избыточностью. Естественный язык обладает значительной информационной избыточностью, которая позволяет восстанавливать слова с несколькими ошибками. Если бы не эта избыточность, нам было бы трудно понимать друг друга. В искусственные языки избыточность закладывается при их создании.

4.2 Контроль четности

Для уменьшения числа ошибок информацию часто передают с дополнительными данными — контрольными суммами. Простейший вид контрольной суммы представляет собой так называемый *бит четности*. При каждой передаче байта внутри компьютера передаются на самом деле не восемь, а девять битов. Девятый — невидимый пользователю

бит четности — является нулем, если количество единиц в байте четно, и единицей в противном случае. Таким образом, сумма цифр расширенного байта (байта с битом четности) всегда четна, и сохранение этой четности контролируется при всех передачах. Если обнаруживается *ошибка четности*, то есть нечетность у полученного расширенного байта, то этот байт считывается заново или выдается сообщение об ошибке чтения.

4.3 Контрольная сумма

Важнейшим программным способом защиты информации являются *контрольное суммирование*. Всякий файл можно рассматривать как последовательность чисел; контрольная сумма определяется как порязрядная сумма этих чисел. После чтения файла вычисляется контрольная сумма у считанного файла и сравнивается с записанной контрольной суммой. Этот метод применяется также для защиты от вирусов. Антивирусная программа вычисляет контрольные суммы файлов или секторов жесткого диска и записывает их в отдельный файл. Если вирус что-то меняет в файле, то контрольная сумма файла меняется и антивирусная программа может это изменение обнаружить и локализовать.

4.4 Локализация ошибки

Если мы точно знаем, в каком бите возникла ошибка, то мы можем ее исправить — достаточно лишь изменить значение этого бита на противоположное.

Занумеруем в двоичной системе все биты передаваемой последовательности. Пусть N — число бит передаваемой последовательности, k — целая часть $\log_2 N$, то есть $2^k \leqslant N \leqslant 2^{k+1}$.

Код Хэмминга позволяет обнаружить и исправить ошибку в последовательности битов при условии, что ошибок не более одной (такие коды называются самоконтролирующимися и самокорректирующимися).

Пусть требуется передать n бит информации. В переданном тексте, кроме n бит информации, должно оставаться место для информации о позиции ошибки — зарезервируем для этой информации k битов, называемых *контрольными разрядами*. Количество контрольных разрядов

должно удовлетворять неравенству $k \geqslant \log_2(k+n+1)$. Видно, что количество дополнительных бит логарифмически зависит от длины кодовой последовательности. Ниже приводим таблицу, показывающую максимальное значение n для $k < 10$:

k	2	3	4	5	6	7	8	9
n	1	4	11	26	57	120	247	502

4.5 Построение кода Хэмминга

Пусть $2^k \geqslant k + n + 1$. Передаваемые биты перенумеруем от 1 до $n + k$, зарезервировав места, номера которых являются степенями двойки, то есть $1, 2, 4, 8, \ldots$. Все остальные места сопоставим местам в исходной последовательности длины n бит.

В первый бит мы поставим 0 или 1 таким образом, чтобы сумма всех нечетных номеров битов передаваемого слова была равна нулю. Заметим, что первый бит — единственный из зарезервированных битов с нечетным номером, поэтому, определяя впоследствии значения остальных резервных битов, мы не нарушим свойства первого бита.

Второй бит мы определяем так, чтобы сумма битов, номер которых имеет в двоичном представлении вторую цифру 1, была четна. Заметим, что номера всех зарезервированных битов после второго имеют вторую цифру ноль, поэтому их значения не влияют на свойства второго бита.

k-ый зарезервированный бит определяется так, чтобы четной была сумма значений тех битов, двоичный номер которых имеет единицу на k-ой позиции своего двоичного представления.

Последний зарезервированный бит выбирается так, чтобы четной была сумма всех битов передаваемой последовательности.

4.6 Декодирование

Получив закодированную с помощью кода Хэмминга битовую последовательность, можно найти и исправить ошибку, если она только одна. Сначала вычисляется сумма s_0 всех битов полученной последовательности. Если s_0 четна, то ошибки нет. Если нечетна, то вычисляем суммы s_i всех значений битов, в номере которых на i-ой двоичной позиции

стоит единица. Если s_i четна, то среди битов с единицей на i-ой позиции нет ошибок, следовательно, бит с ошибкой имеет номер, в котором на i-ой позиции стоит 0. Если же s_i нечетна, то двоичный номер бита с ошибкой имеет на i-ой позиции единицу. Таким образом, значение двоичного номера бита с ошибкой равно $\sum 2^i(1 - s_i)$.

4.7 Определение фальшивой монеты

Задача о фальшивой монете заключается в том, чтобы за минимальное число взвешиваний определить фальшивую монету среди данного количества монет. Если заранее известно, что фальшивая монета среди данных имеется, то мы будем называть задачу *точной*. Если же фальшивой монеты может и не быть, то задачу будем называть *неточной*. Если известно, что фальшивая монета легче (тяжелее) настоящих, то будем называть задачу *определенной*. Если же заранее неизвестно, легче фальшивая монета или тяжелее, то задачу будем называть *неопределенной*.

Взвешивание определяется разбиением множества монет на три подмножества: монеты, положенные на левую чашку, монеты, положенные на правую чашку и монеты, оставшиеся на столе. Если в левой чашке k монет, в правой l монет и на столе осталось m монет, то мы будем будем писать, что взвешивание имеет тип $k + l + m$. Каждое взвешивание дает один из трех возможных результатов: левая чашка легче, правая чашка легче или весы находятся в равновесии.

4.8 Дерево алгоритма

В узлах дерева алгоритма изображаются взвешивания или заключения. Начальный узел соответствует первому взвешиванию. От каждого узла взвешивания вниз идут три ребра, соответствующие трем возможным результатам взвешивания. Если на основе результата взвешивания удалось определить фальшивую монету, то ребро, исходящее из последнего взвешивания завершается *заключительным узлом* — узлом, в котором пишется заключение — номер фальшивой монеты.

Лемма 4.1. *Если z_l обозначает количество заключений, сделанных после l взвешиваний, то при любом k справедливо равенство*

$$z_k + 3z_{k-1} + 3^2 z_{k-2} + \ldots + 3^{k-1} z_1 = 3^k.$$

Доказательство. Модифицируем алгоритм так, чтобы он не делал заключений ранее чем после k взвешиваний. Для этого в том случае, если заключение можно было сделать на основании какого-то из результов i-го взвешивания при $i < k$, мы назначаем в качестве следующего взвешивания повторение предыдущего и делаем заключение лишь при $i = k$. В таком случае количество заключений модифицированного алгоритма будет в точности равно 3^k. Каждое заключение исходного алгоритма, сделанное после $k - i$ взвешиваний, порождает 3^i заключений модифицированного алгоритма после k взвешиваний. Поэтому общее количество заключений модифицированного алгоритма после k взвешиваний выражается суммой $z_k + 3z_{k-1} + \ldots + 3^{k-1}z_1$, что и доказывает нашу лемму. □

Теорема 4.1. *Не существует алгоритма, который решает определенную точную задачу о нахождении фальшивой среди n монет менее чем за $\log_3 n$ взвешиваний.*

Доказательство. Суммарное число заключений $z = z_1 + z_2 + \ldots + z_k$ алгоритма, распознающего фальшивую монету за k взвешиваний, не может быть меньше, чем число монет, поскольку любая монета может оказаться фальшивой. Поэтому $z \geqslant n$. В силу леммы 4.1 получаем $z \leqslant 3^k$. Откуда $k \geqslant \log_3 z \geqslant \log_3 n$. □

Теорема 4.2. *Не существует алгоритма, который решает определенную неточную задачу о нахождении фальшивой среди n монет менее чем за $\log_3(n+1)$ взвешиваний.*

Доказательство. В этом случае должно быть также заключение об отсутствии фальшивых монет. Поэтому $z \geqslant n + 1$. □

Теорема 4.3. *Не существует алгоритма, который решает неопределенную точную задачу об определении фальшивой из n монет менее чем за $\log_3(2n-1)$ взвешиваний.*

Доказательство. Обозначим через $l(i)$ заключительный узел дерева алгоритма, в котором алгоритм завершает работу в том случае, если i-ая монета фальшивая и она легче настоящих, и обозначим через $t(i)$ номер заключительного узла, где алгоритм завершает работу в случае, если i-ая монета фальшивая и тяжелее настоящих.

Если $l(i) = t(i)$, то i-ая монета не взвешивалась. Действительно, если бы i-ая монета попала на весы, то результаты взвешиваний были бы различны в зависимости от того, легкая она или тяжелая.

Для того, чтобы определить фальшивую монету, должны быть взвешены все монеты, за исключением, возможно, одной. Следовательно, $l(i) \neq t(i)$ для всех i, кроме, может быть, одного. Поэтому число заключительных узлов алгоритма не может быть меньше, чем $2n-1$. □

Теорема 4.4. *Не существует алгоритма, который бы решал неопределенную и неточную задачу обнаружения фальшивой среди n монет менее чем за $\log_3(2n+1)$ взвешиваний.*

Доказательство. В этом случае все монеты должны взвешиваться, иначе мы не можем гарантировать отсутствие фальшивой монеты. Следовательно, $t(i) \neq l(i)$ при любом i. Таким образом, число заключительных узлов алгоритма не менее $2n$. Кроме того, должен быть узел, соответствующий случаю, когда фальшивых монет нет. Следовательно, общее количество заключительных узлов не менее чем $2n+1$. □

Теорема 4.5. *Существует алгоритм, решающий определенную точную задачу о нахождении фальшивой монеты за $k \geqslant \log_3 n$ взвешиваний.*

Доказательство. Пусть фальшивая монета легче настоящих. Если $n = 3$, то задачу можно решить за одно взвешивание типа $1+1+1$. В случае неравновесия, фальшивая находится в более легкой чашке, в случае равновесия — на столе. Для $n = 2$ задачу решает взвешивание $1+1+0$. Для $n = 1$ мы без взвешивания знаем, что монета фальшивая.

Теперь алгоритм, распознающий легкую фальшивую монету за обещанное количество взвешиваний, определяется так. Если $n = 3k$, то первое взвешивание будет $k+k+k$. Если $n = 3k+1$, то первое взвешивание — $k+k+(k+1)$. И наконец, если $n = 3k+2$, то первое взвешивание назначается типа $(k+1)+(k+1)+k$. В случае неравновесия фальшивая монета ищется далее среди находящихся в легкой чашке, в случае равновесия — среди оставшихся на столе. □

Теорема 4.6. *Не существует алгоритма, решающего неточную неопределенную задачу обнаружения фальшивой из 13 монет за 3 взвешивания.*

Доказательство. Предположим, что такой алгоритм есть. Пусть первое взвешивание этого алгоритма имеет тип $k + k + m$. Тогда в случае равновесия возникает неточная неопределенная задача для m монет, которая должна решаться за 2 взвешивания. В силу теоремы 4.4 отсюда вытекает неравенство $2m + 1 \leqslant 9$. То есть $m \leqslant 4$. Поскольку m нечетно, получаем, что $m \leqslant 3$.

Модифицируем алгоритм в случае равновесного положения при первом взвешивании. В модифицированном алгоритме во второе взвешивание в левую чашку мы кладем все монеты, оставшиеся на столе в первом взвешивании, а в правую — такое же количество монет из правой чашки первого взвешивания. Поскольку нам известно, что фальшивой монеты среди последних нет, то в качестве результата этого взвешивания мы получаем в случае неравновесия определенную и точную задачу для не более чем трех монет, разрешимую за одно взвешивание. В случае равновесия мы сразу, без третьего взвешивания, приходим к заключению, что все монеты настоящие. Таким образом модифицированный алгоритм также решает задачу для 13 монет.

Модифицированный алгоритм имеет заключительный узел после двух взвешиваний. Следовательно, $z_2 \geqslant 1$ и согласно теореме 4.1 получаем, что $9z_1+3z_2+z_3 = 27$. С другой стороны, количество заключений $z_1 + z_2 + z_3$ должно быть больше либо равно $2 \cdot 13 + 1 = 27$. В таком случае $8z_1 + 2z_2 \leqslant 0$, что невозможно при положительном z_2. □

Задачи

1. Сколько битов нужно дополнительно передать, чтобы обнаружить наличие не более чем двух ошибок?
2. Сколько битов нужно дополнительно передать, чтобы можно было обнаружить и исправить ровно две ошибки в последовательности из n бит?
3. Придумайте код, исправляющий две ошибки при передаче 9 бит информации.
4. Насколько арифметическая контрольная сумма более информативна по сравнению с логической (порязрядной)?

Задачи к главе 4

Задача 1.

1. Докажите, что не существует алгоритма, упорядочивающего любой массив из n элементов за менее чем $\log_2 n!$ сравнений.
2. Определить минимальное количество взвешиваний, необходимых для определения одной фальшивой (более легкой) монеты среди n, если на чашку весов помещается не более 9 монет.

Задача 2.

1. Какое минимальное число сравнений должен сделать алгоритм, находящий медиану среди $2n + 1$ чисел?
2. За какое минимальное число взвешиваний можно определить наличие фальшивых среди n монет, если известно, что фальшивые монеты весят одинаково, они легче настоящих и фальшивых монет не больше двух?

Задача 3.

1. Какое минимальное необходимое число сравнений должен сделать алгоритм, делящий $2n$ чисел пополам таким образом, чтобы все числа одной половины не превосходили всех чисел второй? Минимальность доказать.
2. За какое минимальное число взвешиваний можно определить наличие двух фальшивых среди n монет, если известно, что фальшивые монеты имеют одинаковый вес, но неизвестно, легче они или тяжелее, чем настоящие?

Задача 4.

1. Какое минимальное необходимое число сравнений должен сделать алгоритм, делящий $3n$ чисел на две части в отношении 1:2

таким образом, чтобы все числа меньшей половины не превосходили всех чисел большей? Минимальность доказать.

2. За какое минимальное число взвешиваний можно определить наличие двух фальшивых среди n монет, если известно что они легче настоящих, но вес у них различный?

Задача 5.

1. Каково минимальное число вопросов, позволяющее определить загаданное число из известных n чисел, если разрешенный вопрос имеет вид «это x?», а ответ имеет три варианта: «да», «меньше, чем x» или «больше, чем x», при этом ответ дается на неизвестном языке?
2. Каково минимальное число взвешиваний, позволяющих найти две разноцветные фальшивые монеты среди $2n$ монет, из которых половина белых, половина черных? Все настоящие монеты весят одинаково. Вес фальшивых одинаков, но неизвестно, легче фальшивая монета или тяжелее, чем настоящая.

Задача 6.

1. Угадываем натуральное число от 1 до n. Разрешенный вопрос имеет вид «это x?», возможны три варианта ответа: «это ближе к загаданному, чем число из предыдущего вопроса», «это дальше от загаданного, чем число из предыдущего вопроса» или «это на том же расстоянии от загаданного, что и число из предыдущего вопроса». Каково минимальное число вопросов, позволяющее определить загаданное натуральное число?
2. Каково минимальное число взвешиваний, позволяющих найти одну из двух разноцветных фальшивых монет среди $2n$ монет, из которых половина белых, половина черных? Все настоящие монеты весят одинаково. Вес фальшивых одинаков и отличается в неизвестную сторону от веса настоящих.

Задача 7.

1. Докажите, что не существует алгоритма, находящего наибольшее из n чисел за менее чем $n-1$ сравнение.
2. Оценить минимальное число взвешиваний, необходимых для обнаружения двух фальшивых монет среди n, если известно, что одна из них легче, а другая тяжелее настоящей, а в сумме они весят как две настоящие.

Задача 8.

1. Какое минимальное число сравнений должен сделать алгоритм, находящий число, которое превосходят ровно n среди данных $3n+1$ чисел?
2. Каково минимальное число взвешиваний, позволяющих найти две разноцветные фальшивые монеты среди $3n$ монет, из которых треть белых, две трети черных? Все настоящие монеты весят одинаково. Вес фальшивых одинаков и они легче настоящих.

Задача 9.

1. Какое минимальное число сравнений должен сделать алгоритм, находящий число, являющееся третьим по величине среди n чисел?
2. Определить минимальное количество взвешиваний, необходимых для выявления наличия фальшивых монет неизвестного веса среди $2n$, если на чашку весов помещается не более 9 монет.

Глава 5. Информативность классификатора

5.1 Задача распознавания логов интернет-сессии

Лог интернет-сессии содержит информацию о запросах пользователя (содержание и время запроса), о посещенных им сайтах и многое другое. Так как любой вид информации можно закодировать словами, мы будем упрощенно представлять себе лог интернет-сессии как конечную последовательность слов $x_1, x_2, \ldots, x_n$ переменной длины n.

В частности, имеющиеся в нашем распоряжении логи классифицированы, то есть содержат информацию о поле, возрасте и других характеристиках пользователя, которые нам нужно научиться распознавать. Две основные задачи — распознавание пола и возрастной группы. Отличием этих двух задач является то, что полов два, а возрастных групп можно выделить несколько, причем различными способами. Кроме того, деление по полу дает примерно равные классы, что тоже немного упрощает задачу. Поэтому в дальнейшем мы сконцентрируемся именно на задаче распознавания пола.

Данная нам база логов (случайным образом) разбита пополам на обучающую и контрольную выборки. Наша задача — придумать какую-то функцию на логах, использующую статистику обучающей выборки, по значению которой мы могли бы разделять логи контрольной выборки на мужские и женские. При этом в определенном проценте случаев нам разрешается отказываться от классификации.

Статистика, которой мы собираемся пользоваться — это *статистика пола* слов. А именно, для каждого слова x_i его статистикой пола называется пара чисел (m_i, f_i), выражающие количество мужских m_i и женских f_i логов обучающей выборки, в которых встречается слово x_i. В принципе, для данной задачи могут быть использованы также статистики встречаемости пар или троек слов, но мы ограничимся простейшими статистиками.

Таким образом, мы ищем классификатор в виде функции от лога

$$F(x_1, \ldots, x_2) = F((m_1, f_1), (m_2, f_2), \ldots, (m_n, f_n)), \tag{5.1}$$

в которой входящие в лог слова заменены на их статистики пола. Например, в качестве классификатора можно рассмотреть *отношение сумм статистик*:

$$\frac{\sum\limits_{i=1}^{n} m_i}{\sum\limits_{i=1}^{n} f_i}. \tag{5.2}$$

Если это отношение больше единицы, то естественно классифицировать лог как мужской, а если меньше единицы — как женский. Однако рассчитывать на высокую точность распознавания при таком подходе не следует. Повысить точность распознавания можно за счет отказа от распознавания сомнительных случаев. Например, можно отказаться от распознавания, если отношение (5.2) отличается от единицы меньше чем на 10%. В этом случае точность распознавания несколько возрастет, но уменьшится *покрытие*, т. е. доля случаев, когда производится распознавание.

Ясно, что с ростом отношения сумм статистик растет вероятность, что рассматриваемый лог принадлежит мужчине, а убывание этого отношения повышает вероятность принадлежности лога женщине. Поэтому можно выбрать два порога, мужской M и женский F, и классифицировать лог как мужской, если отношение сумм статистик превышает M, классифицировать лог как женский, если отношение меньше чем F, и отказываться от классификации, если отношение сумм статистик находится в диапазоне между M и F.

Другой пример классификатора дает *среднее логарифмов отношений статистик*:

$$\frac{1}{n} \sum_{i=1}^{n} \log_2 \frac{m_i + 1}{f_i + 1}. \tag{5.3}$$

5.2 Кривая «точность–покрытие»

Четкий классификатор выдает для каждого лога или номер класса, которому он принадлежит, или отказ от классификации.

Покрытием для четкого классификатора называется доля (в процентах) классифицированных логов. *Точностью* классификатора называется доля (процент) правильно классифицированных логов.

Нечеткий однозначный классификатор представляет собой функцию, *ковариантную* вероятности принадлежности к данному классу, то есть такую, возрастание которой влечет возрастание вероятности принадлежности к данному классу.

Нечеткий n-значный классификатор можно трактовать как совокупность n однозначных нечетких классификаторов. В случае двух классов нечеткий классификатор часто реализуется в виде одной функции, возрастание которой увеличивает вероятность принадлежности одному классу, одновременно снижая вероятность принадлежности к другому. Именно таковы приведенные выше два примера классификаторов пола. (На практике, однако, при решении этой задачи более эффективно оказалось строить различные одноклассные классификаторы для мужчин и женщин.)

Четкий однозначный классификатор порождается из нечеткого с помощью *порогового значения*: объекты, для которых значение нечеткого классификатора превосходит выбранный порог, классифицируются как принадлежащие данному классу, а объекты с подпороговым значением нечеткого классификатора не классифицируются. Если нечеткий двузначный классификатор представлен одной функцией, то четкий двузначный классификатор на его основе строится с помощью выбора двух пороговых значений, так что объекты со значением классификатора большим обоих порогов классифицируются как принадлежащие одному классу, а меньшие обоих порогов — как принадлежащие другому. Объекты со значением классификатора между порогами не классифицируются.

Основной характеристикой нечеткого классификатора является *кривая «точность–покрытие»*, которая строится следующим образом: для каждого значения порога мы получаем точность и покрытие порожденного четкого классификатора. Точность откладывается по оси ординат, покрытие — по оси абсцисс.

5.3 Информативность классификатора

Информативность классификатора определяется как количество информации, которое значение классификатора несет о классе объекта.

Информативность четкого n-классного классификатора вычисляется по формуле:

$$\sum_{i=0}^{n} p_i \sum_{j=1}^{n} p_{ij} \log_2 \frac{p_i}{p_{ij}}, \tag{5.4}$$

где p_i при $i > 0$ — доля объектов, отнесенных к i-му классу, p_0 — доля отказов от классификации, p_{ij} — вероятность того, что объект, отнесенный к i-му классу, окажется объектом j-го класса.

Матрицу $\{p_{ij}\}$ мы будем называть *матрицей распознавания* классификатора. Все основные характеристики классификатора выражаются через его матрицу распознавания. А именно, точность дается формулой

$$\frac{\sum\limits_{i=1}^{n} p_{ii}}{1 - p_0}, \tag{5.5}$$

покрытие — формулой

$$1 - \sum_{i=1}^{n} p_{0i}, \tag{5.6}$$

информативность — приведенной выше формулой (5.4) с учетом равенств $p_i = \sum\limits_{j=1}^{n} p_{ij}$ $(i \geqslant 0)$.

Информативность нечеткого классификатора может быть выражена аналитически на основе кривой «точность–покрытие» по формуле:

$$-\int_0^S \left(\frac{dp(s)}{ds} \log_2 \frac{dp(s)}{ds} + \left(1 - \frac{dp(s)}{ds}\right) \log_2 \left(1 - \frac{dp(s)}{ds}\right) \right) ds. \tag{5.7}$$

Здесь s представляет собой покрытие (выраженное долей, то есть числом между нулем и единицей), а $p(s)$ — значение точности классификатора на покрытии s.

Для практического вычисления информативности нечеткого классификатора нам придется сначала произвести квантование множества значений классификатора, причем таким образом, чтобы иметь достаточную статистику по каждому кванту. (С теоретической точки зрения

наилучшим квантованием является *квантильное*, то есть получающееся делением множества значений такими порогами, чтобы доля объектов, относящихся к значениям между любой парой соседних порогов, равнялась фиксированному квантилю — например, одной десятой распределения.) Для каждого кванта считается энтропия представленных в нем классов. После этого посчитанные энтропии квантов усредняются с весами, пропорциональными долям соответствующих квантов. Другими словами, на основе нечеткого классификатора строится аппроксимирующий его четкий классификатор с числом классов, равным количеству квантованных значений, и информативность аппроксимирующего классификатора принимается за информативность исходного.

5.4 Неравенство Фано

Если n-значный классификатор с матрицей распознавания $\{p_{ij}\}$ имеет точность P на покрытии S, то неопределенность, которую мы имеем после классификации, может быть оценена сверху следующим образом. В случае отказа от классификации неопределенность не превосходит $\log_2 n$. В случае назначения объекту некоторого класса наибольшая неопределенность возникает, если все классы численно равны и одинаково классифицированы, т. е. все p_{ij} при $i \neq j$ равны между собой, так же как и все p_{ii} между собой. В таком случае P является общим значением p_{ii}, а общим значением p_{ij} является $\frac{1-P}{n-1}$. Поэтому апостериорная (возникшая после классификации) неопределенность оценивается сверху величиной

$$(1-S)\log_2 n - SP\log_2 P + S(1-P)\log_2 \frac{n-1}{1-P}. \tag{5.8}$$

В то же время неопределенность, которую мы имели до классификации, представляется величиной

$$H_0 = \sum_{j=1}^{n} p^j \log_2 p^j, \tag{5.9}$$

где через p^j обозначена доля объектов j-го класса. Если доли всех классов совпадают (равномерное распределение классов), то последнее выражение совпадает с $\log_2 n$. Тогда среднее количество информации, которую несет классификатор о классе распознаваемого объекта, равно

разности неопределенностей относительно класса объекта, имевшихся до и после классификации. На основе данных выше оценок мы можем в случае равномерного распределения классов написать такое неравенство:

$$I \geqslant SP \log_2 \frac{P(n-1)}{1-P} + S \log_2 \frac{n(1-P)}{n-1}. \qquad (5.10)$$

В частности, именно этот случай мы имеем в задаче распознавания пола лога. Если результатом наших исследований должно быть распознавание пола по логу интернет-сессии с вероятностью 80% (при стопроцентном покрытии), то количество информации о поле, которым мы должны владеть согласно (5.10), оценивается снизу величиной 0.278 бита/лог. (В данном случае эту оценку легко получить, ведь априорная неопределенность пола считается равной одному биту, а апостериорная энтропия измеряется величиной $\frac{1}{5}\log_2 5 + \frac{4}{5}\log_2 \frac{5}{4}$.) Если же мы будем отталкиваться от известного нам соотношения мужского и женского полов (0.55 к 0.45), то исходная неопределенность будет немного меньше.

Правая часть оценки информативности линейно зависит от покрытия. Зависимость от точности при фиксированном покрытии такова, что при точности выше 80% для повышения точности распознавания на один процент количество добавочной информации увеличивается на несколько процентов, тогда как при точности, ненамного отличающейся от 50%, каждый добавочный процент информации дает возможность повышения точности на несколько процентов.

5.5 Закон геометрической прогрессии (ЗГП)

Итак, где нам взять 0.278 бита/лог информации о поле? Для начала нужно определить, сколько информации о поле содержат рассматриваемые признаки, прежде всего — сколько информации о поле в среднем несет статистика пола случайно выбранного слова лога.

Обозначим через H_k энтропию пола для случайно выбранных k слов из случайно выбранного лога (сначала случайно выбирается лог, а потом в нем по очереди выбираются слова). Экспериментально установлен следующий *геометрический закон*: количество новой информации,

которое несет каждое следующее выбранное слово лога, убывает в геометрической прогрессии:

$$\frac{H_{k+1} - H_k}{H_k - H_{k-1}} = q. \tag{5.11}$$

Для рассматриваемых данных $q \approx 0.9$.

Можно оценить количество информации о поле, которое несет статистика пола всех слов любого лога, суммой бесконечной геометрической прогрессии. Это дает возможность получить реалистичную оценку всей доступной нам информации (в виде статистики пола слов) и на основе этого дать оценку сверху для информативности любого классификатора, использующего только статистику пола слов:

$$I \leqslant H_0 - H_\infty = (H_0 - H_1) + (H_1 - H_2) + \ldots = \frac{H_0 - H_1}{1 - q}. \tag{5.12}$$

Здесь и далее H_0 обозначает априорную энтропию классификатора, то есть энтропию пола в рассматриваемой выборке. С учетом равенства (5.11) получаем верхнюю оценку информативности классификатора в виде

$$I \leqslant \frac{(H_0 - H_1)^2}{H_0 - 2H_1 + H_2}, \tag{5.13}$$

для вычисления которой нужно научится правильно определять входящие в нее однословную и двусловную энтропии, чем мы займемся в следующей лекции.

Разработанные методы позволяют быстро и достаточно точно определить количество имеющейся у нас информации и с помощью неравенства Фано оценить принципиальную возможность построения классификатора с заданными характеристиками. Например, ограничение информативности классификатора позволяет оценить максимальное покрытие при заданной точности.

5.6 Проверка ЗГП

Во-первых, ЗГП хорошо предсказывает H_3 (энтропию пола для трех случайно выбранных слов), вычисление которой проводилось многократно на разных данных. Во-вторых, предсказанная ЗГП максимальная информативность хорошо согласуется с наблюдаемой информа-

тивностью у работающих классификаторов. Третьим подтверждающим экспериментом является поведение оценки количества информации при увеличении периода наблюдения. Интуитивно ясно, что с ростом периода наблюдения (то есть длины логов) количество имеющейся в них информации (в нашем случае — о поле пользователя) должно возрастать, тогда как количество информации, которое несет одно слово, может и убывать (ведь число слов в логе растет). Именно такого рода эффект был обнаружен С. Анисовым. При увеличении времени наблюдений с двух недель до месяца уменьшилась величина H_1, но при этом величина H_∞ увеличилась.

Задачи

1. Выразить интегралом информативность двуклассного класификатора по характеристике «точность–покрытие».

Задачи к лекции 5

Задача 1. Точность классификатора равна 80%, покрытие — 50%. На входе классификатора — три класса с частотами 40%, 40%, 20%. Какова минимальная информативность (бит/лог) классификатора?

(Ответ: 0.27 бит/лог)

Задача 2. Точность классификатора равна 80%, покрытие — 50%. Количество имеющейся информации о возрастной категории лога оценивается сверху величиной 1 бит/лог. Количество возрастных категорий равно семи. Оценить сверху долю элементов в наименьшей по численности возрастной категории.

Задача 3. Количество имеющейся в наличии информации оценивается сверху величиной 1/2 бит/лог (распознаются логи интернет-сессий). Распознаваемые логи делятся на три класса в пропорции $3 : 2 : 1$. Оценить максимально возможное покрытие для классификатора с точностью распознавания 80%.

(Ответ: 93%)

Задача 4. На вход классификатора, определяющего возрастную группу по логу интернет-сессии, подаются логи семи возрастных групп, количественные соотношения между которыми выражаются в процентах: 5%, 10%, 20%, 30%, 20%, 10%, 5%. Классификатор в половине случаев отказывается от классификации, а в остальных случаях в качестве ответа выдает номер одной из трех самых многочисленных групп: 25% (третья группа) 50% (четвертая группа), 25% (пятая группа). Какова может быть максимальная точность классификатора, если известно, что количество используемой информации, различающей любую группу от ее дополнения, не превышает 0.2 бита/лог?

(Ответ: 0.38)

Задача 5. Каково может быть максимальное покрытие классификатора при точности 80%, если распознаются три равных по величине класса, для каждого из которых количество имеющейся информации, отличающей класс от его дополнения, не превосходит 0.3 бита/элемент?

(Ответ: 70.5%)

Глава 6. Проблема недостаточной статистики

6.1 Вычисления H_1

Продолжим рассмотрение задачи распознавания класса лога в n-классной классификации (в задаче распознавания пола лога $n = 2$).

Будем рассматривать лог как последовательность слов $w_1, \ldots, w_n$. *Первичной статистикой* слова w называется набор чисел $(c_1, \ldots, c_n)$, где c_i является числом логов i-го класса, в которых встретилось слово w. В случае задачи распознавания пола первичная статистика слова состоит из пары чисел f и m, выражающих соответственно количество женских и мужских логов, в которых встретилось данное слово.

Через H_1 обозначается среднее количество информации о классе лога контрольной выборки, которое несет первичная статистика одного слова лога. Для того, чтобы посчитать H_1, нам нужно оценить вероятности классов по известной первичной статистике. Здесь мы сталкиваемся с проблемой редких слов. Если слово со статистикой $(c_1, \ldots, c_n)$ достаточно частое, то в качестве оценки вероятности принадлежности лога к i-му классу можно было бы взять отношение

$$p_i = \frac{c_i}{\sum\limits_{j=1}^{n} c_j}, \tag{6.1}$$

тогда энтропия класса лога для данного слова w выражалась бы по обычной формуле

$$H_1(w) = -\sum_{i=1}^{n} p_i \log_2 p_i. \tag{6.2}$$

Для лога $W = (w_1, \ldots, w_m)$ через $H_1(W) = \frac{1}{m} \sum\limits_{i=1}^{m} H(w_i)$ обозначим среднюю энтропию его слов, тогда искомая средняя энтропия слова по

всей выборке будет выражена формулой

$$H_1 = \frac{1}{L}\sum_{i=1}^{L} H(W_i), \tag{6.3}$$

где $W_1, \ldots, W_L$ — совокупность всех логов обучающей выборки.

Если же слово редкое, то формула (6.2) дает заниженную оценку энтропии. Например, слово, которое встретилось только один раз, дает нулевую энтропию. Если редких слов относительно немного (так бывает только в случае двух классов), то энтропия H_1, вычисленная на основе формулы (6.1), близка к истинной.

Но если мы попытаемся вычислять по аналогичной формуле среднюю энтропию пар слов H_2, то увидим, что здесь относительное количество редких пар велико. Редкие тройки слов составляют уже большинство, и вычисления H_3, основанные на формуле (6.1), совсем далеки от реальности.

6.2 Байесовская регуляризация

Один из широко применяемых методов борьбы с недостаточной статистикой является *байесовская регуляризация*, заменяющая формулу (6.1) следующей:

$$p_i = \frac{c_i + r}{\sum\limits_{j=1}^{n} c_j + rn}, \tag{6.4}$$

где параметр r теоретически равен единице, на практике же подбирается экспериментально.

Байесовская оценка вероятности представляет собой усреднение по всем гипотезам — вероятность гипотезы считается пропорциональной вероятности наблюдаемой статистики при условии этой гипотезы.

В применении к оценке энтропии байесовский подход дает неоднозначный ответ. А именно, можно вычислять энтропию H_1 по приведенным выше формулам, в которых p_i вычисляется по формуле Байеса (6.4) или же непосредственно усреднять энтропию по гипотезам. Так как энтропия нелинейно зависит от вероятности, второй подход дает другой результат:

$$H_1(w) = -\sum_{i=1}^{n} p_i L(p_i), \tag{6.5}$$

где p_i вычисляются по формулам (6.1), а функция $L(x)$ является логарифмической производной гамма-функции Эйлера.

6.3 Вторичная статистика

Отсутствие хорошей теории в данном случае можно компенсировать экспериментом. Для экспериментальной оценки искомых вероятностей предлагается разделить обучающую выборку пополам; по первой половине обучающей выборки составляется первичная статистика слов, по второй половине составляется статистика *типов слов*. Слова, имеющие одинаковую первичную статистику, относятся к одному типу. Для задачи распознавания пола тип слова определяется парой чисел (m, f), дающих количество мужских m и женских f логов, в которых это слово встретилось. Статистика типа (m, f), подсчитываемая по второй половине обучающей выборки, представляет собой пару чисел (M, F), где M и F — соответственно, число мужских и женских логов второй половины выборки, в которых содержатся слова с типом (m, f). Если слово редкое, то тип его — частый, поэтому для него можно определить вероятности на основе имеющейся статистики по формуле наибольшего правдоподобия.

В таблице приведены экспериментальные результаты: первый столбец — тип по первичной статистике, второй столбец содержит наблюдаемую вероятность $(m/(m+f))$, а третий, четвертый, пятый и шестой столбцы содержат байесовскую оценку этой вероятности с параметрами регуляризации, равными 1, 3, 5 и 10, соответственно.

Делались также попытки выполнять регуляризацию с переменным параметром r, зависящим от суммы $m + f$. Наилучшее приближение к экспериментальной вероятности было получено Д. Хромовым:

$$\frac{m + r(m + f)}{m + f + 2r(m + f)}, \tag{6.6}$$

где регуляризующая добавка имеет вид $r(m + f) = \frac{30}{m+f+1}$.

Байесовская регуляризация типов

m, f	$E(1/1)$	$+1/+2$	$+3/+6$	$+5/+10$	$+10/+20$	(Хромов)
0,0	0.521	0.500	0.500	0.500	0.500	0.500
0,1	0.478	0.333	0.429	0.455	0.476	0.483
0,2	0.447	0.250	0.375	0.417	0.455	0.454
0,3	0.427	0.200	0.333	0.385	0.435	0.416
0,4	0.403	0.167	0.300	0.357	0.417	0.375
1,0	0.554	0.667	0.571	0.545	0.524	0.516
1,1	0.511	0.500	0.500	0.500	0.500	0.500
1,2	0.479	0.400	0.444	0.462	0.478	0.472
1,3	0.453	0.333	0.400	0.429	0.458	0.438
2,0	0.586	0.750	0.625	0.583	0.545	0.545
2,1	0.539	0.600	0.556	0.538	0.522	0.528
2,2	0.507	0.500	0.500	0.500	0.500	0.500
3,0	0.609	0.800	0.667	0.615	0.565	0.583
3,1	0.572	0.667	0.600	0.571	0.542	0.562
4,0	0.634	0.833	0.700	0.643	0.583	0.625
ср. кв. откл.		0.145	0.053	0.022	0.025	0.020
средн.откл.		0.124	0.045	0.017	0.019	0.016

6.4 Типичные вероятности

Для подсчета энтропии признака относительно слова для редких слов можно использовать экспериментально полученные вероятности типов, а для остальных — полученные из первичной статистики по какой-то аналитической формуле. Какие слова следует считать редкими? Для простоты мы будем говорить в этом параграфе о задаче распознавания пола. В таком случае тип слова w представляет собой пару чисел (m, f), представляющих собой количество мужских m и женских f логов в первой половине обучающей выборки, содержащих слово w.

Первичной частотой типа (m, f) называем сумму $m + f$, выражающую частоту, с которой встречается какое-то одно слово этого типа в первой половине обучающей выборки. *Вторичной частотой* типа называется количество логов во второй половине обучающей выборки, содержащих слова этого типа. Если первичная частота типа маленькая, то вторичная — большая, и наоборот. Например, тип $(1000, 1500)$ почти наверное будет уникальным (имеющим вторичную частоту один)

даже в очень больших выборках. С теоретической точки зрения логично считать тип редким, если его первичная частота не превосходит вторичной. С практической точки зрения вполне можно ограничится подсчетом экспериментальных вероятностей типов, скажем, для типов с первичной частотой, не превосходящей 10-20, а для остальных использовать аналитическую формулу.

Другой способ подсчета энтропии основан на экспериментальном вычислении для каждого слова его *типичной вероятности*. Предположим, слово w в обучающей выборке встречается в m мужских и f женских логах. Тогда при случайном делении этой выборки пополам слово w получит тип (m_1, f_1) по первой половине с вероятностью

$$C_m^{m_1} C_f^{f_1} 2^{-m-f}, \tag{6.7}$$

поэтому, если обозначить через $p(m, f)$ (экспериментально полученную) вероятность лога быть мужским при условии, что случайно выбранное в нем слово имеет первичную статистику (m, f), то типичная вероятность слова определится по формуле

$$2^{-m-f} \sum_{m_1, f_1 \leqslant m, f} p(m_1, f_1) C_m^{m_1} C_f^{f_1}. \tag{6.8}$$

Вычисление типичных вероятностей слов можно проводить одновременно с вычислением вероятностей типов путем многократного деления обучающей выборки и усреднения получившихся результатов.

Типичные вероятности слов можно также с успехом использовать в классификаторах.

6.5 Третичная статистика

Типичные вероятности позволяют удовлетворительно решить задачу вычисления среднего количества информации, которую несут о двузначном признаке (скажем, поле) одно (H_1) или два (H_2) слова лога. Однако H_3 для двузначного признака или H_2 для трехзначного признака вычислить таким способом уже не удается. Типов становится слишком много и статистики опять начинает не хватать.

Спасти ситуацию помогает отождествление типов с одинаковыми (или близкими) вторичными статистиками. Таковы, например, типы

$1,1$ и $2,2$ в приведенной выше таблице. Законность такого рода отождествления не приведет к неправильной оценке информативности классификатора ввиду следующего утверждения.

Лемма 6.1. *Пусть классификаторы c и c', определенные на типах первичной статистики, таковы, что $c'(x) = c'(y)$ в том и только том случае, когда совпадают вторичные статистики типов x и y. Тогда информативности этих классификаторов равны.*

Доказательство. Действительно, энтропия классификатора вычисляется по вторичной статистике как среднее значение энтропий, соответствующих различным значениям классификатора. Если же значения имеют одинаковую вторичную статистику, то объединение значений даст точно такую же статистику, поэтому слагаемое в энтропии классификатора c', соответствующее объединенному значению, будет в точности равно сумме слагаемых, соответствующих объединяемым значениям в классификаторе c. □

Теперь обучающую выборку мы делим на три равные части. По первой части вычисляется первичная статистика слов, по второй части вычисляются вторичные статистики — вероятности типов и типические вероятности слов, пар слов и троек слов. По третьей части вычисляются третичные статистики, которые зависят от вторичных статистик.

В выполненной в 2008 году работе вероятностный диапазон (то есть отрезок [0,1]) квантовался, то есть делился на двадцать равных интервалов. *Третичная статистика* признаков определялась уже для квантованных вероятностных типов. А именно, квантовый вероятностный тип тройки слов задается тройкой чисел $(\tilde{p}_1, \tilde{p}_2, \tilde{p}_3)$, где $\tilde{p}_i = 20[p_i/20]$ представляют собой округленные типичные вероятности слов тройки. Третичная статистика (M, F) квантового типа $\tilde{p}_1, \tilde{p}_2, \tilde{p}_3$ определяется как количество мужских и женских логов в третьей части выборки, которые содержат тройку слов с третичной статистикой данного типа.

6.6 Алгоритм Малыхина

Предположим, что всего имеется n классов, занумерованных от 1 до n.

1. Выборка случайным образом делится на три части: часть 1, часть 2 и часть 3 (в отношении 1:1:1 или другом).

2. По части 1 для каждого слова считаем его тип: $(a_1, \ldots, a_n)$; a_j — количество людей класса j, сказавших это слово. Некоторые слова будут иметь тип $(0, \ldots, 0)$.
3. По части 2 для каждого типа $a = (a_1, \ldots, a_n)$ считаем эмпирические вероятности $(p_1, \ldots, p_n)$ следующим образом: пусть $r(\texttt{user}, a)$ — доля слов типа $(a_1, \ldots, a_n)$ среди всех слов, сказанных `user`-ом, тогда

$$p_j = \frac{\sum\limits_{\text{class}(\texttt{user})=j} r(\texttt{user}, a)}{\sum\limits_{\texttt{user}} r(\texttt{user}, a)}$$

   (`user` берётся из части 2).
   Эта формула соответствует следующему эксперименту: выбираем случайно человека, в его логе берем случайное слово; считаем вероятность того, что человек принадлежит j-му классу при условии того, что получилось слово типа $(a_1, \ldots, a_n)$.
   Возможна ситуация, когда какой-то тип не встречается в части 2. Это очень маловероятно в случае деления на два класса, но возможно в случае многих классов или если классы не сбалансированы. Тогда берем оценку по Байесу на основе типа:

$$p_j = \frac{a_j + 1}{a_1 + \ldots + a_n + n}.$$

4. После выполнения шагов 1–3 имеем соответствия

$$\text{слово} \to (a_1, \ldots, a_n) \to (p_1, \ldots, p_n).$$

   Много раз (например, 10) переразбиваем выборку и повторяем шаги 2–3; в качестве результирующих вероятностей для слова берем среднее арифметическое. Отметим, что вероятности определены не для типов (они будут меняться при переразбиении), а для слов. Округляем полученные вероятности до ближайшего числа вида kh, $k \in \mathbb{Z}$ (здесь h – шаг сетки, например, $h = 0.05$).
5. Подсчет энтропии по части 3.

   - H_0: энтропия априорного распределения классов; считается без информации о словах.

$$H_0 = \sum_{j=1}^{n} \frac{N_j}{N} \log_2 \frac{N}{N_j},$$

где N_j — количество людей класса j, N — общее количество людей (люди берутся из части 3).

- H_1: энтропия класса при условии слова. На самом деле, вместо слов мы смотрим на вероятности, полученные после шага 4.

$$H_1 = \sum_{(p_1,\ldots,p_n)} \mathrm{freq}(p_1,\ldots,p_n)\cdot \mathrm{entr}(p_1,\ldots,p_n),$$

где freq и entr для вектора $p = (p_1,\ldots,p_n)$ определяются следующим образом. Пусть $r(\texttt{user}, p)$ — доля слов с вектором вероятностей p среди всех слов, сказанных `user`-ом.

$$\mathrm{freq}(p) = \frac{\sum\limits_{\texttt{user}} r(\texttt{user}, p)}{\#\{\texttt{user}\}},$$

$$\mathrm{entr}(p) = h(P_1,\ldots,P_n), \qquad \text{где}$$

$$P_j = \frac{\sum\limits_{\mathrm{class}(\texttt{user})=j} r(\texttt{user}, p)}{\sum\limits_{\texttt{user}} r(\texttt{user}, p)}$$

(везде в суммировании `user` берётся из части 3).

Фактически, P_j считаются для p_j так же, как p_j считались для a_j.

- H_1': энтропия класса при условии типа. Немного другой способ подсчета H_1. Проделываем шаги 1–2 (один раз), после чего по части 3 считаем непосредственно энтропию класса при условии типа. Все аналогично подсчету H_1, только слова «склеиваются» не по вероятностям, а по типам.

- H_2: энтропия класса при условии пары слов. Теперь вместо $p = (p_1,\ldots,p_n)$ будет пара $\{p, p'\}$.

$$H_2 = \sum_{\{p,p'\}} \mathrm{freq}(p, p')\cdot \mathrm{entr}(p, p'),$$

где freq и entr определяются следующим образом. Пусть $r(\texttt{user}, p, p')$ — доля пар слов с векторами вероятностей p и p' среди всех пар слов, сказанных `user`-ом. (Под парами мы

понимаем двухэлементные подмножества множества сказанных юзером слов.)

$$\mathrm{freq}(p,p') = \frac{\sum\limits_{\mathtt{user}} r(\mathtt{user},p,p')}{\#\{\mathtt{user}\}},$$
$$\mathrm{entr}(p,p') = h(\mathrm{PP}_1,\ldots,\mathrm{PP}_n), \qquad \text{где}$$
$$\mathrm{PP}_j = \frac{\sum\limits_{\mathrm{class}(\mathtt{user})=j} r(\mathtt{user},p,p')}{\sum\limits_{\mathtt{user}} r(\mathtt{user},p,p')}.$$

- H_3: аналогично H_2, однако возникают технические проблемы, когда слов слишком много. Поэтому, если количество слов пользователя больше некоторого c (скажем, $c = 80$), выбираем случайно $\binom{c}{3}$ троек (они могут совпадать). (Под тройками мы понимаем трехэлементные подмножества множества сказанных юзером слов. В тройке учитывается кратность, тройка p,p',p' не равна p,p,p'.)

6. Мы получили числа H_0, H_1, H_1', H_2, H_3. Все эксперименты повторяются некоторое количество раз (скажем, 10), считается среднее и дисперсия результатов.

6.7 Закон сложения вероятностей

Представленные выше методы можно применять для экспериментальной проверки законов сложения вероятностей. А именно, пусть лог содержит два слова, w_1 и w_2, которым соответствуют типические вероятности p_1 и p_2 принадлежать мужскому логу. Какова ожидаемая вероятность этого лога быть мужским? Байесовская формула сложения независимых вероятностей дает такой ответ на этот вопрос:

$$\frac{p_1 p_2}{p_1 p_2 + (1-p_1)(1-p_2)}, \tag{6.9}$$

тогда как формула $\frac{p_1+p_2}{2}$ выражает правило сложения вероятностей зависимых событий. Какая из формул ближе к действительности, можно

проверить экспериментально. Более того, можно получить экспериментальную таблицу сложения вероятностей и потом попробовать подыскать формулу, наилучшим образом интерполирующую полученные результаты.

Часть вторая

Введение

Понятие информации не является строго определенным в математическом смысле. То же самое относится и к часто употребляемым выражениям типа «преобразование информации», «количество информации» и так далее. Скорее, у нас имеется некоторое интуитивное представление о том, какие вопросы относятся к теории информации, а какие — нет. В настоящих лекциях исследуются задачи, которые, по мнению автора, имеют явный информационный «привкус».

Что касается строгих определений, то больше повезло понятию количества информации — для его уточнения имеется несколько математических моделей. Наиболее известными из них являются «информация по Хартли», энтропия Шеннона и колмогоровская сложность. В своей работе «Три подхода к определению понятия количества информации», в которой сравниваются эти три понятия (и определяется последнее из них), Колмогоров называл подход Хартли к определению количества информации «комбинаторным». Кратко, он состоит в следующем: для того, чтобы идентифицировать любой неизвестный элемент из известного конечного множества A, необходимо и достаточно $\log_2 |A|$ бит информации (игнорируя ошибки округления). Энтропия Шеннона является более сложным понятием. Она определена для случайных величин и отражает «неопределенность» случайной величины. Наконец, колмогоровская сложность измеряет количество информации в произвольном тексте (слове). В наших лекциях мы расскажем обо всех трех понятиях (и не только), и начнем мы с самого простого — информации по Хартли.

Глава 7. Информация по Хартли

Пусть имеется (известное нам) конечное множество A, элементы которого мы называем *исходами*. Чтобы идентифицировать любой элемент множества A, нам необходимо и достаточно $\log_2 |A|$ бит информации (игнорируя ошибки округления). По этой причине говорят, что исходы из множества A имеют $\log_2 |A|$ бит информации (по Хартли). В дальнейшем двойку в основании логарифма мы писать не будем — по умолчанию все логарифмы берутся по основанию 2.

Пусть нам было известно, что $x \in A$, а затем нам дополнительно сообщили, что $x \in B$, где B некоторое другое конечное множество. Теперь нам необходимо и достаточно $\log |A \cap B|$ бит, чтобы идентифицировать x. Таким образом, мы можем представлять себе, что нам сообщили $\log |A| - \log |A \cap B|$ бит информации.

Пример 1. Пусть, нам нужно узнать неизвестное упорядочение пятиэлементного множества $\{1, 2, 3, 4, 5\}$. Множество A состоит из всех перестановок этого множества и его мощность равна 5!. Пусть нам стало известно, что $1 > 2$ или $3 > 4$. Сколько бит в этой информации? Множество B состоит из всех перестановок, в которых 1 идет после 2, а 3 после 4. Количество таких перестановок, как нетрудно подсчитать, равно 90, а значит в этой информации $\log 120 - \log 90 = \log(4/3)$ бит.

Решения обсуждаемых ниже задач можно излагать как на теоретико-информационном языке, так и на на чисто комбинаторном языке. Решения первого вида более интуитивно ясные (хотя и менее строгие). Поэтому мы, как правило, будем предпочитать теоретико-информационный язык.

В предлагаемых задачах нам нужно найти неизвестный объект x из данного множества A, при этом разрешается проводить тесты определенного вида с возможными результатами ДА/НЕТ. Любой такой тест задается некоторым подмножеством B множества A и имеет два

результата: положительный ($x \in B$) и отрицательный ($x \notin B$). Положительный результат приносит нам $\log |A|/|B|$ бит информации, а отрицательный приносит $\log |A|/|\overline{B}|$ бит.

7.1 Игра в 10 вопросов

Задача 1. Имеется неизвестное число от 1 до 1000. Разрешается задавать любые вопросы с ответами ДА/НЕТ. Сколько необходимо и достаточно вопросов для отгадывания числа?

Читатель, привыкший к математической строгости, может законно возмутиться — в задаче используются неопределенные понятия «задать вопрос», «неизвестное число», «отгадать число» и так далее. Тем не менее, мы предпочитаем не определять их формально, опираясь на интуицию.

Решение.

Верхняя оценка: достаточно 10 вопросов, поскольку неизвестное число можно записать 10 двоичными цифрами и спросить про каждую из них.

Нижняя оценка: 9 вопросов недостаточно.

Теоретико-информационное доказательство. Любой тест с двоичным ответом в худшем случае приносит нам не более 1 бита информации. Действительно, любой вопрос имеет вид $\langle x \in B\,? \rangle$ для некоторого множества B. Очевидно, что для любого B либо $\log |A|/|B| \leqslant 1$, либо $\log |A|/|\overline{B}| \leqslant 1$. (Иными словами, либо $|B| \geqslant |A|/2$, либо $|\overline{B}| \geqslant |A|/2$.) Нам нужно $\log 1000$ битов информации, а каждый тест дает в худшем случае не более 1 бита информации. Поэтому в худшем случае необходимо будет задать $\lceil \log 1000 \rceil = 10$ вопросов.

То же самое доказательство, но более формально изложенное. Пусть нам каждый раз дают худший для нас ответ. Это означает следующее: в любой момент у нас имеется некоторое текущее множество $X \subset \{1, \ldots, 1000\}$, состоящее их всех $x \in \{1, \ldots, 1000\}$, которые согласованы с ответом на уже заданные вопросы. Пусть при очередном ответе на вопрос $\langle x \in B\,? \rangle$ нам отвечают ДА/НЕТ в зависимости от того, какое из множеств $X \cap B$, $X \cap \overline{B}$ больше. Тогда новое множество X не более чем вдвое меньше старого. В самом начале у нас 1000-элементное X, а в конце мы должны получить одноэлементное X, поэтому 9 вопросов недостаточно ($2^9 < 1000$).

Комбинаторное доказательство. Любой алгоритм, который решает задачу за 9 вопросов, определяет некоторую сюръективную функцию из множества всех двоичных последовательностей длины 9 в множество чисел от 1 до 1000 (ее значение на данной последовательности $a_1 \dots a_9$ равно ответу, выданному алгоритмом, если на задаваемые алгоритмом вопросы давать ответы $a_1 \dots a_9$). Поскольку такой функции не существует (по соображениям мощности), то и такого алгоритма не существует.

7.2 Упорядочивание n чисел: верхняя и нижняя оценки

Задача 2. Дано n различных по весу камешков $a_1, \dots, a_n$. Надо их упорядочить по весу, используя наименьшее количество взвешиваний (за одно взвешивание можно про любые два камешка a_i, a_j выяснить, какой из них легче).

Решение. Мы докажем, что необходимо и достаточно произвести $\log n! + O(n)$ взвешиваний. Точнее, мы установим, что любой алгоритм в худшем случае выполняет $\lceil \log n! \rceil$ взвешиваний и существует алгоритм, который упорядочивает камешки за $\lfloor \log n! \rfloor + n - 1$ взвешиваний. Точное значение наименьшего количества сравнений при больших n, насколько я понимаю, неизвестно.

Нижняя оценка. Всего существует $n!$ разных упорядочиваний n камешков. Поэтому необходимо $\lceil \log n! \rceil$ взвешиваний.

Верхняя оценка. Чтобы упорядочить камешки за $\log n! + O(n)$ взвешиваний, можно применить алгоритм последовательной вставки. А именно, пусть у нас уже имеется упорядоченный массив из k камешков. Чтобы вставить в него $(k+1)$-ый камешек, применим бинарный поиск: сравним его со средним камешком из массива, затем сравним со средним камешком из верхней или нижней половины массива (в зависимости от исхода первого взвешивания) и так далее. Для этого нам понадобится $\lceil \log(k+1) \rceil$ взвешиваний (например, при $k = 2, 3$ будет 2 взвешивания, а при $k = 4, 5, 6, 7$ — 3 взвешивания).

Начнем с упорядоченного массива из одного камешка и будем последовательно вставлять в него второй, третий, и так далее. Общее количество сравнений будет

$$\lceil \log 2 \rceil + \lceil \log 3 \rceil + \dots + \lceil \log n \rceil \leqslant \log n! + n - 1.$$

Замечание 7.1. Формула Стирлинга учит, что $n! \sim \sqrt{2\pi n} \cdot (n/e)^n$, а значит $\log n! = n \log n + O(n)$. Таким образом, необходимо и достаточно сделать $n \log n + O(n)$ взвешиваний.

При всех n, меньших 5, полученные верхняя и нижняя оценки совпадают. Действительно:

$$\lceil \log 2! \rceil = 1, \qquad \lceil \log 2 \rceil = 1,$$
$$\lceil \log 3! \rceil = 3, \qquad \lceil \log 2 \rceil + \lceil \log 3 \rceil = 3,$$
$$\lceil \log 4! \rceil = 5, \qquad \lceil \log 2 \rceil + \lceil \log 3 \rceil + \lceil \log 4 \rceil = 5.$$

Однако, уже для $n = 5$ они различаются на единицу:

$$\lceil \log 5! \rceil = 7, \qquad \lceil \log 2 \rceil + \lceil \log 3 \rceil + \lceil \log 4 \rceil + \lceil \log 5 \rceil = 8.$$

Возникает естественное желание узнать, можно ли упорядочить 5 камешков с помощью 7 взвешиваний.

7.3 Упорядочивание 5 различных чисел с помощью 7 сравнений

Задача 3. Дано 5 различных по весу камешков $a_1, \ldots, a_5$. Можно ли их упорядочить по весу с помощью 7 взвешиваний?

Решение. Общее количество исходов в этой задаче равно $5! = 120$. Каждое сравнение дает два возможных результата. Поэтому 7 сравнений дают $2^7 = 128$ возможных результатов, что хоть и больше, чем 120, но совсем ненамного. Поэтому каждое взвешивание должно делить множество всех исходов на почти равные части (в информационных терминах: при обоих исходах взвешивание должно давать примерно 1 бит информации о неизвестном порядке). Мы будем пользоваться этим обстоятельством для отсечения заведомо неподходящих вариантов сравнений. В результате мы либо установим, что искомого алгоритма нет, либо найдем его.

При первом сравнении у нас в сущности нет выбора, поскольку все варианты симметричны. Допустим, сравниваем a_1 и a_2. Без ограничения общности, пусть оказалось $a_1 < a_2$. Остается 60 возможных исходов и $2^6 = 64$ возможных результатов оставшихся 6 взвешиваний.

Теперь у нас имеется 2 принципиально разных варианта продолжения: сравнить два новых камешка или один новый с одним старым. Покажем, что второй вариант не годится. Пусть, например, мы сравнили

a_1 с a_3. Такое сравнение разделит множество из оставшихся возможными 60 порядков на 20 и 40. Действительно, если $a_3 < a_1 < a_2$, то имеется ровно 20 возможных исходов (четырьмя способами можно вставить a_4 в этот упорядоченный массив и для каждого из этих способов можно 5 способами вставить a_5 в полученный упорядоченный массив). Отсюда следует, что при другом результате сравнения ($a_1 < a_3$) остается $60 - 20 = 40$ исходов. Поскольку $40 > 2^5$, с помощью оставшихся 5 взвешиваний невозможно будет найти порядок.

Итак, во втором взвешивании мы должны сравнить два новых камешка, например, a_3 и a_4. Такое сравнение разделит оставшиеся исходы поровну на 30 и 30. Без ограничения общности, пусть оказалось $a_3 < a_4$.

Каким должно быть третье взвешивание? Докажем, что пятый камешек не может в нем участвовать. Допустим, что третьим взвешиванием мы сравним новый камешек a_5 с одним из предыдущих. По симметрии, неважно, с чем его сравнить. Пусть, скажем, мы сравнили a_1, a_5. Такое сравнение разделит оставшиеся 30 исходов на 10 и 20. Действительно, существует 10 порядков, для которых $a_5 < a_1 < a_2$ и $a_3 < a_4$ (слить два упорядоченных массива из соответственно k, l элементов в один упорядоченный массив из $k + l$ элементов можно $(k+1)+(k+2)+\ldots+(k+l) = C_{k+l}^{l}$ способами). Поэтому слить массивы $a_5 < a_1 < a_2$ и $a_3 < a_4$ в один можно $C_{2+3}^{2} = 10$ способами. Поскольку $20 > 16$, такое сравнение нам не подходит.

Таким образом, при третьем взвешивании мы должны сравнить между собой какие-то из a_1, a_2, a_3, a_4. Есть два принципиально разных варианта: сравнить между собой два максимальных (или минимальных) элемента и сравнить между собой максимальный и минимальный элементы. Второй вариант очевидно плох, поскольку разделяет множества из оставшихся 30 исходов на 5 и 25. Действительно, если оказалось, что максимальный элемент меньше минимального, то мы получаем полностью упорядоченный массив из $a_1, \ldots, a_4$, в который можно вставить a_5 пятью способами.

Итак, в третьем взвешивании мы должны сравнить между собой два максимальных или два минимальных элемента. Пусть, например, мы сравниваем a_1, a_3. Без ограничения общности, пусть $a_1 < a_3$. Осталось 15 возможных исходов и 4 взвешивания.

Докажем, что в четвертом взвешивании необходимо сравнивать a_5. Действительно, при сравнении a_2 с a_3 или a_4 нам может повезти и мы

узнаем полностью, как упорядочены $a_1, \dots a_4$ (при сравнении a_2 с a_3 хороший для нас результат $a_2 < a_3$, а при сравнении a_2 с a_4 хороший для нас результат $a_2 > a_4$). В этом случае останется 5 возможных исходов. Значит, в случае невезения их останется $10 > 8$.

С чем нужно сравнивать a_5? Ясно, что не с минимальным элементом a_1 (такое сравнение разделит оставшиеся 15 исходов на 3 и 12) и не с максимальными a_2, a_4 (поскольку в случае везения останется соответственно 6 и 4 возможных исхода, а при невезении — 9 и 11). Итак, методом исключения мы убедились, что в четвертом взвешивании мы должны сравнить a_3 и a_5. Нетрудно убедиться, что это сравнение разделит оставшиеся 15 исходов на 8 и 7 и в каждом из двух результатов взвешивания можно с помощью оставшихся 3 сравнений найти порядок на $a_1, \dots, a_5$ целиком. Поэтому ответ в задаче положительный.

7.4 Поиск фальшивой монетки из 81 за 4 взвешивания

Задача 4. Дана 81 монетка, из них ровно одна фальшивая, которая легче всех настоящих (настоящие монеты имеют одинаковый вес). Сколько нужно взвешиваний, чтобы на чашечных весах найти фальшивую монетку? (На чашечных весах можно сравнивать по весу любые непересекающиеся группы монеток. Результатов сравнения три — равенство, больше и меньше.)

Решение. Нижняя оценка: необходимо не менее 4 взвешиваний. Действительно, на этот раз каждый тест имеет 3 исхода (больше, меньше и равно). Поэтому в худшем случае один тест приносит не более $\log 3$ бит информации, а нам нужно $\log 81 = 4 \log 3$ бит. Рассуждая так же, как и в первой задаче (при худших для нас ответах текущее множество X после каждого теста уменьшается не более чем втрое), убеждаемся, что необходимо 4 взвешивания.

Верхняя оценка: 4 взвешиваний достаточно. Делим монетки на 3 равные группы (по 27 монеток) и сравниваем первые 2 группы. В случае равенства их весов, фальшивая монета находится в третьей группе. В случае неравенства — она в более легкой группе. Далее мы применяем эту стратегию рекурсивно еще 3 раза.

7.5 Поиск фальшивой монетки из 12 за 3 взвешивания

Задача 5. Дано 12 монеток, из них ровно одна фальшивая, которая может быть как легче, так и тяжелее настоящих. Надо за 3 взвешивания на чашечных весах найти фальшивую монетку и узнать, легче она или тяжелее остальных.

Решение. Общее количество исходов в этой задаче равно 24 (любая из 12 монеток может быть фальшивой, причем она может быть легче или тяжелее). Каждое взвешивание дает три возможных результата (равенство, больше или меньше), поэтому 3 взвешивания дают 27 возможных результатов, что хоть и больше, чем 24, но совсем ненамного. Поэтому каждое взвешивание должно делить множество всех исходов на три почти равных части (в информационных терминах: каждое взвешивание при любом его результате должно давать примерно $\log 3$ битов информации). Мы будем пользоваться этим обстоятельством для отсечения заведомо неподходящих вариантов взвешивания.

Прежде всего заметим, что глупо сравнивать разные по количеству группы монеток. Действительно, можно предполагать, что вес фальшивой монетки примерно равен весу настоящей. При этом предположении группа в которой больше монеток всегда тяжелее группы, в которой их меньше, поэтому такие взвешивания вообще не дают информации.

Итак, первое взвешивание должно быть сравнением двух групп из k монеток (для некоторого k). Нетрудно понять, что k должно быть равно 4. В самом деле, при равенстве весов групп остается $2(12-2k)$ возможных исходов (остается $12-2k$ невзвешенных монеток, любая из них может оказаться фальшивой, причем она может быть, как легче, так и тяжелее остальных). А при неравенстве весов групп остается $2k$ возможных исходов (любая из взвешенных монет может быть фальшивой). Поэтому оба числа $2(12-2k)$ и $2k$ не должны превосходить 9, что возможно только при $k=4$.

Итак, в первом взвешивании мы должны сравнить две группы из 4 монеток. Если их веса равны, то остается 8 возможных исходов (любая из невзвешенных 4 монеток может быть фальшивой, причем легче или тяжелее остальных). По симметрии, при любом из двух других результатов взвешивания остается также 8 возможных исходов.

Разберем три результата первого взвешивания по отдельности. Пусть оказалось, что веса двух групп равны. Тогда все взвешенные

монеты настоящие. Остальные монеты будем называть *подозрительными*. Сколько подозрительных монет должны участвовать во втором взвешивании? Мы утверждаем, что ровно три. Действительно, их должно быть не меньше трех, поскольку при равенстве весов сравниваемых групп любая из k невзвешенных подозрительных монет может быть фальшивой и может быть легче или тяжелее: остается не менее $2k$ вариантов и всего одно взвешивание. Поэтому $2k \leqslant 3$ и $k \leqslant 1$. С другой стороны, все 4 подозрительных монеты не могут участвовать во втором взвешивании: если результатом взвешивания будет неравенство весов, то любая из подозрительных монет может быть фальшивой (остается 4 варианта).

Итак, во втором взвешивании должны участвовать ровно 3 подозрительных монеты. Можно проверить, что любое взвешивание двух равночисленных групп, содержащих вместе ровно 3 подозрительных монетки, годится. Например, можно сравнить две подозрительных монетки с одной подозрительной и одной настоящей. При равенстве весов у нас остается 2 возможных исхода. Если первая группа тяжелее второй, то остается 3 возможных исхода (фальшивой может быть любая из трех взвешенных подозрительных, а больше она или меньше определяется тем, на какой из чашек она лежит). Если первая группа легче, то по аналогичным причинам остается 3 возможных исхода. При каждом из трех результатов с помощью третьего взвешивания нетрудно найти исход: в первом случае фальшивой является оставшаяся подозрительная монетка и надо сравнить ее с настоящей; во втором и третьем случаях достаточно сравнить между собой две подозрительных монетки, лежавших во втором взвешивании на одной чашке весов.

Осталось разобрать случай, когда первое взвешивание обнаружило, что одна группа из четырех монеток тяжелее другой группы из четырех монеток. Теперь у нас имеется 8 подозрительных монеток, 4 *легких* и 4 *тяжелых*. Максимум 3 из них могут не участвовать во втором взвешивании (поскольку любая из них может быть фальшивой). Есть ли верхняя оценка на количество подозрительных монеток, участвующих во втором взвешивании? Такая оценка получается из следующего соображения. Пусть во втором взвешивании на одной чашке весов ровно k легких подозрительных монеток, а на другой чашке ровно l тяжелых. Тогда $k+l$ должно не может превосходить 3. Действительно, если первая чашка перевесит вторую, то любая из этих $k+l$ монеток может быть фальшивой, и у нас остается $k+l$ возможных исходов.

Любое второе взвешивание, удовлетворяющее обоим этим требованиям, годится. Например, положим во втором взвешивании на каждую

из чашек по две тяжелых и по одной легкой подозрительной монетке. При равенстве весов остается два варианта (фальшивая монета легче и она находится среди не участвовавших во втором взвешивании 2 легких подозрительных монет; ее можно найти, сравнив одну из этих 2 монет с настоящей). По симметрии, при каждом из двух других результатов остается 3 возможных исхода и их легко различить, сравнив между собой две тяжелых монеты с той чашки, которая перевесит.

7.6 Цена информации

Задача 6. Имеется неизвестное число от 1 до n (где $n \geqslant 2$). Разрешается задавать любый вопросы с ответами ДА/НЕТ. При этом при ответе ДА мы платим 1 рубль, а при ответе НЕТ — 2 рубля. Сколько необходимо и достаточно заплатить для отгадывания числа?

Решение.

Верхняя оценка. На этот раз информация стоит денег и представляется разумным задавать вопросы так, чтобы отрицательные ответы приносили вдвое больше информации, чем положительные. Таким образом один бит информации будет нам обходиться в некоторое постоянное число c рублей (независимо от неизвестного числа) и при любом задуманном числе мы заплатим $c \log n$ рублей.

Это означает, что каждый раз разумно задавать такой вопрос $\langle x \in T\,? \rangle$, чтобы для текущего множества X было выполнено

$$2(\log|X| - \log|X \cap T|) = \log|X| - \log|X \cap \overline{T}|.$$

Это равенство означает, что T делит X в отношении золотого сечения: $|X \cap T| = \alpha|X|$, $|X \cap \overline{T}| = (1-\alpha)|X|$, где $\alpha^2 = 1-\alpha$, $\alpha = (\sqrt{5}-1)/2$. При ответе ДА мы получим $\log|X| - \log\alpha|X| = -\log\alpha$ бит информации, а при ответе НЕТ — $\log|X| - \log(1-\alpha)|X| = -\log(1-\alpha) = -2\log\alpha$ бит информации. В любом случае мы заплатим $c = 1/(-\log\alpha)) \approx 1.44$ рублей за один бит, а значит в целом — $\log n/(-\log\alpha)$ рублей.

В этом рассуждении мы игнорировали проблему округления. На самом деле мы не можем разделять текущее множество исходов в отношении золотого сечения, поскольку последнее иррационально. Поэтому вместо множества X будем хранить отрезок $Y \subset [1, n]$ с действительными концами. Текущее множество возможных исходов будет равно множеству всех целых чисел, принадлежащих отрезку Y. Очередной

вопрос будет иметь вид $\langle x \in T\,?\rangle$, где T — это начало Y длины $\alpha|Y|$, и новое множество Y' будет равно T или $Y \setminus T$ (в зависимости от ответа на вопрос). При любом ответе на этот вопрос отношение количества заплаченных денег к $\log|Y|/|Y'|$ в точности равно c. Мы будем задавать вопросы до тех пор, пока длина отрезка Y не станет меньшей 1 (в этом случае мы знаем x). Поскольку после каждого вопроса длина отрезка уменьшается максимум в $1/\alpha^2$ раз, длина последнего отрезка будет не меньше α^2. Значит, в целом длина отрезка Y сократилась не более чем в $(n-1)/\alpha^2$ раз. Поскольку мы платим по c рублей за сокращение $\log|Y|$ на 1, общая уплаченная сумма будет не больше $c\log((n-1)/\alpha^2) = c\log(n-1) + 2$. Поскольку заплаченная сумма является целым числом, мы заплатим при любом исходе не больше $\lfloor c\log(n-1)\rfloor + 2$ рублей.

Нижняя оценка. Попробуем теперь «обратить» это рассуждение. Пусть имеется некоторый алгоритм и надо указать такое $x = 1, \ldots, n$, для которого алгоритм заплатит не меньше $\log n/(-\log\alpha)$ рублей, где $\alpha = (\sqrt{5}-1)/2$. При каждом тесте будем выбирать наиболее невыгодный для алгоритма ответ. А именно, пусть X — множество исходов, оставшихся возможными к данному моменту, а $\langle x \in T\,?\rangle$ — текущий вопрос. Выбираем ответ ДА/НЕТ в зависимости от того, какое из двух чисел $1/(\log|X| - \log|X\cap T|)$, $2/(\log|X| - \log|X\cap\overline{T}|)$ больше (эти числа равны количеству рублей, заплаченных за один бит полученной информации). При любых X, T хотя бы одно из этих чисел не меньше $c = 1/(-\log\alpha)$. Действительно, мы уже видели, что при $\log|X\cap T| = \alpha|X|$ эти числа равны. Одно из них убывает с ростом $\log|X\cap T|$, а другое возрастает. Поэтому наибольшее из них всегда не меньше $c = 1/(-\log\alpha)$.

Таким образом, мы заставляем алгоритм платить не менее $c = 1/(-\log\alpha)$ рублей за бит. Поскольку в целом алгоритм получает $\log n$ бит информации, он в худшем случае заплатит $\lceil c\log n\rceil$ рублей.

При $n = 2, 3$ полученные верхняя и нижняя оценки совпадают, а при остальных n отличаются на 1.

7.7 Задачи для самостоятельной работы

Задача 7. Среди одинаковых на вид n камешков один радиоактивен. Имеется счетчик Гейгера, позволяющий выяснить, есть ли радиоактивный камешек в любой группе камешков. Докажите, что необходимо и

достаточно $\log n$ применений счетчика Гейгера для нахождения радиоактивного камня.

Задача 8. Дано пять монет разного веса и известно, что первая монета легче второй, а третья легче второй и четвертой. Имеются весы, позволяющие сравнить по весу любые две монеты. Доказать, что для того, чтобы упорядочить по весу все монеты, необходимо и достаточно 5 взвешиваний.

Задача 9. Имеются 24 монеты, среди которых ровно одна фальшивая (неизвестно какая). Все настоящие монеты одного веса, а фальшивая легче или тяжелее. На чашечных весах можно сравнивать по весу любые две группы монет. Нужно найти фальшивую монету и выяснить, легче она или тяжелее. Докажите, что необходимо и достаточно сделать 4 взвешивания.

Задача 10. Даны две группы камешков, причем камешки в каждой группе упорядочены по весу. В первой группе n камешков, а во второй — m. Требуется упорядочить все камешки по весу. Докажите, что для этого достаточно сделать $m + n - 1$ взвешиваний. Докажите, что необходимо сделать $\log C_{m+n}^m$ взвешиваний.

Задача 11. Имеется неизвестное число от 1 до n (где $n > 1$). Разрешается задавать любые вопросы с ответами ДА/НЕТ. При этом при ответе ДА мы платим 2 рубля, а при ответе НЕТ — 3 рубля. Сколько необходимо и достаточно заплатить для отгадывания числа?

Задача 12. (сложная) Первый игрок задумал число x от 1 до n. На своем ходу второй игрок задает ему вопросы вида: «верно ли, что $x \leqslant k$» выбирая конкретное k по своему усмотрению. После каждого такого хода первый игрок обязан ему ответить на этот вопрос утвердительно, если в самом деле задуманное число меньше или равно k, и может дать любой ответ (ДА/НЕТ) иначе. В конце игры второй игрок должен найти x при условии, что общее количество ошибочных ответов, данных первым игроком, не превысило сотую часть от общего количества заданных ему вопросов. Докажите, что второй игрок может справиться с этой задачей, задав не более $O(\log n)$ вопросов. [Указание. Нужно запустить алгоритм бинарного поиска, модифицированный следующим образом. Каждая вершина дерева поиска соответствует некоторой гипотезе $x \in [a, b]$. При этом мы имеем гарантию, что $x \geqslant a$. На каждом шаге алгоритма мы сначала спрашиваем, верно ли, что $x \leqslant b$. Если

ответ опровергают текущую гипотезу, то мы возвращаемся в дереве поиска к отцу текущей вершины. А иначе делим отрезок пополам и переходим к одному из сыновей текущей вершины, как в обычном бинарном поиске (если мы уже находимся в некотором листе, то деление отрезка пополам пропускаем). Рассмотрим следующую «потенциальную» функцию: (расстояние в дереве поиска от текущей вершины до цели) $-\,2\cdot$(количество неправильных ответов, полученных к текущему моменту). Нетрудно видеть, что если мы находимся не в целевом листе (дерева поиска), то значение потенциальной функции убывает не менее чем на 1 в ходе выполнения одного шага алгоритма. В начале значение этой функции равно $\log n$, а в конце оно не меньше $-2\cdot$(количество неправильных ответов) $\geqslant -0.02\cdot$(количество заданных вопросов). На каждом шаге мы задаем не более двух вопросов. Поэтому если мы ни разу не попали в целевой лист, то уменьшение целевой функции не меньше половины количества заданных вопросов. Это доказывает, что задав $\log n/0.48$ вопросов, мы хотя бы однажды попадем в целевой лист. Осталось заметить, что однажды попав в него, мы никогда из него не выйдем. Действительно, при $x = a = b$ на вопрос $x \leqslant b$ противник обязан отвечать положительно.]

Задача 13. (сложная) Докажите, что в условиях предыдущей задачи даже если первый игрок может врать в случае $x \leqslant k$, но общее количество неверных ответов по-прежнему не превосходит одной сотой количества вопросов, то второй игрок может узнать x, задав не более $O(\log n)$ вопросов. [Указание. Алгоритм из предыдущей задачи теперь не годится по двум причинам: во-первых, теперь нужно проверять и левую границу текущего отрезка, во-вторых, противник может привести нас в целевой лист, затем увести из него. Можно его модифицировать так: во-первых, перед делением пополам нужно проверять оба конца отрезка, во-вторых, надо увеличить количество вопросов примерно втрое. Тогда большую часть времени мы проведем в целевом листе дерева поиска (иначе потенциальная функция уменьшится слишком сильно) и таким образом сможем его найти.]

Глава 8. Логика знания

Пусть A — некоторое множество. Как и в предыдущем параграфе, элементы A — это неизвестные нам исходы, о которых имеется некоторая информация. В логике знания его элементы принято называть *мирами*. Пусть f — некоторая функция из A в некоторое множество I (значение $f(x)$ надо понимать, как известную нам информацию о мире x). Для логики знания неважно, какие значения принимает f, важно лишь, на какие классы эквивалентности значения f разбивают A (каждый класс эквивалентности состоит из всех исходов с одинаковым значением $f(x)$). Итак, задано множество A и некоторое отношение эквивалентности $\sim$ на множестве A.

Пример 1. Пусть мирами являются целые числа от 1 до 5, а известная нам информация есть остаток от деления на 3. Тогда классами эквивалентности будут $\{1,4\}$, $\{2,5\}$, $\{3\}$.

Пусть B — это некоторое утверждение о мирах. Формально, B — это некоторое подмножество A и мы считаем утверждение B истинным в мире x, если $x \in B$, и ложным иначе. Говорят, что в мире x мы *знаем*, что B истинно, если все y, эквивалентные x, принадлежат B (то есть B включает целиком весь класс эквивалентности элемента x). Обозначение: $x \vdash B$. Из определения следует, что если в мире x мы знаем, что B верно, то и в самом деле B истинно в этом мире. Возвращаясь к нашему примеру, в мирах $1, 4$ и 3 мы знаем, что наш мир меньше 5, а в мирах 2, 5 мы этого не знаем.

К утверждениям о мирах можно применять обычные логические связки. А именно, утверждение «A и B» означает пересечение соответствующих множеств миров, утверждение «A или B» означает объединение соответствующих множеств миров, а утверждение «неверно, что A» означает дополнение к множеству миров, задающих A. Утверждение «мы не знаем, что B» означает в точности то же самое, что

«неверно, что мы знаем, что B». В указанном выше примере мы не знаем, что мир меньше пяти в мирах 2 и 5. Заметим, что в обычной речи в выражении «некто не знает, что B» мы часто подразумеваем, что B истинно. В задачах, приводимых ниже, это не подразумевается: множество миров, в которых мы не знаем, что B истинно, есть дополнение к множеству миров, в которых мы знаем, что B истинно.

Из определения непосредственно следует, что если в мире x мы знаем, что B, то в мире x мы знаем, что мы это знаем. Точно так же, если в мире x мы не знаем, что B, то в мире x мы знаем, что мы этого не знаем.

Усложнение: пусть опять имеется некоторое множество миров A и k человек, у каждого из которых своя информация о мире. Это означает, что заданы k отношений эквивалентности на A, которые мы будем обозначать, через $\sim_1, \ldots, \sim_k$. Тогда определены понятия: в мире x человек i знает, что B, а значит и более сложные понятия (например, Петя знает, что Вася не знает, что B).

Пример 2. Пусть мирами являются целые числа от 1 до 5, известная Алёне информация есть остаток от деления на 3, а известная Боре информация — остаток от деления на 2. Тогда Алёниными классами эквивалентности будут $\{1,4\}$, $\{2,5\}$, $\{3\}$, а Бориными — $\{1,3,5\}$, $\{2,4\}$. В мире 1 Алёна знает, что мир меньше 5, а вот Боря этого не знает. А в мире 4 оба это знают. В мире 1 Алёна не знает, что Боря не знает, что мир меньше 5. (Действительно, в мире 4, эквивалентном с точки зрения Алёны миру 1, Боря это знает.)

8.1 Карточки с цифрами

Задача 14. Имеется карточка, на одной стороне которой написано некоторое целое неотрицательное число n, а на другой — $n+1$. Алёна видит одну из двух сторон карточки, а Боря — другую. Происходит следующий диалог. Алёна: я не знаю, что на твоей стороне. Боря: я не знаю, что на твоей стороне. После этого, Алёна опять говорит, что не знает, что на стороне Бори, и Боря говорит, что не знает, что на стороне Алены. И так продолжается 10 раз (то есть, всего они делают 10 пар сообщений). После этого Алёна говорит, что теперь она знает Борино число. Какая у них карточка?

Ответ: у них может быть любая из карточек $(19,20)$, $(20,21)$.

Решение. Множество A возможных миров в этой задаче состоит из всех пар натуральных чисел вида $(n, n+1)$, $(n+1, n)$ (первое число — Алёнино, а второе — Борино). Утверждение о том, что Алёна знает Борино число означает, что она знает, что мир равен тому, чему он в самом деле равен. То есть это утверждение истинно в мире x тогда и только тогда, когда класс эквивалентности мира x (с точки зрения Алёны) одноэлементен.

Изначально, одноэлементный класс (с точки зрения Алёны) единствен — это $\{(0,1)\}$. Из-за чего могут возникнуть новые одноэлементные классы? Из-за того, что сообщения Бори сокращают множество возможных миров (нужно удалить те миры, в которых это сообщение ложно) и поэтому уменьшаются некоторые из Алёниных классов эквивалентности.

Первое сообщение Алёны равносильно тому, что Алёнино число положительно, поскольку единственный мир, в котором Алёна знает Борино число, есть (0,1). Таким образом, после первого сообщения Алёны остаются возможными все миры, кроме мира (0,1). Теперь уже существует два мира, (1,0) и (2,1), в которых Боря знает Алёнино число. Поэтому после первого сообщения Бори остаются возможными все миры, кроме (0,1), (1,0) и (2,1). Рассуждая по индукции, нетрудно установить, что после i-ого сообщения Алёны возможными мирами являются все миры квадранта $\{(m,n) \in A \mid m \geqslant 2i-1,\ n \geqslant 2i-2\}$ (и только они), а после i-ого сообщения Бори, возможными мирами являются все миры квадранта $\{(m,n) \in A \mid m \geqslant 2i-1,\ n \geqslant 2i\}$ (и только они). Действительно, базу индукции (после первого сообщения Алёны возможные миры составляют множество $\{(m,n) \in A \mid m \geqslant 1,\ n \geqslant 0\}$) мы уже проверили.

Индуктивный переход. Пусть после после i-ого сообщения Алёны возможные миры составляют множество $\{(m,n) \in A \mid m \geqslant 2i-1,\ n \geqslant 2i-2\}$. Классы эквивалентности каких миров (с точки зрения Бори) одноэлементны? Нетрудно проверить, что таких мира ровно два — $(2i-1, 2i-2)$ и $(2i, 2i-1)$. В этих и только этих мирах Боря знает число Алёны. Вычитая эти миры из множества возможных миров, мы получим множество $\{(m,n) \in A \mid m \geqslant 2i-1,\ n \geqslant 2i\}$, что и требовалось доказать. Теперь Алёна знает Борино число в мирах $(2i-1, 2i)$, $(2i, 2i+1)$, и вычитая эти миры, мы получим множество $\{(m,n) \in A \mid m \geqslant 2i+1,\ n \geqslant 2i\}$. Это завершает индуктивный переход.

Таким образом, после десятого сообщения Бори остаются возмож-

ными все миры из $\{(m,n) \in A \mid m \geqslant 19,\ n \geqslant 20\}$, и Алена знает Борино число в мирах $(19,20)$, $(20,21)$.

8.2 Задача о шляпах

Задача 15. Три джентльмена заходят в комнату, в которой имеется три красных и две белых шляпы (которые они видят). Внезапно выключается свет, и в темноте каждый джентльмен надевает одну шляпу, цвета которой он не видит. После включения света происходит следующий диалог. Первый: я не знаю, какая на мне шляпа. Второй: и я тоже не знаю, какая на мне шляпа. Третий: а я теперь знаю, какая на мне шляпа. Какая шляпа на третьем джентльмене?

Решение. Множество A возможных миров в этой задаче состоит из всех слов длины три в алфавите {к, б}, кроме слова ббб (поскольку белых шляп всего две). Классы эквивалентности с точки зрения первого джентльмена есть {кбб}, {ббк,кбк}, {бкб,ккб} и {бкк,ккк}. Аналогичным образом определяются классы эквивалентности с точки зрения второго и третьего. (Можно себе представлять миры как вершины куба. Тогда эквивалентные с точки зрения первого миры соединяются ребрами куба, параллельными оси абсцисс. Эквивалентность второго задается ребрами, параллельными оси ординат, а третьего — оси аппликат.)

В мире x джентльмен знает цвет своей шляпы, если класс мира x одноэлементен. Поэтому единственным миром, в котором первый знает цвет своей шляпы является мир ккб, и после сообщения первого джентльмена этот мир удаляется из множества возможных миров. Теперь второй джентльмен знает цвет своей шляпы в двух мирах, ккб и бкб. Удалив и эти миры, мы получим четыре мира, ббк, бкк, кбк и ккк, в каждом из которых третий знает цвет своей шляпы (красный).

8.3 Задачи для самостоятельной работы

Задача 16. Вова загадал число $x = 0,1,2,3,4,5$ и сообщил Алене $\lfloor x/2 \rfloor \pmod 3$, а Боре — $\lceil x/2 \rceil \pmod 3$. При каких значениях x Боря знает, что $x > 1$? При каких значениях x Алена знает, что он это знает? При каких значениях x Алена знает, что он этого не знает?

Задача 17. Вова загадал число $x = 0,1,2,3,4,5$ и сообщил Алене $\lfloor x/2 \rfloor \pmod 3$, а Боре — $\lceil x/2 \rceil \pmod 3$. При каких значениях x Алена знает,

что $x < 3$? При каких значениях x Боря знает, что она это знает? При каких значениях x Боря знает, что она этого не знает?

Задача 18. Вова загадал число $x = 0, 1, 2, 3, \ldots, 10$ и сообщил Алене $\lfloor x/2 \rfloor$, а Боре — $\lceil x/2 \rceil$. При каких значениях x Алена знает, что x не делится на 3? При каких значениях x Боря знает, что она это знает? При каких значениях x Боря знает, что она этого не знает?

Задача 19. В условиях задачи 14 происходит следующие диалоги.
(а) Алёна: «Я знаю, что ты не знаешь моего числа». Боря: «Теперь я знаю твое число».
(б) Алёна: «Я не знаю твоего числа». Боря: «А я это и без тебя знал». Алёна: «Ага, теперь я знаю твоё число».
В какой ситуации возможен каждый из диалогов (какая карточка у Алёны и Бори)?

Задача 20. В условиях задачи 15 происходит следующие диалоги.
(а) Первый: «Я знаю, что никто не знает, какая на нём шляпа». Второй: «А я этого не знал».
(б) Первый: «Я знаю, что третий не знает цвета своей шляпы». Второй: «Даже теперь я не знаю цвета моей шляпы».
В какой ситуации возможен каждый из диалогов?

Задача 21. (Задача о неверных женах.) В некотором городе имеется k неверных жен (и какое-то количество верных жен). Муж каждой из неверных жен не знает, что его жена ему неверна, но знает, что все остальные неверные жены изменяют своим мужьям. В некоторый день в 12 часов дня объявляется во всеуслышание, что в городе имеются неверные жены, и что любой муж, который знает, что его жена неверна, обязан на следующий день в 12 часов дня ее прилюдно наказать (по любой другой причине прилюдное наказание запрещено). Докажите, что через k дней, но не раньше, все неверные жены будут наказаны. [Указание. Если $k > 1$, то каждый и без сделанного объявления знает, что в городе есть неверные жены. Однако, объявление и в такой ситуации не бессмысленно, поскольку оно гарантирует, что все знают, что $k > 0$. Например, при $k = 2$ муж любой из двух неверных жен без такого объявления не был бы уверен, что муж другой неверной жены знает, что $k > 0$.]

8.3.1 Отступление: парадокс неожиданной контрольной

Формулировка парадокса. Учитель сообщил своим ученикам, что на следующей неделе даст контрольную, но она будет неожиданной: накануне контрольной ученики не будут знать, что на следующий день она состоится. Ученики рассуждают следующим образом. «В субботу контрольной не будет, поскольку иначе накануне мы знали бы об этом. Раз в субботу контрольной точно не будет, то и в пятницу ее не будет, поскольку иначе мы знали бы в четверг вечером, что она будет на следующий день.» Рассуждая подобным образом, они пришли к выводу, что контрольной вообще не будет. Однако учитель дал контрольную в пятницу, тем самым выполнив своё обещание: контрольная была неожиданной.

Где здесь ошибка? В самом ли деле учитель выполнил свое обещание? И правильно ли рассуждали ученики?

Отличие этой задачи от предыдущих в том, что в ней идет речь о знании в будущем. Трудность в том, что знания учеников увеличиваются после сделанного учителем сообщения. Поэтому можно было бы просто сказать, что неясно, о каком именно знании идет речь в сообщении учителя и тем самым задача не поставлена корректно.

Однако ничто нам не мешает рассмотреть оба возможных уточнения, используя формальное определение знания. У нас имеется шесть возможных миров, соответствующих шести рабочим дням недели. В мире «D» контрольная состоялась в день недели D. Накануне каждого дня недели D у нас имеется свое отношение эквивалентности $\sim_D$. По отношению $\sim_D$ все миры, предшествующие D, образуют одноэлементные классы, а все миры после D (включая D) эквивалентны друг другу.

Первое уточнение. Под знанием во фразе

(A) *накануне контрольной вы не будете знать, что она на следующий день*

имеется в виду знание до этого сообщения, то есть то понятие знания, которое задается выше определенными отношениями эквивалентности. Тогда учитель своё обещание выполнил, то есть сообщение (A) истинно в реальном мире, а ученики рассуждали неправильно, потому что имели в виду совсем другое уточнение.

Второе уточнение. Под знанием в сообщении (A) понимается знание после него. То есть (A) понимается как конъюнкция многих утвер-

ждений: $A_1, A_2, A_3, \ldots$, определяемых по индукции. Отношения эквивалентности $\sim_D$, с помощью которых определяется A_{i+1}, определены на множестве всех миров, в которых истинны все суждения $A_1, \ldots, A_i$, и получаются просто сужением исходных отношений на это множество. Суждение A_{i+1} имеет смысл только в мирах этого множества и истинно в мире D, если существует мир, отличный от D и эквивалентный D относительно $\sim_D$. В соответствии с определением, утверждение A_1 истинно во всех мирах, кроме «субботы», утверждение A_2 в мире «суббота» бессмысленно, в мире «пятница» ложно, а в остальных мирах истинно, и так далее. В частности, все утверждения $A_7, A_8, \ldots$ бессмысленны во всех мирах.

Такое понимание сообщения учителя уже чревато парадоксом, поскольку его составные части в некоторых мирах бессмысленны. Если все-таки разрешить бессмысленные суждения и считать конъюнкцию ложного и бессмысленного суждений ложной (а не бессмысленной), то утверждение (A) становится ложным во всех мирах, и ученики рассуждали правильно.

Можно только заметить, что поняв, что суждение учителя ложно во всех мирах, ученики могли бы сообразить, что учитель не сообщил им никакой информации о реальном мире. Знание в бытовом понимании этого слова предполагает возможность делать правильные предсказания. В частности, в бытовом понимании знание не монотонно: после принятия на веру ложного суждения знания уменьшаются, а не увеличиваются. Приняв суждение учителя на веру, ученики могут вывести из него что угодно: как то, что контрольная будет в пятницу, так и то, что ее в пятницу не будет. Поэтому после принятия на веру суждения (A) ученики знают всё в математическом понимании знания, но не знают ничего в бытовом понимании знания: руководствуясь ложным суждением, они не могут ничего предсказать.

Глава 9. Коммуникационная сложность

Пусть имеются три конечных множества X, Y, Z и некоторая функция $f : X \times Y \to Z$. В задачах о коммуникационной сложности Алисе дается некоторое $x \in X$, а Бобу — некоторое $y \in Y$. При этом Алиса не знает y, а Боб не знает x, но оба знают X, Y, Z и f. Обмениваясь информацией, они должны совместно найти $f(x, y)$. Точнее, каждый из них должен узнать значение $f(x, y)$. Мы интересуемся количеством битов, которое необходимо и достаточно для этого передать. Эта величина обозначается через $D(f)$.

Формально, $D(f)$ определяется с помощью понятия *коммуникационного протокола* (в коммуникационной сложности алгоритмы общения называются протоколами). *Коммуникационным протоколом вычисления* $f : X \times Y \to Z$ называется конечное дерево с корнем и двоичным ветвлением. Вершины этого дерева означают текущее состояние вычисления. Внутренние вершины этого дерева помечены буквами А (Алиса) и Б (Боб), означающими очередь хода. Каждая вершина v с пометкой А помечена еще некоторой функцией $g_v : X \to \{0, 1\}$, которая говорит Алисе, какой бит (0 или 1) нужно посылать, если вычисление находится в состоянии v. Аналогично, каждая вершина v с пометкой Б помечена еще и некоторой функцией $g_v : Y \to \{0, 1\}$. Листы дерева соответствуют состояниям вычисления, в которых Алиса и Боб уже знают значение функции. Каждый лист v дерева помечен двумя функциями $g_v : X \to Z$ и $h_v : Y \to Z$, сообщающими Алисе и Бобу значение функции.

Вычисление в соответствии с протоколом P устроено так. Поставим фишку в корень дерева и будем двигать ее вдоль дерева вверх следующим образом. Если фишка в данный момент находится во внутренней вершине v, помеченной функцией g_v и буквой А, то в следующий момент она передвигается в левого сына v, если $g_v(x) = 0$, и в правого сына v, если $g_v(x) = 1$. Если в данный момент фишка находится в

вершине, помеченной буквой Б, то действуем аналогичным образом: применяем g_v к y. Двигаем фишку в соответствии с этими правилами до тех пор, пока не окажемся в листе дерева, скажем v. Результат вычисления равен $g_v(x)$ с точки зрения Алисы и $g_v(y)$ с точки зрения Боба. Мы говорим, что протокол вычисляет f, если для любой пары $(x, y) \in X \times Y$ для листа, в который мы придем, будет выполнено $g_v(x) = h_v(y) = f(x, y)$.

Длина пройденного нами пути будет количеством переданных битов (для данных x, y). Высота дерева является количеством переданных битов в худшем случае и называется коммуникационной сложностью протокола. В этих терминах $D(f)$ есть наименьшая высота (глубина) протокола для вычисления f. Величина $D(f)$ называется *коммуникационной сложностью* f.

Например, пусть $X = Y = \{0,1\}^n$ и $Z = \{0,1\}$. Тогда $D(f) \leqslant n+1$ для любой функции $f : X \times Y \to Z$. Действительно, Алиса передает Бобу свой вход, затем Боб передает Алисе $f(x, y)$. Этот протокол задается следующим сбалансированным деревом высоты $n+1$. Все вершины первых n уровней помечены буквой А и $g_v(x) = x_i$ (i-ый бит x) для любой вершины v уровня i (уровни нумеруются начиная с 1). Вершины уровня $n+1$ соответствуют всевозможным бинарным словам x длины n, причем вычисление на входе x, y как раз и приходит в вершину, соответствующую x. Каждая вершина x уровня $n+1$ помечена буквой Б и функцией $g_x(y) = f(x, y)$. Листы находятся на уровне $n+2$ и помечены константными функциями, равными последнему переданному биту.

9.1 Среднее арифметическое и медиана мультимножества

Пусть $X = Y$ есть множество непустых подмножеств множества $\{1, \ldots, n\}$. Функция MED_n выдает средний элемент в упорядоченном мультимножестве $x \cup y$. (Средний элемент в мультимножестве определяется следующим образом. Пусть $x = \{x_1, \ldots, x_k\}$, $y = \{y_1, \ldots, y_m\}$. Отсортируем массив $x_1, \ldots, x_k, y_1, \ldots, y_m$ и обозначим через $z_1, \ldots, z_{k+m}$ полученный массив. Тогда $\mathrm{MED}_n(x, y) = z_{\lfloor (k+m)/2 \rfloor}$. Функция AVE_n выдает среднее арифметическое всех элементов из мультимножества $x \cup y$.

Задача 22. Докажите, что $\log n \leqslant D(\mathrm{AVE}_n) \leqslant c \log n$ для некоторой константы c.

Решение. Первое неравенство доказывается с помощью информационного подхода. Пусть у Боба одноэлементное множество $\{n\}$. Очевидно, что при варьировании множества Алисы функция может принимать все значения от 1 до n. Поэтому Алиса обязана послать Бобу не менее $\log n$ битов.

Можно рассуждать и по-другому: среднее арифметическое может оказаться равным любому из чисел от 1 до n, поэтому в протоколе должно быть не менее n листьев, а его высота не может быть меньше $\log n$.

Второе неравенство: Алиса и Боб передают друг другу общее количество и сумму своих элементов.

Задача 23. Докажите, что $\log n \leqslant D(\mathrm{MED}_n) \leqslant c \log^2 n$ для некоторой константы c.

На самом деле верхняя оценка может быть улучшена до $D(f) \leqslant c \log n$, но это сложнее доказать.

Решение. Первое неравенство доказывается точно так же, как и в предыдущей задаче.

Докажем второе неравенство. Алиса и Боб проводят бинарный поиск медианы. Для этого они сначала выясняют, на каком из двух отрезков $[1, n/2)$ или $[n/2, n]$ находится медиана. Для этого им достаточно знать общее число элементов в мультимножестве $x \cup y$ и сколько из них меньше $n/2$. Для этого каждый из них посылает другому общее количество элементов в своем множестве, а также количество тех из них, которые меньше $n/2$. После этого они опять делят пополам отрезок, на котором находится медиана и так далее. В каждом раунде они посылают $2 \log n$ битов (а в первом вдвое больше). Через $\log n$ раундов они узнают медиану точно. Всего будет передано $O(\log^2 n)$ битов.

9.2 Предикат равенства

Пример 1. Пусть $X = Y = \{0,1\}^n$, а $f(x,y) = 1$, если $x = y$, и $f(x,y) = 0$ иначе. Эта функция называется предикатом равенства на n-битовых строках и обозначается через EQ_n. Для вычисления $f(x,y)$ достаточно передать $n+1$ битов — Алиса передает свою строку Бобу, а затем Боб передает Алисе $f(x,y)$.

Задача 24. Доказать, что в для вычисления предиката равенства необходимо передать $n+1$ битов. То есть для любого алгоритма вычисления

EQ_n найдутся входы x, y Алисы и Боба, на которых алгоритм передаст не менее $n+1$ битов.

Решение. Нам нужно доказать, что высота любого протокола вычисления EQ_n не меньше $n+1$. Попробуем применить для этого уже известный подход. Представим себе постороннего наблюдателя (Виктора), который знает протокол, но не знает пары x, y. Будем выбирать путь в дереве рекурсивно, следя за тем, чтобы Виктор получил как можно меньше информации о паре (x, y). Это означает следующее: пусть скажем корень дерева помечен буквой А. Множество X всех возможных x делится в корне на два множества: X_0 и X_1, в зависимости от первого переданного бита. Пойдем налево, если X_0 больше X_1, и направо, иначе. То есть мы фиксируем первый посланный Алисой бит так, чтобы Виктор получил поменьше информации о неизвестном ему исходе. Теперь множество возможных исходов есть наибольшее из множеств $X_0 \times Y$, $X_1 \times Y$. Точно также поступим в следующей вершине.

На каждом шаге мы находимся в некоторой вершине v протокола. Множество возможных исходов, то есть множество тех пар (x, y), для которых вычисление придет в вершину v, имеет вид $A \times B$ (такие множества принято называть прямоугольниками). По построению его мощность не меньше $2^{2n-l(v)}$, где $l(v)$ — длина слова v, то есть высота текущей вершины. Если дерево имеет высоту не больше n, то в конечном итоге мы окажемся в листе, для которого множество возможных исходов $A \times B$ имеет не менее 2^n элементов. При этом для любой пары (x, y) из $A \times B$ и Алиса, и Боб знают, равны ли x и y. К сожалению, в этом еще нет противоречия, поскольку такие множества бывают. Например, в качестве A можно взять множество всех x, у которых 1-ый бит равен 0, а в качестве B — множество всех y, у которых 1-ый бит равен 1. Размер прямоугольника $A \times B$ есть $2^{2n-2} \geqslant 2^n$ (если $n \geqslant 2$). И при этом $x \neq y$ для любой пары (x, y) из $A \times B$.

Однако в этом неудавшемся рассуждении на самом деле много полезного. А именно, по существу мы доказали следующую лемму.

Лемма 9.1. *Для любого коммуникационного протокола и любой вершины v в этом протоколе множество R_v, состоящее из тех пар (x, y), для которых вычисление приходит в вершину v, является прямоугольником.*

Доказательство. Это легко доказывается индукцией по высоте вершины. Прямоугольники для сыновей вершины v получаются из прямо-

угольника $A \times B$ для вершины v следующим образом: если v помечена буквой А, то прямоугольники, соответствующие левому и правому сыну v, есть $A_0 \times B$ и $A_1 \times B$, где $A_i = \{x \in A \mid g_v(x) = i\}$. Аналогично, если v помечена буквой Б, то прямоугольники, соответствующие левому и правому сыну v, есть $A \times B_0$ и $A \times B_1$. □

Кроме того, нетрудно понять следующее.

Лемма 9.2. *Для любого коммуникационного протокола вычисления f и любого листа v в этом протоколе прямоугольник R_v является одноцветным для f (так называются множества, на которых функция f постоянна, а общее значение f на парах из этого множества, называется цветом множества).*

Доказательство. Пусть $R_v = A \times B$. Тогда $g_v(x) = f(x, y) = h_v(y)$, для всех $(x, y) \in A \times B$, где g_v, h_v — функции, которыми помечен лист v. В частности, $g_v(x) = h_v(y)$ для всех $(x, y) \in A \times B$, что означает, что g_v постоянна на A, а h_v постоянна на B, значит, f постоянна на $A \times B$. (В этом рассуждении нам важно, что R_v есть прямоугольник. Иначе утверждение было бы неверным, например, для $R_v = \{(0, 0), (1, 1)\}$ можно положить $g_v(x) = x$, $h_v(y) = y$ и $f(x, y) = x = y$ для всех $(x, y) \in R_v$.) □

Теперь мы можем закончить решение задачи. Множество $X \times Y$ разбивается на одноцветные прямоугольники, соответствующие листам протокола. Если глубина протокола меньше $n + 1$, то листьев в нем не больше 2^n. Ни один одноцветный прямоугольник не может содержать две разных диагональных пары (так называются пары вида (x, x)). Действительно если прямоугольник содержит пары (x, x) и (y, y), где $x \neq y$, то он содержит и пару (x, y), а значит не является одноцветным. Диагональных пар ровно 2^n штук, поэтому для их покрытия необходимо по крайней мере 2^n прямоугольников цвета 1. Кроме того, нужен хотя бы один прямоугольник цвета 0. Поэтому 2^n одноцветных прямоугольников не хватит даже чтобы покрыть $X \times Y$ с пересечениями.

9.3 Разбиения на одноцветные прямоугольники

Рассуждения из предыдущего параграфа можно обобщить следующим образом. Матрицей функции $f : X \times Y \to Z$ называется матрица, стро-

ки и столбцы которой нумеруются элементами X и Y соответственно, а элементы равны $f(x, y)$.

Теорема 9.1. *Пусть f — произвольная функция из $X \times Y$ в Z. Обозначим через $C^R(f)$ наименьшее количество одноцветных для f прямоугольников, на которые можно разбить матрицу функции f. Тогда $D(f) \geqslant \log C^R(f)$.*

Доказательство. Действительно, в протоколе наименьшей высоты для вычисления f не больше $2^{D(f)}$ листьев и каждому листу соответствует одноцветный прямоугольник (см. леммы 3.1 и 3.2). Эти прямоугольники образуют разбиение матрицы функции. □

9.4 Метод трудных множеств и метод размера прямоугольников

Будем множество $H \subset X \times Y$ называть *трудным для f*, если не существует одноцветного для f прямоугольника, содержащего две различных пары из H. Очевидно, $C^R(f)$ не меньше размера любого трудного множества. Поэтому можно оценивать снизу $C^R(f)$, а значит и коммуникационную сложность, указывая большие трудные множества. В сущности мы это и сделали для оценки $C^R(\mathrm{EQ}_n)$, предъявив трудное множество, состоящее из всех диагональных пар и еще одной недиагональной пары.

Задача 25. Функция GT_n определена на парах натуральных чисел от 1 до 2^n и принимает значение 1, если $x > y$. Докажите, что $D(\mathrm{GT}_n) = n + 1$.

Задача 26. Предикат DISJ_n определен на парах подмножеств $x, y \subseteq \{1, \ldots, n\}$ и принимает значение 1, если x, y не пересекаются. Доказать, что коммуникационная сложность этой функции не меньше $n + 1$.

Решение. Рассмотрим множество H состоящее из всех пар вида $(x, \overline{x})$. Любая такая пара состоит из непересекающихся множеств, поэтому может покрываться только прямоугольником цвета 1. При этом никакой прямоугольник цвета 1 не может содержать двух таких пар, поскольку для любых разных x, y, либо x пересекает $\overline{y}$, либо $\overline{x}$ пересекает y (либо и то, и другое). Таких пар ровно 2^n и добавив в H еще любую одну пару пересекающихся множеств, мы получим трудное множество

размера $2^n + 1$ (добавленная пара может покрываться только прямоугольником цвета 0, поэтому она не может входить в один одноцветный прямоугольник вместе ни с одной парой из H).

Обобщением метода трудных множеств является метод размера прямоугольников. Пусть на пространстве $X \times Y$ задано некоторое распределение вероятностей μ. Тогда $C^R(f)$ не меньше $1/\max\mu(A)$, где максимум берется по всем одноцветным прямоугольникам A. Действительно, для любого разбиения матрицы f на одноцветные прямоугольники сумма $\mu(A)$ по всем прямоугольникам из разбиения равна 1. Поскольку сумма не превосходит количества слагаемых, умноженного на максимальное из слагаемых, мы получаем неравенство

$$C^R(f)\max\mu(A) \geqslant 1.$$

Таким образом, подобрав μ, для которого вероятность любого одноцветного прямоугольника мала, мы получим хорошую нижнюю оценку для $C^R(f)$.

Объясним, почему метод трудных множеств является частным случаем метода размера прямоугольников. Пусть дано трудное множество H. Рассмотрим равномерное распределение на элементах H. Тогда вероятность любого одноцветного прямоугольника не больше $1/|H|$, поскольку он содержит не более одной пары из H. Значит, в этом случае $\max\mu(A) = 1/|H|$.

Задача 27. Рассмотрим функцию $\mathrm{SUM}_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^{n+1}$, возвращающую сумму чисел, записанных в двоичной системе счисления. Докажите, $D(\mathrm{SUM}_n) = 2n$. [Указание. Множество, состоящее из всех входных пар, является трудным.]

Задача 28. Функция IP_n определена на парах двоичных слов длины n и принимает значение 1, если $\sum x_i y_i \equiv 1 \bmod 2$. Докажите, что любой коммуникационный протокол, вычисляющий функцию IP_n, имеет глубину не менее n. [Указание. Докажите, что любой одноцветный прямоугольник содержит не более 2^n пар (используйте соображения размерности линейных пространств).]

9.5 Метод ранга матрицы

Пусть множество Z состоит из действительных чисел. Тогда $C^R(f)$ не превосходит ранга матрицы функции f. Действительно, фиксируем

произвольное разбиение полученной матрицы на одноцветные прямоугольники. Для каждого прямоугольника разбиения рассмотрим матрицу, у которой в этом прямоугольнике значение функции равно 1, а вне его всюду нули. Такая матрица имеет ранг не больше 1 (поскольку все ненулевые ее столбцы одинаковы). А вся матрица функции f равна сумме таких матриц. Поскольку ранг суммы не превосходит суммы рангов, ранг матрицы f не превосходит количества прямоугольников разбиения.

Задача 29. Докажите, что ранг матрицы отрицания предиката DISJ_n равен 2^n.

Задача 30. Докажите, что ранг матрицы предиката IP_n не меньше $\varepsilon 2^n$ для некоторого положительного ε.

9.6 Вероятностные протоколы

В вероятностном протоколе у Алисы и Боба есть собственные независимые источники случайности. Биты, пересылаемые Алисой, могут зависеть не только от ее входа и текущей вершины дерева, но и от ее случайного входа. Аналогично для Боба. Формально, функции g_v, которыми помечены вершины Алисы, в качестве аргументов получают пары из $X \times R$, а функции h_v, которыми помечены вершины Боба, — пары из $Y \times S$ (и то же самое для функций, которыми помечены листья). Перед началом работы Алисе выдается пара $(x, r) \in X \times R$, а Бобу — пара $(y, s) \in Y \times S$. Множества R, S могут быть произвольными (они входят в спецификацию протокола). Результат работы протокола теперь зависит от четверки x, y, r, s.

Мы говорим, что вероятностный протокол ε-вычисляет f, если для любой пары x, y с вероятностью не менее $1 - \varepsilon$ (при случайном выборе пары $(r, s) \in R \times S$) результат работы протокола равен $f(x, y)$ и с точки зрения Алисы, и с точки зрения Боба. Через $R^\varepsilon(f)$ обозначается минимальная высота вероятностного протокола, ε-вычисляющего f.

Оказывается, для некоторых предикатов $R^\varepsilon(f)$ может быть значительно меньше $D(f)$ даже для очень малых ε. Другими словами, использование случайности может значительно сократить количество передаваемых битов.

Задача 31. Докажите, что $R^\varepsilon(\mathrm{EQ}_n) = O(\log n + \log(1/\varepsilon))$. [Указание. Алиса передает Бобу значение некоторой хэш-функции на x, а Боб

сравнивает полученное значение со значением этой же хэш-функции на своем входе и сообщает результат проверки Алисе. Оба считают, что $x = y$, если хэш-значения совпали. Хэш-функция выбирается Алисой случайно из некоторого семейства функций, зависящего от n и ε и содержащего не более $(n/\varepsilon)^c$ функций (поскольку Алиса должна передать Бобу номер выбранной функции), которые принимают значения в некотором множестве мощности $(n/\varepsilon)^c$.

В качестве семейства хэш-функции годятся следующие два семейства: $x \mapsto x \bmod p$, где p — простое число подходящей величины, задающее конкретную функцию из семейства, и $x \mapsto (x_0 + x_1 a + \ldots + x_{n-1}a^{n-1}) \bmod p$, где p — простое число подходящей величины и зависит только от n, а a — натуральное число меньшее p, задающее конкретную функцию из семейства.]

Задача 32. Докажите, что $R^\varepsilon(\mathrm{GE}_n) = O(\log n(\log n + \log(1/\varepsilon)))$. [Указание. Алиса и Боб используют бинарный поиск первого слева бита, в котором различаются их входы. Для этого они применяют вероятностный протокол из предыдущей задачи для сравнения своих первых половин: если они одинаковы, то первое различие левее середины, а иначе правее, и так далее. Вероятность ошибки на каждом шаге не должна превышать $\varepsilon/\log n$, тогда суммарная вероятность ошибки не превысит ε.]

Задача 33. Докажите, что если Алиса и Боб имеют доступ к общему источнику случайности (то есть им выдается одинаковая случайная строка), то они могут ε-вычислить предикат EQ_n, передав $O(\log(1/\varepsilon))$ бит. [Указание. В этом случае они могут использовать хэш-функции из значительно большего семейства функций, прочитав номер хэш-функции из общего источника случайности. Хэш-значение должно иметь не более $O(\log(1/\varepsilon))$ бит. Например, годится семейство $x \mapsto Ax$, где A — 0-1-матрица размера $k \times n$. Умножение матрицы на столбец x проводятся в поле вычетов по модулю 2. В качестве k можно взять целую часть от $\log(1/\varepsilon)$.]

Задача 34. Докажите, что если Алиса и Боб имеют доступ к общему источнику случайности (то есть им выдается одинаковая случайная строка), то для любого фиксированного положительного ε они могут ε-вычислить предикат GE_n с ошибкой не более ε, передав $O(\log n)$ бит. [Указание. Используйте задачу 13 или 12.]

Можно доказать, что любой протокол ε-вычисления функции, использующий общий источник случайности, можно переделать в вероятностный протокол её 2ε-вычисления, который использует только частные случайные биты, ценой увеличения количества переданных бит на $O(\log\log|X\times Y|+\log(1/\varepsilon))$. (Правда, алгоритм преобразования требует полиномиального от $|X\times Y|$ времени.) Поэтому из задачи 34 следует, что $R^{\varepsilon}(\mathrm{GE}_n)=O(\log n)$ для любого фиксированного положительного ε.

Можно доказать, что для предикатов DISJ_n и IP_n случайность мало помогает: для них R^{ε} имеет порядок [4] $\Omega(n)$ для всех достаточно малых ε. Об этом можно прочитать в превосходном учебнике по коммуникационной сложности [8].

[4] То есть $R^{\varepsilon}>\delta n$ для некоторого положительного δ.

Глава 10. Энтропия Шеннона

10.1 Определение

Пусть фиксированы неотрицательные числа $p_1, \ldots, p_k$, в сумме равные единице. Их энтропия Шеннона определяется как

$$H = p_1(-\log p_1) + p_2(-\log p_2) + \ldots + p_k(-\log p_k)$$

(при $p = 0$ мы полагаем $p \log p = 0$, доопределяя тем самым функцию $p \mapsto p \log p$ по непрерывности).

Мотивировка этой формулы такова: пусть имеется алфавит из k букв $a_1, \ldots, a_k$, причем буква a_i появляется с частотой p_i, а каждое её появление несёт $-\log p_i$ битов информации; в среднем получается H битов на букву. Нужно только объяснить, почему мы считаем, что появление буквы с вероятностью p несёт $-\log p$ битов информации. Это можно сделать так. Пусть частоты всех букв совпадают (и равны $1/k$). Тогда $H = \log k$, то есть в этом частном случае энтропия совпадает с информацией по Хартли в исходах из множества $\{a_1, \ldots, a_k\}$. Таким образом, энтропия Шеннона обобщает количество информации по Хартли — последнее есть шенноновская энтропия для равновероятных исходов.

Задача 35. Докажите, что энтропия Шеннона не меньше *минимальной энтропии*, определяемой как $H_{\min} = \min_i(-\log p_i)$.

Энтропия Шеннона возникает в самых разных контекстах. Первым таким примером является задача бинарного кодирования букв данного алфавита.

10.2 Коды

Пусть нам нужно закодировать буквы некоторого алфавита A, состоящего из k букв $a_1, \ldots, a_k$, двоичными словами (наподобие азбуки Мор-

зе, только вместо точки и тире используются нуль и единица). Пусть буква a_i кодируется некоторым словом c_i. Естественно требовать, чтобы все слова c_i были различны. Но этого мало, если мы хотим записывать коды подряд. Скажем, если алфавит состоит из букв А, Б и В, имеющих коды 0, 1 и 01, то мы не сможем отличить слово АБАБ от слова АБВ: в обоих случаях будет последовательность 0101. (В азбуке Морзе, кстати, такой проблемы нет, поскольку между отдельными буквами делается перерыв — больший, чем между точками и тире внутри буквы). Поэтому надо отдельно позаботиться об однозначности декодирования.

Другой предмет заботы при построении кода — его экономность. Полезно выбрать кодовые слова c_i по возможности более короткими (насколько это возможно при сохранении однозначности декодирования). Более того, если не удаётся сделать короткими все кодовые слова, разумно в первую очередь позаботиться о наиболее часто встречающихся буквах. (Это обстоятельство учитывалось и при составлении азбуки Морзе.)

Перейдём к формальным определениям. *Кодом* для алфавита A, состоящего из k букв $a_1, \ldots, a_k$, называется набор из k двоичных слов $c_1, \ldots, c_k$. Они называются *кодовыми словами* данного кода; слово c_i называется *кодом* буквы a_i; всякое слово в алфавите A кодируется двоичным словом, получаемым соединением кодов соответствующих букв.

Будем называть код *инъективным*, если коды различных букв различны, и *однозначно декодируемым*, если коды любых двух различных слов различны. Код называется *префиксным*, если ни одно из кодовых слов (соответствующих буквам алфавита A) не является началом (префиксом) другого. (Это название стало традиционным, хотя более логичное — *беспрефиксный* код — также используется.)

Теорема 10.1. *Всякий префиксный код является однозначно декодируемым.*

Доказательство. Первое кодовое слово (код первой буквы) отщепляется однозначно (в силу префиксности), затем отщепляется код второй буквы и т.п. □

Задача 36. Покажите, что не всякий однозначно декодируемый код является префиксным. [Указание. Он может быть, например, «суффиксным».]

Задача 37. Укажите явно взаимно-однозначное соответствие между множеством бесконечных последовательностей цифр $0, 1, 2$ и множеством бесконечных последовательностей нулей и единиц. [Указание. Используйте префиксный код $0 \mapsto 00$, $1 \mapsto 01$, $2 \mapsto 1$.]

Задача 38. Пусть слова $c_1, \ldots, c_k$ и $d_1, \ldots, d_k$ образуют префиксный код (по отдельности). Покажите, что kl слов $c_i d_j$ (приписываем одно слово к другому без разделителя) также образуют префиксный код.

Чтобы сравнивать коды по их экономности, нужно фиксировать частоты букв. Пусть даны неотрицательные числа $p_1, \ldots, p_k$, в сумме равные единице; число p_i будем называть *частотой* (или *вероятностью*) буквы a_i. Для каждого кода $c_1, \ldots, c_k$ (для букв $a_1, \ldots, a_k$) определим *среднюю длину* кода как сумму

$$\sum_i p_i l(c_i).$$

Возникает задача: для данных $p_1, \ldots, p_k$ найти код по возможности меньшей средней длины (в том или ином классе кодов). Что можно сказать о минимально возможной длине префиксного кода для данных частот $p_1, \ldots, p_k$? Оказывается, она близка к энтропии Шеннона (отличается от нее менее чем на 1).

Теорема 10.2. (**а**) *Для любого префиксного кода с кодовыми словами $c_1, \ldots, c_k$ выполняется неравенство*

$$\sum_i p_i l(c_i) \geqslant H$$

(средняя длина кода не меньше энтропии).

(**б**) *Существует префиксный код, для которого*

$$\sum_i p_i l(c_i) < H + 1$$

Доказательство. Заметим, что в этой теореме реально фигурируют не сами кодовые слова, а их длины. Поэтому важно знать, какие наборы чисел могут быть длинами кодовых слов префиксного кода. Ответ даёт такая лемма:

Лемма (неравенство Крафта). *Пусть фиксированы целые неотрицательные числа $n_1, \ldots, n_k$ и требуется найти двоичные слова*

$c_1, \ldots, c_k$ указанных длин ($l(c_i) = n_i$), причём так, чтобы ни одно из этих слов не было началом другого. Это возможно тогда и только тогда, когда $\sum_i 2^{-n_i} \leqslant 1$.

Доказательство. Сопоставим каждому двоичному слову v отрезок I_v на действительной прямой по следующему рекурсивному правилу: $I_\Lambda = [0, 1]$ (пустому слову сопоставляется отрезок $[0, 1]$), и для любого слова v словам $v0$ и $v1$ сопоставляется левая и правая половина отрезка I_v. Например, $I_{01} = [1/4, 1/2]$. Можно и явно задать это соответствие формулой:

$$I_v = [0.v, 0.v + 2^{-l(v)}].$$

По построению это соответствие имеет следующие свойства:

- v есть начало u тогда и только тогда, когда $I_u \subset I_v$,
- Если v и u не согласованы (ни одно из них не является началом другого), то I_u и I_v не перекрываются (общая граничная точка перекрытием не считается),
- Длина I_v равна $2^{-l(v)}$.

Теперь легко убедиться в первом утверждении: если слова $c_1, \ldots, c_k$ попарно не согласованы, то $\sum_i 2^{-l(c_i)} \leqslant 1$. Действительно, $2^{-l(c_i)}$ есть длина отрезка I_{c_i}. Поскольку отрезки $I_{c_1}, \ldots, I_{c_k}$ не перекрываются, сумма их длин не превосходит длины всего отрезка $[0, 1]$.

Обратное утверждение: без ограничения общности можно считать, что $n_1 \geqslant \ldots \geqslant n_k$. Отложим на луче $[0, \infty)$, начиная с начала без пропусков не перекрывающиеся отрезки длин $2^{-n_1}, \ldots, 2^{-n_k}$. По условию сумма этих чисел не больше 1, поэтому все отложенные отрезки являются под-отрезками отрезка $[0, 1]$. Докажем, что каждый из отложенных отрезков имеет вид I_c для некоторого двоичного слова c (этого достаточно, поскольку из того, что отрезки не перекрываются, следует, что соответствующие слова $c_1, \ldots, c_k$ попарно не согласованы; кроме того они имеют нужные длины по построению). Фиксируем любое $i \leqslant k$. В силу неравенства $n_1 \geqslant \ldots \geqslant n_k$, суммарная длина первых $i - 1$ отложенных отрезков имеет вид $k/2^{n_i}$ для некоторого натурального k. То есть i-ый отрезок имеет вид $[k/2^{n_i}, (k + 1)/2^{n_i}]$, причем $k < 2^{n_i}$, так как в противном случае он не был бы подотрезком отрезка $[0, 1]$. Поэтому i-ый отрезок имеет вид I_c для некоторого двоичного слова c. □

Вернёмся к доказательству теоремы. Можно считать, что все p_i положительны, поскольку нулевые p_i не дают вклада ни в среднюю длину

кода, ни в энтропию. В пункте (а) нам надо доказать, что если n_i — неотрицательные целые числа и $\sum_i 2^{-n_i} \leqslant 1$, то сумма $\sum p_i n_i$ не меньше шенноновской энтропии H. Это удобнее доказывать сразу для произвольных n_i (не обязательно целых) и перейдя к другим координатам. Обозначим через q_i величину 2^{-n_i}. В этих координатах утверждение таково: если $q_i > 0$ и $\sum q_i \leqslant 1$, то

$$\sum p_i(-\log q_i) \geqslant \sum p_i(-\log p_i)$$

Это неравенство иногда называют *неравенством Гиббса*. Чтобы доказать его, заметим, что разница между правой и левой частью равна

$$\sum_i p_i \log \frac{q_i}{p_i}$$

и в силу выпуклости логарифма (взвешенная сумма логарифмов не превосходит логарифма взвешенной суммы: $\sum p_i \log u_i \leqslant \log(\sum_i p_i u_i)$; это верно для любых положительных u_i) не превосходит

$$\log\left(\sum_i p_i \frac{q_i}{p_i}\right) = \log\left(\sum q_i\right) \leqslant \log 1 = 0.$$

Утверждение (а) доказано.

Отметим кстати, что неотрицательную величину

$$\sum_i p_i \log \frac{p_i}{q_i}$$

называют *расстоянием Кульбака–Лейблера* (Kullback–Leibler distance) между распределениями вероятностей p_i и q_i (при этом предполагается, что $\sum q_i = 1$), хотя это «расстояние» и не симметрично. Выпуклость логарифма (отрицательность второй производной) гарантирует, что это расстояние неотрицательно и обращается в нуль, лишь если $p_i = q_i$ при всех i. Величину

$$\sum_i p_i \log \frac{1}{q_i}$$

называют *перекрёстной энтропией* между распределениями вероятностей p_i и q_i. Перекрёстная энтропия не меньше энтропии распределения p_i, причем разница между ними равна расстоянию Кульбака-Лейблера.

Чтобы доказать утверждение (б), рассмотрим числа $n_i = \lceil -\log_2 p_i \rceil$ (где $\lceil u \rceil$ обозначает наименьшее целое число, большее или равное u). Тогда

$$\frac{p_i}{2} < 2^{-n_i} \leqslant p_i.$$

Неравенство $2^{-n_i} \leqslant p_i$ гарантирует, что выполнены условия леммы (и потому можно найти кодовые слова соответствующих длин). Неравенство $p_i/2 < 2^{-n_i}$ означает, что n_i превосходит $(-\log p_i)$ менее чем на 1, что сохраняется и после усреднения: средняя длина кода $(\sum p_i n_i)$ превосходит $H = \sum p_i(-\log p_i)$ менее чем на 1. □

Кратко доказательство теоремы можно резюмировать так: если забыть, что длины кодовых слов должны быть целыми, и разрешать любые числа n_i, только бы сумма 2^{-n_i} не превышала единицы, то выгоднее всего взять $n_i = -\log p_i$ (следует из выпуклости логарифма). Требование же целочисленности приводит к увеличению n_i, но не более чем на единицу.

Замечание 10.1. Кодирование, примененное в доказательстве пункта (б), обладает следующим свойством: $l(c_i) < -\log p_i + 1$. Будем кодирования с таким свойством (с произвольной константой вместо 1) называть *сбалансированными* (поскольку они сохраняют баланс между $l(c_i)$ и $-\log p_i$).

Теорема 10.3. *Энтропия распределения $p_1, \ldots, p_n$ с n значениями не превосходит $\log n$ и равна $\log n$ в единственном случае, когда все p_i равны.*

Доказательство. Если n есть степень двойки, то неравенство $H \leqslant \log n$ прямо следует из теоремы 10.2, поскольку можно рассмотреть префиксный код, в котором n кодовых слов имеют длину $\log n$. В общем случае надо применить неравенство Гиббса с $q_i = 1/n$ при всех i и вспомнить, что это неравенство обращается в равенство при $p_i = q_i$. □

Задача 39. Пусть целое число x выбирается случайным образом в интервале от 1 до 1000 (все возможные значения x равновероятны). Докажите, что любой алгоритм, который с помощью вопросов с ответами ДА/НЕТ находит x, задает в среднем не меньше $\log 1000$ вопросов. [Указание. Любой такой алгоритм задает префиксное кодирование чисел от 1 до 1000.]

Задача 40. Докажите, что любое инъективное кодирование можно преобразовать в префиксное ценой небольшого увеличения средней длины кода: если у исходного кода средняя длина была l, то у нового она будет не больше $l + 2\log l + 2$. [Указание. Используйте композицию исходного кода и префиксного кодирования $x \mapsto \widehat{x}$ со стр. 187, а также вогнутость логарифмической функции.]

Задача 41. Пусть $\{a_1, \ldots, a_n\}$ — произвольный алфавит и $p_1, \ldots, p_n$ — вероятности букв этого алфавита. Докажите, что для любого инъективного кодирования букв этого алфавита средняя длина кода не меньше $H - 2\log H - 2$. [Указание. Используйте предыдущую задачу и теорему 10.2.]

10.2.1 Некоторые известные коды

Арифметический код. Пусть все вероятности букв $p_1, \ldots, p_n$ положительны. Отложим на отрезке $[0, 1]$, начиная с начала, без пропусков, неперекрывающиеся отрезки длин $p_1, \ldots, p_n$. Для каждого i рассмотрим отрезок вида I_c наибольшей длины, целиком включенный в i-ый из полученных отрезков (отрезки I_v определены в доказательстве неравенства Крафта на стр. 113). Одно из таких c (их может быть несколько) и возьмем в качестве кодового слова c_i для i-ой буквы.

Задача 42. Докажите, что $c_1, \ldots, c_n$ есть беспрефиксный код, причем $l(c_i) < -\log p_i + 2$ (то есть арифметическое кодирование сбалансировано).

Задача 43. Докажите, что константу 2 в неравенстве $l(c_i) < -\log p_i + 2$ для длин слов арифметического кода нельзя понизить, даже в предположении, что $p_1, \ldots, p_n$ упорядочены по величине.

Код Шеннона–Фано. Пусть вероятности букв упорядочены по убыванию: $p_1 \geqslant \ldots \geqslant p_n$. Уложим на прямой без пропусков неперекрывающиеся отрезки длин $p_1, \ldots, p_n$ и обозначим i-ый из полученных отрезков через S_i, а их объединение — через S. Коды тех букв a_i, для которых отрезок S_i попал на левую половину отрезка S, будут начинаться на 0, а коды тех букв, для отрезок S_i попал на правую половину отрезка S, будут начинаться на 1. Один из отрезков (центральный) может не попасть целиком ни на левую, ни на правую половину. С ним поступим так. Если этот отрезок первый или последний, то начнём его код с, соответственно, 0 или 1. Иначе отнесём его куда угодно. Далее

применяем ту же стратегию отдельно к буквам, код которых начинается на 0, и к буквам, код которых начинается на 1.

Задача 44. (а) Докажите, что код Шеннона–Фано является префиксным.
(б) Докажите, что если центральный отрезок относить туда, куда попала его большая часть, то кодирование Шеннона–Фано не является сбалансированным (то есть не существует константы d, для которой выполнено $l(c_i) < -\log p_i + d$ для любых k и любых исходных вероятностей $p_1, \ldots, p_k$).
(в) Докажите, что если центральный отрезок всегда относить к правой половине, то кодирование Шеннона–Фано также не является сбалансированным.
(г) Докажите, что если центральный отрезок всегда относить к левой половине, то кодирование Шеннона–Фано *является сбалансированным*. [Указание. Зафиксируем номер буквы i. При определении кода этой буквы мы делаем несколько рекурсивных вызовов алгоритма, которые будем нумеровать натуральными числами $0, 1, \ldots$. При вызове номер d алгоритм получает на вход некоторые вероятности (среди которых вероятность и i-ой буквы), сумму которых мы обозначим через l_d. Будем следить за произведением $2^d l_d$. Сначала $d = 0$ и $l_0 = 1$, а при последнем вызове d есть длина кода i-ой буквы, а l_d — ее вероятность. Таким образом, нам надо ограничить величину $2^d l_d$ некоторой константой. Всякий раз, когда наша буква оказывалась на правой половине, величина $2^d l_d$ не увеличивалась, поскольку суммарная вероятность букв из правой половины не больше половины l_d. А при тех рекурсивных вызовах, когда наша буква лежала на левой половине (будем называть их «плохими») величина $2^d l_d$ увеличивается не более, чем в $1 + 2q$ раз, где q есть относительная длина отрезка, по которому прошло деление пополам (то есть отношение длины этого отрезка к l_d). Поэтому нам достаточно доказать, что произведение величин $1 + 2q$ по всем плохим вызовам ограничено. Для этого достаточно установить, что при каждом плохом рекурсивном вызове значение q не менее чем в полтора раза превышает значение q при предыдущем плохом рекурсивном вызове. Пусть при очередном плохом вызове выполнено неравенство $q \leqslant 1/6$. Тогда левая половина составляет не более $2/3$ всего отрезка, и длины всех отрезков на ней не меньше отрезка, по которому прошло деление. Поэтому при любом следующем делении на левой половине,

относительная длина отрезка, по которому прошло деление, не меньше $3q/2$. Поэтому общее возрастание $2^d l_d$ при всех плохих рекурсивных вызовах, для которых выполнено неравенство $q \leqslant 1/6$, не превосходит $(1 + 2 \cdot (1/6))(1 + 2 \cdot (1/6)(2/3))(1 + 2 \cdot (1/6)(2/3)^2) \cdot \ldots = O(1)$. Наконец, после того, как q станет больше $1/6$, может произойти не более 3 вызовов.]

Замечание 10.2. Чуть позже мы увидим, что куда бы мы не относили центральный отрезок, средняя длина кода Шеннона-Фано не превосходит $H + O(1)$.

Код Хаффмана. Мы доказали, что средняя длина оптимального префиксного кода (для данных $p_1, \ldots, p_k$) заключена между H и $H + 1$. Попробуем разобраться, как найти этот код.

Пусть $n_1, \ldots, n_k$ — длины кодовых слов оптимального кода для данных $p_1, \ldots, p_k$. Будем предполагать (переставив буквы), что

$$p_1 \leqslant p_2 \leqslant \ldots \leqslant p_k.$$

В этом случае

$$n_1 \geqslant n_2 \geqslant \ldots \geqslant n_k$$

(если бы более частая буква кодировалась длиннее, чем более редкая, то обмен кодов уменьшил бы среднюю длину).

Заметим, что для оптимального кода $n_1 = n_2$ (две наиболее редкие буквы всегда имеют одну и ту же длину кода). В самом деле, если $n_1 > n_2$, то n_1 больше всех остальных n_i. Поэтому в сумме $\sum_i 2^{-n_i}$ первое слагаемое меньше всех других, неравенство $\sum_i 2^{-n_i} \leqslant 1$ не может обратиться в равенство по соображениям чётности, и левая часть его меньше правой по крайней мере на 2^{-n_1}. А значит, n_1 можно уменьшить на единицу, не нарушив неравенства $\sum_i 2^{-n_i} \leqslant 1$, и код не является оптимальным.

Поэтому при выборе оптимального кода достаточно ограничиться кодами с $n_1 = n_2$, и оптимальный среди них соответствует минимуму выражения

$$p_1 n_1 + p_2 n_2 + p_3 n_3 + \ldots + p_k n_k = (p_1 + p_2)n + p_3 n_3 + \ldots + p_k n_k$$

(если через n обозначить общее значение n_1 и n_2) по всем $n, n_3, \ldots, n_k$, для которых

$$2^{-n} + 2^{-n} + 2^{-n_3} + \ldots + 2^{-n_k} \leqslant 1.$$

Перепишем это неравенство как

$$2^{-(n-1)} + 2^{-n_3} + \ldots + 2^{-n_k} \leqslant 1,$$

а выражение, подлежащее минимизации, как

$$(p_1 + p_2) + (p_1 + p_2)(n-1) + p_3 n_3 + \ldots + p_k n_k.$$

Член (p_1+p_2) постоянен и не влияет на поиск минимума, так что задача сводится к поиску оптимального префиксного кода для $k-1$ букв с вероятностями $p_1 + p_2, p_3, \ldots, p_k$.

Получаем рекурсивный алгоритм: соединить две наиболее редкие буквы в одну (сложив вероятности), найти оптимальный префиксный код для этого случая (рекурсивный вызов), а потом вместо одного кодового слова x для соединённой буквы взять два кодовых слова на единицу длиннее ($x0$ и $x1$); ясно, что префиксность кода при этом не нарушится.

Построенный с помощью такого алгоритма оптимальный префиксный код называется *кодом Хаффмана* для данного набора вероятностей p_i.

Задача 45. Докажите, что кодирование Хаффмана не является сбалансированным.

Все три кода (арифметический, Шеннона–Фано и Хаффмана) можно использовать и при неизвестных заранее вероятностях букв алфавита. В этом случае в вероятности букв принимаются равными их частотам, а в начало кода записывается кодовая таблица (кодовые слова всех букв). Например, для исходного слова `abdecaefbcffcbcaddbaacbccedcccbb` (в шестибуквенном алфавите), содержащего 5 букв `a`, 7 букв `b`, 10 букв `c`, 4 буквы `d`, 3 буквы `e` и 3 буквы `f`, кодовые таблицы будут выглядеть следующим образом:

Код	Кодовая таблица
арифметический	000, 010, 100, 1011, 1101, 1111
Шеннона–Фано	100, 01, 00, 101, 110, 111
Хаффмана	010, 10, 00, 011, 110, 111

Длины кодовых слов будут равны, соответственно, 106, 79 и 79. Код Хаффмана, естественно, наиболее короткий (в данном примере его длина совпадает с длиной кода Шеннона–Фано). К кодам, основанным на подсчёте частот букв, мы ещё вернемся в разделе 11.1.

10.3 Коммуникационная сложность в среднем и энтропия Шеннона

Раньше мы интересовались, сколько требуется в худшем случае передать битов, чтобы вычислить данную функцию $f(x, y)$. То есть сложность коммуникационного протокола определялась как его высота (длина максимального пути от корня к листу). Теперь же будем интересоваться количеством битов, передаваемым протоколом в среднем, где усреднение происходит по некоторой вероятностной мере на парах (x, y).

Итак, зафиксируем некоторое распределение μ вероятностей на парах (x, y). Средним количеством переданных битов для данного протокола P называется сумма $\sum_{(x,y)\in X\times Y} \mu(x, y)c_P(x, y)$, где $c_P(x, y)$ обозначает количество переданных протоколом P битов на входах x, y.

Мерой множества пар будем называть сумму всех вероятностей пар из R. Обозначение: $\mu(R)$. Через α_μ будем обозначать максимальную меру одноцветного для f прямоугольника.

Лемма 10.1. *Для любого распределения вероятностей μ среднее количество битов, переданных любым протоколом P вычисления f, не меньше $-\log \alpha_\mu$.*

Доказательство. Рассмотрим случайную величину, исходы которой равны одноцветным прямоугольникам, соответствующим листьям протокола, а вероятность прямоугольника R есть $\mu(R)$. Энтропия Шеннона этой величины не меньше ее минимальной энтропии, которая не меньше $-\log \alpha_\mu$. Протокол P задает беспрефиксный код для своих листьев (код листа есть путь к этому листу). Средняя длина этого кода равна среднему количеству переданных протоколом битов. Осталось воспользоваться неравенством, связывающим среднюю длину беспрефиксного кода и энтропию. □

Задача 46. Пусть f — один из предикатов EQ_n, DISJ_n, GE_n (см. раздел 8.3.1). Построить распределение вероятностей на парах x, y, обладающее следующим свойством. Любой коммуникационный протокол, вычисляющий функцию f, в среднем передает не менее n битов. [Указание. Рассмотрите равномерное распределение на трудном множестве.]

Задача 47. Пусть IP_n — предикат скалярного произведения (см. раздел 8.3.1). Докажите, что при случайном выборе пары x, y (все па-

ры равновероятны) любой коммуникационный протокол, вычисляющий функцию IP_n, в среднем передает не менее n битов. [Указание. Коммуникационный протокол задает префиксный код для некоторого разбиения матрицы предиката IP_n на одноцветные прямоугольники. Вероятность попадания случайной точки в любой прямоугольник в таком разбиении не больше 2^{-n} (задача 28). Поскольку энтропия Шеннона не меньше минимальной энтропии, средняя длина такого кода не меньше n.]

10.4 Неравенство Макмиллана

До сих пор мы изучали в основном префиксные коды. Оказывается, переход к произвольным однозначно декодируемым кодам ничего не даёт с точки зрения сокращения кода:

Теорема 10.4 (неравенство Макмиллана)**.** *Пусть $c_1, \ldots, c_k$ — кодовые слова однозначно декодируемого кода, а $n_i = l(c_i)$ — их длины. Тогда*

$$\sum_i 2^{-n_i} \leqslant 1.$$

Тем самым (лемма Крафта) можно построить и префиксный код с теми же длинами слов.

Доказательство. Будем считать, что в кодовых словах вместо цифр 0 и 1 используются буквы u и v. (Скажем, кодовые слова 0, 01 и 11 мы запишем как u, uv и vv.) Напишем формальную сумму $(c_1 + \ldots + c_k)$ всех кодовых слов, возведём её в N-ую степень (число N мы потом выберем) и раскроем скобки, не переставляя u и v (как если бы они не коммутировали). Например, для $N = 2$ и для приведённого выше примера получится

$$\begin{aligned}(u + uv + vv)(u + uv + vv) = uu + uuv + uvv + uvu + \\ + uvuv + uvvv + vvu + vvuv + vvvv.\end{aligned}$$

Каждое слагаемое в правой части есть соединение некоторых кодовых слов, причём все слагаемые различны (свойство однозначности декодирования). Теперь подставим вместо u и v число $1/2$. В левой части $(c_1 + \ldots + c_k)^N$ превратится при этом в $(2^{-n_1} + \ldots + 2^{-n_k})^N$. Правую

часть оценим сверху: если бы в неё входили все возможные слова данной длины t, то получилось бы 2^t членов, каждый из которых равен 2^{-t}, и сумма равнялась бы 1 (для каждой длины). Поэтому сумма в правой части не превосходит максимальной длины слагаемых, то есть не больше $N \max(n_i)$.

Теперь видно, что если $\sum 2^{-n_i} > 1$, то при больших N левая часть (растущая экспоненциально) становится больше правой (растущей линейно). □

10.5 Энтропия пары случайных величин

При обсуждении энтропии удобно использовать стандартную для теории вероятностей терминологию. Пусть ξ — случайная величина, принимающая конечное число значений $\xi_1, \ldots, \xi_k$ с вероятностями $p_1, \ldots, p_k$. Тогда её *шенноновская энтропия* определяется формулой

$$H(\xi) = p_1(-\log p_1) + \ldots + p_k(-\log p_k)$$

Это определение позволяет говорить об энтропии пары случайных величин ξ и η (определённых на одном и том же вероятностном пространстве), поскольку такая пара сама образует случайную величину. Следующая теорема утверждает, что энтропия пары не превосходит суммы энтропий:

Теорема 10.5.

$$H(\langle \xi, \eta \rangle) \leqslant H(\xi) + H(\eta)$$

Мы предполагаем, что величины ξ и η принимают конечное число значений, поэтому эта теорема представляет собой некоторое неравенство с суммами логарифмов. Именно, пусть ξ принимает k значений $\xi_1, \ldots, \xi_k$, а η принимает l значений $\eta_1, \ldots, \eta_l$. Тогда величина $\langle \xi, \eta \rangle$ может принимать, вообще говоря, kl значений $\langle \xi_i, \eta_j \rangle$ (некоторые из значений могут не встречаться или встречаться с вероятностью нуль). Распределение вероятностей для пары $\langle \xi, \eta \rangle$, таким образом, задаётся таблицей из k строк и l столбцов: число p_{ij}, стоящее в i-ой строке и j-ом столбце, представляет собой вероятность события «$(\xi = \xi_i)$ и $(\eta = \eta_j)$» (здесь $i = 1, \ldots, k$ и $j = 1, \ldots, l$). Все числа p_{ij} неотрицательны и в сумме равны единице (некоторые из них могут равняться нулю).

Сложив числа в строках, мы получим распределение вероятностей для величины ξ: она принимает значение ξ_i с вероятностью $\sum_j p_{ij}$; эту

сумму удобно обозначить p_{i*}; аналогичным образом η принимает значение η_j с вероятностью p_{*j}, которая есть сумма чисел в j-ом столбце.

Таким образом, сформулированная теорема представляет собой неравенство, справедливое для любой прямоугольной таблицы с неотрицательными числами, в сумме равными единице:

$$\sum_{i,j} p_{ij}(-\log p_{ij}) \leqslant \sum_i p_{i*}(-\log p_{i*}) + \sum_j p_{*j}(-\log p_{*j})$$

(где p_{i*} и p_{*j} определяются как суммы по строкам и столбцам).

Это неравенство в конечном счёте сводится к выпуклости логарифма, но полезно понимать его интуитивный смысл. Если отождествить (забыв про разницу порядка единицы) энтропию с длиной кратчайшего префиксного кода, то теорему можно доказать так: пусть имеются короткие префиксные коды для ξ и η (с кодовыми словами $c_1, \ldots, c_k$ и $d_1, \ldots, d_l$). Тогда можно рассмотреть код для пары $\langle \xi, \eta \rangle$, кодируя значения $\langle \xi_i, \eta_j \rangle$ словом $c_i d_j$ (приписываем d_j справа к c_i без разделителя). Это будет, как легко проверить, префиксный код (чтобы отщепить кодовое слово от бесконечной последовательности, надо сначала отщепить c_i, а потом d_j; в обоих случаях это делается однозначно). Средняя длина этого кода будет равна сумме средних длин кодов для ξ и η. Он не обязан быть оптимальным (ведь и неравенство может быть строгим), но даёт оценку сверху для оптимального кода.

Доказательство. Это рассуждение можно превратить в строгое доказательство, если вспомнить, что при доказательстве теоремы 10.2 (с. 113) мы установили, что энтропия равна минимуму величины $\sum_i p_i(-\log_2 q_i)$ по всем наборам неотрицательных чисел q_i с единичной суммой. В частности, энтропия пары (левая часть неравенства) есть минимум сумм

$$\sum_{i,j} p_{ij}(-\log q_{ij})$$

по всем наборам q_{ij} неотрицательных чисел с суммой единица. Будем рассматривать не все наборы, а лишь наборы «ранга 1», которые получаются как произведения

$$q_{ij} = q_{i*} \cdot q_{*j}$$

для некоторых наборов неотрицательных чисел q_{i*} и q_{*j}, каждый из которых имеет сумму 1. Тогда $(-\log q_{ij})$ распадётся в сумму $(-\log q_{i*}) + (-\log q_{*j})$, а вся сумма — в две суммы, которые (после суммирования по одному из индексов) окажутся равными

$$\sum_i p_{i*}(-\log q_{i*})$$

и

$$\sum_j p_{*j}(-\log q_{*j})$$

соответственно. Минимумы этих сумм равны $H(\xi)$ и $H(\eta)$.

Таким образом, левая часть неравенства есть минимум некоторой величины по всем наборам, а правая — по наборам ранга 1, откуда и вытекает требуемое неравенство. □

10.6 Условная энтропия

Условной вероятностью некоторого события B при условии события A называют отношение вероятности события «A и B» к вероятности события A. Это определение имеет смысл, если вероятность события A отлична от нуля. Мотивировка понятна: мы рассматриваем долю исходов, где произошло B, не среди всех исходов, а только среди тех, где произошло A.

Если A — событие, а ξ — случайная величина с конечным числом значений $\xi_1, \ldots, \xi_k$, то можно рассмотреть (помимо вероятностей $\Pr[\xi = \xi_i]$) и условные вероятности $\Pr[(\xi = \xi_i)|A]$. Их сумма тоже равна единице, и получается некоторое новое распределение вероятностей. Его энтропия называется *условной энтропией величины ξ при условии A* и обозначается $H(\xi|A)$, а само это распределение вероятностей можно обозначить $(\xi|A)$.

Задача 48. Покажите, что величина $H(\xi|A)$ может быть и больше, и меньше величины $H(\xi)$. [Указание: распределение $(\xi|A)$ (особенно при малой вероятности события A) мало связано с распределением вероятностей для ξ.]

Неформально говоря, $H(\xi|A)$ — это минимально возможная средняя длина кода, если нас интересуют лишь случаи, когда произошло событие A.

Пусть теперь (как и в прошлом разделе) даны две случайные величины ξ и η. Будем предполагать, что для каждой из них все значения имеют ненулевую вероятность (нулевые можно выбросить). Для каждого значения η_j величины η рассмотрим событие $\eta = \eta_j$ (его вероятность мы обозначали p_{*j}). Рассмотрим условную энтропию величины ξ при условии этого события. Она соответствует распределению вероятностей $i \mapsto p_{ij}/p_{*j}$. Далее усредним эти энтропии с весами, равными вероятностям событий $\eta = \eta_j$. Полученное среднее называют *условной энтропией ξ при известном η* и обозначают $H(\xi|\eta)$. Формально говоря,

$$H(\xi|\eta) = \sum_j \Pr[\eta = \eta_j] H(\xi|\eta = \eta_j),$$

или, в наших обозначениях,

$$H(\xi|\eta) = \sum_j p_{*j} \sum_i \frac{p_{ij}}{p_{*j}} \left(-\log \frac{p_{ij}}{p_{*j}} \right).$$

Основные свойства условной энтропии перечислены в следующей теореме, справедливой для любых случайных величин ξ, η:

Теорема 10.6. (**а**) $H(\xi|\eta) \geqslant 0$;

(**б**) $H(\xi|\eta) = 0$ *тогда и только тогда, когда* $\xi = f(\eta)$ *с вероятностью* 1 *для некоторой функции* f (*оговорка про «вероятность* 1*» означает, что мы пренебрегаем значениями, которые имеют нулевую вероятность*);

(**в**) $H(\xi|\eta) \leqslant H(\xi)$;

(**г**) $H(\langle \xi, \eta \rangle) = H(\eta) + H(\xi|\eta)$.

Доказательство. Первое из этих утверждений очевидно: все $H(\xi|\eta = \eta_j)$ неотрицательны, потому неотрицательна и их взвешенная сумма.

(б) Если взвешенная сумма равна нулю, то все слагаемые с ненулевыми коэффициентами равны нулю, то есть при каждом значении η_j величина $(\xi|\eta = \eta_j)$ имеет нулевую энтропию (и потому принимает лишь одно значение с точностью до событий нулевой вероятности).

Утверждение (в) можно объяснить так: $H(\xi|\eta)$ будет средней длиной оптимального кода для ξ, если разрешить кодировать значения ξ по-разному, в зависимости от значения величины η (в каждом случае свой код, который оптимизируется с учётом условных вероятностей).

Ясно, что это облегчает построение оптимального кода, поэтому средняя длина получается меньше $H(\xi)$.

Более формально: при каждом j величина $H(\xi|\eta=\eta_j)$ равна минимуму суммы

$$\sum_i \frac{p_{ij}}{p_{*j}}(-\log q_{ij})$$

по всем неотрицательным $q_{1j}+q_{2j}+\ldots+q_{kj}=1$ (мы используем свой набор переменных для каждого j) и потому $H(\xi|\eta)$ равна минимуму суммы

$$\sum_j p_{*j}\sum_i \frac{p_{ij}}{p_{*j}}(-\log q_{ij})$$

по всем таблицам, составленным из неотрицательных чисел q_{ij}, у которых сумма каждого столбца равна единице. Если мы теперь ограничимся таблицами, у которых все столбцы одинаковы, $q_{ij}=q_i$, то сумма превратится в

$$\sum_j p_{*j}\sum_i \frac{p_{ij}}{p_{*j}}(-\log q_i)=\sum_j\sum_i p_{ij}(-\log q_i)=\sum_i p_{i*}(-\log q_i)$$

и её минимум станет равным $H(\xi)$. Поэтому $H(\xi|\eta)\leqslant H(\xi)$.

Наконец, пункт (г) представляет собой равенство, которое непосредственно следует из определений:

$$\begin{aligned}\sum_{i,j} p_{ij}(-\log p_{ij}) &= \sum_j p_{*j}\sum_i \frac{p_{ij}}{p_{*j}}\Bigl(-\log\frac{p_{ij}}{p_{*j}}-\log p_{*j}\Bigr)=\\ &=\sum_j p_{*j}\sum_i \frac{p_{ij}}{p_{*j}}\Bigl(-\log\frac{p_{ij}}{p_{*j}}\Bigr)+\sum_j p_{*j}\sum_i\frac{p_{ij}}{p_{*j}}(-\log p_{*j})=\\ &=\sum_j p_{*j}H(\xi|\eta=\eta_j)+\sum_j p_{*j}(-\log p_{*j})=H(\xi|\eta)+H(\eta).\end{aligned}$$

Теорема доказана. □

Из этой теоремы немедленно вытекает теорема 10.5 (с. 123). Кроме того, из неё видно, что энтропия пары не меньше энтропии любого из её членов (поскольку условная энтропия неотрицательна). Отсюда легко вытекает такое утверждение:

Теорема 10.7. *Пусть ξ — случайная величина с конечным числом значений, а f — функция, определённая на множестве значений величины ξ. Тогда*

$$H(f(\xi)) \leqslant H(\xi),$$

где $f(\xi)$ — случайная величина, получающаяся применением f к ξ (формально говоря, композиция f и ξ).

С точки зрения наборов чисел переход от ξ к $f(\xi)$ означает, что мы объединяем некоторые значения (складывая соответствующие вероятности).

Доказательство. В самом деле, величина $\langle \xi, f(\xi) \rangle$ имеет в точности то же распределение, что и ξ, поэтому $H(\xi) = H(\xi, f(\xi))$, а энтропия пары не меньше энтропии второго члена пары. □

Задача 49. Укажите прямое доказательство в терминах кодирования и поиска минимума.

Задача 50. В каких случаях неравенство теоремы 10.7 обращается в равенство?

Величину $I(\alpha : \beta) \rightleftharpoons H(\beta) - H(\beta|\alpha)$ называют *количеством информации в α о β*. Как видно из формулы, оно показывает, на сколько уменьшится энтропия β, если становится известным α. Используя равенство $H(\beta|\alpha) = H(\beta, \alpha) - H(\alpha)$, можно заметить, что $I(\alpha : \beta) = H(\beta) + H(\alpha) - H(\beta, \alpha)$. Отсюда следует, что количество информации коммутативно: количество информации в α о β равно количеству информации в β о α. Поэтому часто величину $I(\alpha : \beta)$ называют *взаимной информацией* α и β.

Задача 51. Докажите, что взаимная информация обладает следующими свойствами:

$$I(\alpha : \beta) \leqslant H(\alpha),$$
$$I(\alpha : \beta) \leqslant H(\beta),$$
$$I(f(\alpha) : \beta) \leqslant I(\alpha : \beta) \qquad \text{для любой функции } f.$$

10.7 Независимость и энтропия

Понятие независимости случайных величин легко выражается в терминах энтропии. Напомним, что величины ξ и η *независимы*, если вероятность события «$\xi = \xi_i$ и $\eta = \eta_j$» равна произведению вероятностей событий «$\xi = \xi_i$» и «$\eta = \eta_j$» по отдельности. (Переформулировка: если распределение вероятностей ξ относительно условия $\eta = \eta_j$ совпадает с исходным; аналогично и для η относительно $\xi = \xi_i$.) В наших обозначениях независимость означает, что $p_{ij} = p_{i*}p_{*j}$ (матрица вероятностей имеет ранг 1).

Теорема 10.8. *Величины ξ и η независимы тогда и только тогда, когда*

$$H(\langle \xi, \eta \rangle) = H(\xi) + H(\eta),$$

то есть у α и β нулевая взаимная информация.

Другими словами, критерий независимости состоит в том, что неравенство теоремы 10.5 обращается в равенство. Используя теорему 10.6, можно переписать это равенство в виде $H(\xi) = H(\xi|\eta)$ или $H(\eta) = H(\eta|\xi)$.

Доказательство. Логарифм является строго выпуклой функцией: неравенство

$$\log\Big(\sum p_i x_i\Big) \geqslant \sum p_i \log x_i,$$

справедливое для неотрицательных p_i с единичной суммой и произвольных положительных x_i, обращается в равенство, лишь если все x_i равны (за исключением тех, которые входят с нулевыми коэффициентами p_i).

Отсюда следует, что для любых неотрицательных p_i, в сумме равных единице, минимум выражения

$$\sum p_i(-\log q_i),$$

который берётся по всем неотрицательным q_i, в сумме равным 1, достигается в единственной точке, когда $q_i = p_i$. (Надо уточнить, что при $p_i = 0$ мы полагаем $p_i(-\log q_i) = 0$ и разрешаем q_i быть нулевым.)

Теперь вспомним, что делалось при доказательстве теоремы 10.5. Минимум по матрицам ранга 1 (при котором правая часть равна сумме

энтропий) достигался при

$$q_{ij} = p_{i*} \cdot p_{*j}$$

Если он совпадает с минимумом по всем наборам q_{ij}, который достигается при $q_{ij} = p_{ij}$, то это значит, что имеет место равенство

$$p_{ij} = p_{i*} \cdot p_{*j}$$

и величины ξ и η независимы. □

Задача 52. Докажите, что величины α, β, γ независимы в совокупности (вероятность события $(\alpha = \alpha_i, \beta = \beta_j, \gamma = \gamma_k)$ равна произведению трёх отдельных вероятностей) тогда и только тогда, когда

$$H(\langle \alpha, \beta, \gamma \rangle) = H(\alpha) + H(\beta) + H(\gamma).$$

10.8 «Релятивизация» и информационные неравенства

Доказанные нами утверждения имеют свои «условные» варианты. Например, неравенство

$$H(\langle \xi, \eta \rangle) \leqslant H(\xi) + H(\eta)$$

при добавлении случайной величины α в качестве условия превращается в

$$H(\langle \xi, \eta \rangle | \alpha) \leqslant H(\xi|\alpha) + H(\eta|\alpha).$$

Нового доказательства по существу не требуется, так как при каждом значении α_i случайной величины α выполнено неравенство

$$H(\langle \xi, \eta \rangle | \alpha = \alpha_i) \leqslant H(\xi|\alpha = \alpha_i) + H(\eta|\alpha = \alpha_i)$$

(применяем неравенство для пары к условным распределениям вероятностей случайных величин ξ и η), и остаётся сложить эти неравенства с весами $\Pr[\alpha = \alpha_i]$.

Теперь можно выразить условные энтропии через безусловные, используя формулу $H(\beta|\gamma) = H(\langle \beta, \gamma \rangle) - H(\gamma)$, и привести подобные члены. Получится такое утверждение:

Теорема 10.9 (базисное неравенство)**.**

$$H(\xi,\eta,\alpha)+H(\alpha)\leqslant H(\xi,\alpha)+H(\eta,\alpha).$$

Для краткости мы опускаем угловые скобки и пишем $H(\xi,\eta,\alpha)$ вместо $H(\langle\xi,\eta,\alpha\rangle)$ или ещё более подробного $H(\langle\langle\xi,\eta\rangle,\alpha\rangle)$.

Аналогичную «релятивизацию» (добавление случайных величин как условий) можно применить и к понятию взаимной информации и определить, скажем, $I(\alpha:\beta|\gamma)$ как

$$H(\alpha|\gamma)+H(\beta|\gamma)-H(\langle\alpha,\beta\rangle|\gamma).$$

Базисное неравенство (теорема 10.9) утверждает, что $I(\alpha:\beta|\gamma)\geqslant 0$ для любых случайных величин α,β,γ.

Задача 53. Докажите, что $I(\langle\alpha,\beta\rangle:\gamma)\geqslant I(\alpha:\gamma)$.

Задача 54. Докажите, что

$$I(\langle\alpha,\beta\rangle:\gamma)=I(\alpha:\gamma)+I(\beta:\gamma|\alpha).$$

Если $I(\alpha:\gamma|\beta)=0$, говорят, что α и γ *независимы при известном* β, а также что α,β,γ образуют *марковскую цепь* («прошлое» α связано с «будущим» γ лишь через «настоящее» β).

Задача 55. Докажите, что в этом случае $I(\alpha:\gamma)\leqslant I(\alpha:\beta)$, и потому $I(\alpha:\gamma)\leqslant H(\beta)$.

При решении этих задач полезно использовать диаграммы, похожие на диаграммы Венна. Диаграмма для двух величин состоит из трёх областей, каждой из которых соответствует неотрицательное значение; суммы значений в двух областях слева равна $H(\alpha)$, а в двух областях справа — $H(\beta)$ (рис.1).

Диаграмма для трёх величин α,β,γ показана на рис. 2. Центральной области соответствует значение, которое мы обозначили через $I(\alpha:\beta:\gamma)$; его можно определить как $I(\alpha:\beta)-I(\alpha:\beta|\gamma)$, а также как $I(\alpha:\gamma)-I(\alpha:\gamma|\beta)$ и т. п. — при переходе к безусловным энтропиям получается выражение

$$\begin{aligned}I(\alpha:\beta:\gamma)=H(\alpha)+H(\beta)+H(\gamma)-\\-H(\alpha,\beta)-H(\alpha,\gamma)-H(\beta,\gamma)+H(\alpha,\beta,\gamma).\end{aligned}$$

В отличие от шести других величин на рисунке, $I(\alpha:\beta:\gamma)$ может быть отрицательной. Так будет, например, если величины α и β независимы, но зависимы при известном γ.

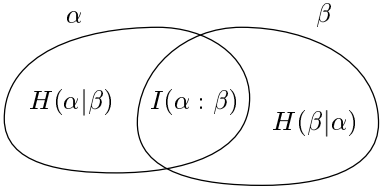

Рис. 1. Энтропии двух случайных величин.

Задача 56. Приведите пример таких величин α, β, γ. [Указание. Можно рассмотреть независимые величины α и β, равномерно распределённые на $\{0, 1\}$, и положить $\gamma = \alpha + \beta \bmod 2$.]

Задача 57. Придумайте алгоритм, который по любому линейному неравенству, содержащему энтропии α, β, γ и кортежей, составленных из них, определяет, истинно ли данное неравенство для всех α, β, γ. [Указание. Составьте диаграмму 2 и для каждой части диаграммы найдите коэффициент, с которым соответствующая часть появляется в левой части исходного неравенства (слева от знака $\geqslant$). Если все части появляются с неотрицательными коэффициентами, причем коэффициент в центральной части не превосходит суммы коэффициентов своих трех соседей, то неравенство всегда истинно, а иначе нет.]

Теорема 10.10. *Если величины α и β различаются с вероятностью ε, и число различных значений величины α равно a, то выполняется* неравенство Фано*:*

$$H(\alpha|\beta) \leqslant \varepsilon \log a + h(\varepsilon, 1-\varepsilon),$$

где $h(\varepsilon, 1-\varepsilon)$ — энтропия случайной величины с двумя значениями, имеющими вероятности ε и $1-\varepsilon$.

Доказательство. Введём величину γ, которая принимает два значения — 0 при $\alpha \neq \beta$ и 1 при $\alpha = \beta$. Тогда $H(\alpha|\beta) \leqslant H(\gamma) + H(\alpha|\beta, \gamma)$. Первое слагаемое равно $h(\varepsilon, 1-\varepsilon)$, а второе надо записать как

$$\Pr[\gamma = 0] H((\alpha|\beta)|\gamma = 0) + \Pr[\gamma = 1] H((\alpha|\beta)|\gamma = 1),$$

то есть

$$\Pr[\alpha \neq \beta] H((\alpha|\beta)|\alpha \neq \beta) + \Pr[\alpha = \beta] H((\alpha|\beta)|\alpha = \beta),$$

что не превосходит $\varepsilon \log a + 0$. □

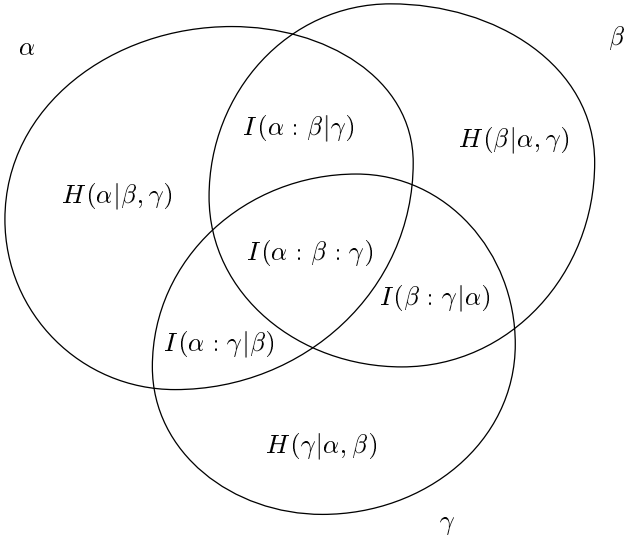

Рис. 2. Энтропии трёх случайных величин.

Теорема 10.11. *Пусть* $H(\alpha|\beta,\gamma) = 0$ *и* $I(\beta : \alpha) = 0$. *Тогда* $H(\gamma) \geqslant H(\alpha)$. *Более того, для любых* α, β, γ *выполнено* $H(\alpha) \leqslant H(\gamma) + I(\beta : \alpha) + H(\alpha|\beta,\gamma)$.

Это утверждение называют иногда *теоремой Шеннона об идеальном шифре*. (Если агент хочет передать в Центр секретное сообщение α в виде открытого текста β с помощью заранее согласованного с Центром ключа γ, причём так, чтобы враги, не знающие γ, не получили никакой информации об α, то энтропия ключа должна быть не меньше энтропии сообщения.)

Доказательство. Нарисовав диаграмму случайных величин α, β, γ, нетрудно убедиться, что разница между правой и левой частями последнего неравенства равна $I(\beta : \gamma) + H(\gamma|\alpha,\beta)$, а значит неотрицательна. Можно и не привлекать диаграммы, а вместо этого перенести $I(\beta : \alpha)$ в левую часть. Получится неравенство $H(\alpha|\beta) \leqslant H(\gamma) + H(\alpha|\beta,\gamma)$, которое следует из неравенства $H(\alpha|\beta) \leqslant H(\gamma|\beta) + H(\alpha|\beta,\gamma)$. Последнее неравенство является релятивизацией очевидного неравенства $H(\alpha) \leqslant H(\gamma) + H(\alpha|\gamma)$. □

10.9 Задачи для самостоятельной работы

Задача 58. Пусть вероятности исходов случайной величины есть $1/2$, $1/4$, $1/8$, …, $1/2^n$, $1/2^n$. К чему стремится ее энтропия, когда $n \to \infty$? Тот же вопрос для случайной величины с вероятностями исходов $1/3$, $1/3$, $1/9$, $1/9$, …, $1/3^n$, $1/3^n$, $1/3^n$.

Задача 59. Нарисуйте график функции $p \mapsto h(p, 1-p)$, задающей энтропию случайной величины с двумя исходами, имеющими вероятности $p, 1-p$. Найдите первые три члена ряда Тейлора этой функции в точке $1/2$.

Задача 60. Докажите формулу

$$h(p_1, \ldots, p_n, q_1, \ldots, q_m) = h(p_1 + \ldots + p_n, q_1 + \ldots + q_m) + \\ + (p_1 + \ldots + p_n) h(p'_1, \ldots, p'_n) + (q_1 + \ldots + q_m) h(q'_1, \ldots, q'_m),$$

где $p'_i = p_i/(p_1 + \ldots + p_n)$ и $q'_i = q_i/(q_1 + \ldots + q_m)$.

Задача 61. Докажите замечание после задачи 44. [Указание. Можно по индукции доказывать, что $\sum_i p_i(l(c_i) + \log p_i)$ не превосходит некоторой константы, умноженной на сумму p_i по всем таким i, что i-ый отрезок при каком-то из рекурсивных вызовов оказался в середине.]

Статистическим расстоянием между двумя распределениями вероятностей μ и ν называется максимум по всем событиям A величины $|\mu(A) - \nu(A)|$.

Задача 62. Докажите, что указанный максимум достигается и равен сумме половине суммы $\sum_x |\mu(x) - \nu(x)|$ (суммирование производится по всем исходам x).

Статистическим расстоянием между случайными величинами называется статистическое расстояние между их распределениями.

Задача 63. Пусть даны случайные величины α, β с n возможными исходами. Докажите, что $|H(\alpha) - H(\beta)| \leqslant d \log n + h(d, 1-d)$, где d обозначает статистическое расстояние между α, β, а $h(d, 1-d)$ есть энтропия Шеннона случайной величины с вероятностями исходов d и $1-d$. [Указание. Это следует из неравенства Фано. Для его применения надо «спарить» случайные величины α, β, то есть построить их совместное распределение так, чтобы с вероятностью $1-d$ было выполнено $\alpha = \beta$.]

Задача 64. Докажите, что $I(\langle \alpha, \beta \rangle : \gamma) \geqslant I(\alpha : \gamma)$ и что разность между левой и правой частями равна $I(\beta : \gamma | \alpha)$.

Задача 65. Доказать неравенство $2H(\alpha,\beta,\gamma) \leqslant H(\alpha,\beta) + H(\alpha,\gamma) + H(\beta,\gamma)$.

Задача 66. Докажите следующее обобщение предыдущего неравенства. Пусть $T_1,\ldots,T_k$ — произвольные кортежи, составленные из переменных $\alpha_1,\ldots,\alpha_n$, причем каждая переменная входит ровно в r кортежей. Тогда $rH(\alpha_1,\alpha_2,\ldots,\alpha_n) \leqslant H(T_1) + \ldots + H(T_k)$ (неравенство Шерера (Shearer)).

Глава 11. Кодирование текстов с учетом частотных закономерностей

Пусть имеется последовательность длины m из букв n-буквенного алфавита Σ. Известны частоты всех букв в этой последовательности, они предполагаются положительными и обозначаются в дальнейшем через $p_1, \ldots, p_n$. Требуется передать эту последовательность по каналу, способному за один такт работы передать один бит без искажений (такой канал называется бесшумным). Сколько тактов необходимо и достаточно для этого? Число m предполагается значительно бóльшим n (иначе частоты букв большого смысла не имеют).

Будем рассматривать следующие две постановки:

1. за данное количество тактов надо уметь передавать все последовательности длины m с данными частотами букв $p_1, \ldots, p_n$;
2. передаваемая последовательность длины m выбирается случайно по бернуллиевскому распределению с параметрами $p_1, \ldots, p_n$ (то есть в каждой позиции i-ая буква выбирается с вероятностью p_i независимо от других позиций); при этом кодирующая и декодирующая функции должны обеспечивать безошибочную передачу с вероятностью не менее данного порога.

Оказывается, в обеих постановках необходимо и достаточно передать примерно $mH(\alpha)$ бит, где α есть случайная величина с вероятностями исходов $p_1, \ldots, p_n$, то есть нужно примерно $H(\alpha)$ тактов на один на один исходный символ. Первая задача исследуется в разделе 11.1, а вторая в разделе 11.2.

11.1 Безошибочные кодирования

Уточним постановку задачи. Пусть имеются *кодирующая* и *декодирующая* функции E, D. Кодирующая отображает слова длины m над n-

буквенным алфавитом в двоичные слова некоторой длины k, а декодирующая, наоборот, двоичные слова длины k в слова длины m над n-буквенным алфавитом. Будем такую пару называть *корректным m,k-кодированием*, если для любого слова длины m, в котором частоты букв равны $p_1,\ldots,p_n$, выполнено равенство $D(E(x))=x$ (числа $p_1,\ldots,p_n$ предполагаются фиксированными). Нас интересует наименьшее k, для которого существует корректное m,k-кодирование. Здесь мы предполагаем, что все числа $mp_1,\ldots,mp_n$ целые (иначе слов с указанными частотами букв нет).

Наименьшее такое k равно целой части сверху от двоичного логарифма количества слов с частотами букв $p_1,\ldots,p_n$. Действительно, если k выбрано таким образом, то можно определить значение E на i-ом (в лексикографическом порядке) слове с данными частотами букв, как i-ое (в лексикографическом порядке) двоичное слово длины k. А затем определить D как обратное к E отображение. Как будет определено E на остальных словах, неважно. Будем определенную таким образом пару функций E,D называть *кодированием на основе частот букв*[5] (для данных m и $p_1,\ldots,p_n$).

Заметим, что k не может быть меньше логарифма количества слов с частотами букв $p_1,\ldots,p_n$, так как для каждого двоичного слова u существует не более одного x, для которого $D(u)=x$, $E(x)=u$.

Количество слов с частотами букв $p_1,\ldots,p_n$ равно биномиальному коэффициенту

$$C_m^{p_1m,\ldots,p_nm}=\frac{m!}{(p_1m)!\cdots(p_nm)!}$$

По формуле Стирлинга $m!\sim\sqrt{2\pi m}(m/e)^m$. Поэтому

$$\begin{aligned}\log\frac{m!}{(p_1m)!\cdots(p_nm)!}&=\\&=(n\log m)/2-mp_1\log p_1-\ldots-mp_n\log p_n+O(1)=\\&=mH(\alpha)+(n\log m)/2+O(1).\end{aligned}$$

Итак, мы доказали следующее утверждение.

Теорема 11.1. *Пусть фиксированы неотрицательные рациональные числа $p_1,\ldots,p_n$ в сумме равные 1. Пусть d обозначает НОК их знаменателей. Тогда для всех m, кратных d, наименьшее k, для которого*

[5] Кажется, общепринятого названия для этого кода нет.

существует m, k-кодирование, есть $mH(\alpha) + O(\log m)$ (константа в $O(\log m)$ зависит от $p_1, \ldots, p_n$).

За сколько шагов можно найти $E(x)$ и $D(y)$ (в зависимости от длин слов x, y) для этого кодирования? Эта задача сводится к нахождению по слову x его номера в лексикографическом порядке на словах с теми же, что и у x, частотами букв и, обратно, самого слова по его номеру.

Задача 67. Пусть дано слово x. Докажите, что за полиномиальное от длины x количество шагов можно найти номер x в лексикографическом порядке на словах той же, что и x, длины и с теми же, что и у x, частотами букв.

Задача 68. Пусть даны натуральные числа $k_1, \ldots, k_n$ и еще одно натуральное число i. Докажите, что за полиномиальное от $m = k_1 + \ldots + k_n$ число шагов можно найти i-ое в лексикографическом порядке слово на словах длины m с частотами букв $k_1/m, \ldots, k_n/m$.

Теперь немного изменим постановку задачи. А именно, предположим, что частоты букв заранее не фиксированы. В этом случае в начало кода можно приписать частоты букв, на что потребуется $O(n \log m)$ битов. При большом m (по сравнению с n) эта добавка будет незначительной по сравнению с $mH(\alpha)$.

На практике, однако, ситуация промежуточная — нам известны частоты букв, но лишь приблизительно. В этой ситуации в начало кода можно дописать лишь отклонения фактических частот от известных приближений к ним, что даст небольшую экономию.

11.2 Кодирования с ошибками: теорема Шеннона

Пусть опять фиксированы числа $p_1, \ldots, p_n$ и кодируемая последовательность x выбирается случайно по бернуллиевскому распределению с параметрами $p_1, \ldots, p_n$. При декодировании нам разрешается давать неправильный ответ с некоторой вероятностью ошибки ε. Оказывается, минимально возможная длина кода мало зависит от ε. Как и в предыдущем разделе, необходимо и достаточно передать примерно $mH(\alpha)$ бит.

Уточним постановку задачи. Пару функций E, D будем называть *ε-корректным* m, k-кодированием, если E отображает слова длины m над n-буквенным алфавитом в двоичные слова длины k, а D, наоборот,

двоичные слова длины k в слова длины m над n-буквенным алфавитом и при этом вероятность события $D(E(x)) \neq x$ не превосходит ε.

Теорема 11.2. *Для всех $p_1, \ldots, p_n$ и всех положительных ε найдется такое c, что для почти всех m существует ε-корректное m,k-кодирование с $k \leqslant mH(\alpha) + c\sqrt{m}$. Обратно, для всех $p_1, \ldots, p_n$ и всех $\varepsilon < 1$ найдётся константа c такая, что для всех достаточно больших m и всех m, k, ε-кодирований выполнено $k \geqslant mH(\alpha) - c\sqrt{m}$.*

Как видно из формулировки, главный член $mH(\alpha)$ не зависит от допустимой вероятности ошибки, от нее зависит лишь добавочный член меньшего порядка. Это утверждение называют *Shannon noiseless coding theorem* (теорема Шеннона о кодировании без помех). Имеется в виду, что в теореме изучается передача информации по каналу, не вносящему помех.

Доказательство. Зафиксируем $p_1, \ldots, p_n, \varepsilon$. Сначала докажем первое утверждение. Для этого укажем сразу, какие именно слова будут правильно декодироваться. Это будут слова, в которых частоты букв близки к $p_1, \ldots, p_n$. Точнее, выберем некоторое b и назовем слово x длины m над n-буквенным алфавитом *сбалансированным*, если для всех i частота вхождения i-ой буквы в него отличается от p_i не более чем на $b/\sqrt{m}$. Величину b выберем из условия, чтобы вероятность несбалансированности была меньше ε.

По неравенству Чебышёва вероятность того, что частота вхождений i-ой буквы в случайном слове отличается от p_i не более чем на δ, не превосходит $\frac{1}{\delta^2 m}$. При $\delta = b/\sqrt{m}$ эта вероятность не превосходит $1/b^2$, поэтому можно выбрать настолько большое b, чтобы она была меньше ε/n (для всех i). При таком b вероятность несбалансированности будет меньше ε для всех m.

Теперь нам осталось подсчитать количество сбалансированных слов и убедиться, что его логарифм не больше $mH + O(\sqrt{m})$. Действительно, если это уже доказано, то можно определить значение E на i-ом сбалансированном слове (в лексикографическом порядке) как i-ое двоичное слово длины $k = mH + O(\sqrt{m})$. А затем определить D, как обратное к E отображение. Как будет определено E на несбалансированных словах, неважно.

Количество сбалансированных слов есть сумма биномиальных коэффициентов, и его можно оценить по формуле Стирлинга. Но мы сде-

лаем проще: мы оценим вероятность каждого сбалансированного слова и докажем, что она равна $2^{-mH+O(\sqrt{m})}$. Поскольку суммарная вероятность всех сбалансированных слов не может превосходить 1, отсюда следует требуемая верхняя оценка $2^{mH+O(\sqrt{m})}$ количества сбалансированных слов.

Вероятность любого слова есть произведение p_i в степени, равной количеству вхождений i-ой буквы. Для сбалансированных слов это произведение равно

$$\prod_i p_i^{mp_i+O(\sqrt{m})} = 2^{\sum_i (\log p_i)(mp_i+O(\sqrt{m}))} = 2^{-mH+O(\sqrt{m})},$$

что и требовалось доказать.

Теперь докажем второе утверждение. Оно доказывается «обратными рассуждениями». Пусть имеется m, k, ε-кодирование E, D. Опять рассмотрим сбалансированные слова. Но на этот раз выберем b таким большим, чтобы суммарная вероятность несбалансированных слов была меньше $(1-\varepsilon)/2$ при всех достаточно больших m. Тогда суммарная вероятность сбалансированных слов x, для которых $D(E(x)) = x$, будет не меньше $(1-\varepsilon)/2$. Действительно, по условию доля слов, для которых $D(E(x)) = x$, не меньше $1-\varepsilon$. Причем не более чем $(1-\varepsilon)/2$ из них являются несбалансированными.

Теперь оценим сверху суммарную вероятность сбалансированных слов x, для которых $D(E(x)) = x$. Как мы видели, вероятность каждого сбалансированного слова не больше $2^{-mH+O(\sqrt{m})}$. По условию, слов, для которых $D(E(x)) = x$, не больше 2^k (действительно, для каждого двоичного слова u длины k существует не более одного x, для которого $D(u) = x$, $E(x) = u$). Поэтому суммарная вероятность сбалансированных слов x, для которых $D(E(x)) = x$, не превосходит $2^{k-mH+O(\sqrt{m})}$.

Соединив верхнюю и нижнюю оценки, мы получаем

$$(1-\varepsilon)/2 \leqslant 2^{k-mH+O(\sqrt{m})},$$

откуда и следует утверждение теоремы. □

Первое утверждение теоремы можно доказать и с помощью арифметического кода (см. с. 42). А именно, рассмотрим слова длины m в n-буквенном алфавите, как новые буквы, и рассмотрим на буквах нового алфавита то же самое распределение вероятностей, что и в теореме. Напомним, что при арифметическом кодировании длина l кодового

слова буквы связана с её вероятностью p неравенством $l < -\log p + 2$. Поэтому длины кодовых слов всех сбалансированных слов не превосходят $mH + O(\sqrt{m})$. Остальные слова кодирующая функция может отображать куда угодно. Поэтому будет достаточно $mH + O(\sqrt{m})$ бит.

Задача 69. Пусть рациональные числа $p_1, \ldots, p_n$ фиксированы. Докажите, что за полиномиальное от m число шагов по данному слову длины m можно найти его код для арифметического кодирования, соответствующего бернуллиевскому распределению вероятностей на словах длины m.

На практике описанные выше кодирования применяются для фиксированного достаточно большого m. Если длина исходной последовательности больше m, то ее разрезают на блоки длины m и кодируют блоки по отдельности. Такое кодирование называют *блочным*.

Задача 70. Пусть $p_1 \geqslant p_2 \geqslant \ldots \geqslant p_n$ и $i \leqslant m$.
(а) Докажите, что

$$h(p_1/(p_1 + \ldots + p_i), \ldots, p_i/(p_1 + \ldots + p_i)) \leqslant h(p_1, \ldots, p_n).$$

(б) Докажите, что

$$h(p_1, \ldots, p_i, \varepsilon) \leqslant h(p_1, \ldots, p_n)(1 - \varepsilon) + h(\varepsilon, 1 - \varepsilon),$$

где $\varepsilon = p_{i+1} + \ldots + p_n$.

Задача 71. Пусть $p_1, p_2, \ldots, p_n$ — вероятности букв $a_1, a_2, \ldots, a_n$, а p — действительное число от 0 до 1. Докажите, что существует кодирующая и декодирующая функции E, D такие, что с вероятностью не менее p выполнено $D(E(a_i)) = a_i$ и при этом средняя длина $E(a_i)$ не превосходит $ph(p_1, p_2, \ldots, p_n) + 2$.

11.3 Учёт частот пар, троек и т. д.

Для реальных текстов кодирование на основе частот букв далеко от оптимального, поскольку не учитывает зависимостей между соседними буквами. Например, частота вхождения слога `ла` в «среднестатистический» русский текст больше, чем произведение частот вхождений букв `л` и `а` по отдельности. Аналогичное справедливо для троек букв и так

далее. Поэтому представляется разумным обобщить теорему Шеннона и коды, использованные в ней, в следующем направлении.

Пусть дано слово x длины m в n-буквенном алфавите. Для каждого двух-буквенного слова u в n-буквенном алфавите (в этом контексте такие слова называются *диграммами (биграммами)*) обозначим через $p(u)$ частоту вхождения u в x. Она определяется как деленное на $(m-1)$ число позиций, начиная с которых в x написано слово u. Допустим, что нам известны числа $p(u)$ для всех диграмм u. Используя эту информацию, мы хотим придумать более короткий код, чем бернуллиев.

Опять возможны два различных уточнения:

1. кодирование должно быть безошибочным для всех слов с данными частотами диграмм;
2. разрешена некоторая вероятность ошибки, при этом исходная последовательность берется по некоторому естественному распределению, гарантирующему с высокой вероятностью близость частот диграмм к данным нам значениям.

Эти постановки исследуются по очереди в следующих двух разделах.

11.3.1 Безошибочные кодирования

Пусть для каждой диграммы w (в n-буквенном алфавите) задано неотрицательное число $p(w)$, и в сумме все заданные числа дают 1. Будем слово x длины m в n-буквенном алфавите называть *правильным*, если для всех w частота вхождений w в x в точности равна $p(w)$. Будем пару функций E, D, первая из которых отображает слова длины m над n-буквенным алфавитом в двоичные слова некоторой длины k, а вторая отображает двоичные слова длины k в слова длины m над n-буквенным алфавитом, называть *корректным m,k-кодированием*, если для любого правильного слова выполнено равенство $D(E(x)) = x$. Нас интересует наименьшее k, для которого существуют корректные m,k-кодирования. Очевидно, что оно равно двоичному логарифму количества правильных слов.

Как оценить количество правильных слов? Для оценки сверху применим тот же метод, что и в предыдущем разделе. А именно, для каждой буквы a обозначим через $p(a)$ частоту вхождений буквы a в пра-

вильные слова (без учета последней буквы):

$$p(a) = \sum_b p(ab).$$

Обозначим также частоту вхождений в правильные слова буквы b после буквы a через

$$p(b|a) \rightleftharpoons \frac{p(ab)}{p(a)}.$$

Далее рассмотрим следующее распределение вероятностей μ на словах длины m:

$$\mu(y_1 \dots y_m) = \frac{1}{n} \prod_{i=1}^{m-1} p(y_{i+1}|y_i). \tag{11.1}$$

Нетрудно убедиться, что это в самом деле распределение вероятностей (сумма указанных чисел равна 1); оно соответствует следующему процессу порождения случайного слова. Первая буква выбирается случайно с равномерным распределением (множитель $1/n$), а затем каждая следующая буква берется в зависимости от предыдущей по правилам марковской цепи: вероятность появления b после a равна $p(b|a)$.

Если слово x правильно, то

$$\mu(x) = \frac{1}{n} \prod_{a,b} p(b|a)^{(m-1)p(ab)} = 2^{\log n + (m-1)\sum_{a,b} p(a)\log p(b|a)} =$$

$$= 2^{\log n - (m-1)H(\beta|\alpha)}.$$

Здесь через α, β обозначены случайные величины, получаемые следующим образом: выбираем случайным образом в правильном слове две подряд идущих буквы и обозначаем первую через α, а вторую через β. Количество правильных слов не превосходит обратного к этому числу, следовательно, не больше $2^{mH(\beta|\alpha)+O(1)}$.

Итак, мы доказали, следующую теорему.

Теорема 11.3. *Пусть для каждой биграммы w зафиксировано неотрицательное рациональное число $p(w)$ и в сумме эти числа дают 1. Тогда для всех m существует корректное существует m,k-кодирование с $k \leqslant mH(\beta|\alpha) + O(1)$ (константа в $O(1)$ зависит от чисел $p(w)$).*

Теорема дает только верхнюю оценку для минимально возможного k. Что можно сказать о нижней оценке? Ясно, что частоты биграмм можно подобрать так, что правильных слов не будет вовсе (даже если НОД их знаменателей кратен m). Например, не существует слов в алфавите 0,1, в которых количества вхождений биграмм 01 и 10 отличаются более чем на 1 (докажите это!). А если правильных слов нет, то для любого k существует m, k-кодирование, и, следовательно, минимальное k равно нулю (что может быть намного меньше $mH(\beta|\alpha)$). С другой стороны, если для данных частот биграмм существует хотя бы одно правильное слово, то можно показать, что логарифм их количества примерно равен $mH(\beta|\alpha)$, а значит, верхняя оценка теоремы 11.3 точна.

Теорема 11.4. *Пусть для каждой биграммы $w \in \Sigma^2$ зафиксировано неотрицательное рациональное число $p(w)$ и в сумме эти числа дают 1. Пусть дано m, для которого существует слово длины m с частотами биграмм $\{p(w) \mid w \in \Sigma^2\}$. Тогда логарифм количества таких слов не меньше $mH(\beta|\alpha) + O(\log m)$ (константа в $O(\log m)$ зависит от чисел $p(w)$). Здесь $\alpha\beta$ обозначает случайную величину, которая принимает значение w с вероятностью $p(w)$ для всех биграмм $w \in \Sigma^2$.*

Доказательство. Рассмотрим ориентированный граф, с кратными ребрами и петлями, вершины которого суть буквы исходного алфавита Σ, и между вершинами a и b имеется $(m-1)p(ab)$ ребер. Пусть x — любое правильное слово длины m (в котором частоты биграмм равны $p(w)$). Обозначим первую букву x через u_0, а последнюю — v_0. По условию граф имеет эйлеров путь u_0 в v_0 (вдоль слова x). Следовательно, входная и выходная степень любой вершины, кроме u_0, v_0, совпадают и граф слабо связен. (Если $u_0 = v_0$, то это верно и для u_0, v_0, а иначе у u_0 выходная степень на 1 больше входной, а у v_0 — наоборот.) Эйлеровы пути в этом графе соответствуют правильным словам, начинающимся и заканчивающимся на ту же букву, что и x. Поэтому нам достаточно показать, что количество эйлеровых путей в нём не меньше $2^{mH(\beta|\alpha)+O(\log m)}$.

Для этого мы покажем, что эйлеровы пути в этом графе можно строить следующим жадным алгоритмом. Этот алгоритм будет использовать некоторые фиксированные $n = |\Sigma|$ рёберно непересекающихся простых по ребрам циклов $C_1, \ldots, C_n$, каждый из которых проходит через все вершины и имеет не более n^2 ребер. Позже мы докажем,

что если m достаточно велико по сравнению с n и для каждой буквы из Σ вероятность хотя бы одной диграммы, содержащей эту букву, положительна, то такие циклы существуют. А сейчас продолжим доказательство, предположив, что это так.

Удалим из графа ребра всех циклов $C_1, \ldots, C_n$. Жадный алгоритм действует так. Сначала он строит простой по ребрам путь P из u_0 в v_0 и не более n простых по ребрам циклов $D_1, \ldots, D_k$ таких, что каждое ребро графа принадлежит ровно одному из путей $P, D_1, \ldots, D_k$. Затем с помощью циклов $C_1, \ldots, C_n$ пути $P, D_1, \ldots, D_k$ соединяются в один эйлеров путь из u_0 в v_0.

Построение пути P: из вершины u_0 пойдем по любому ребру, затем по любому еще не использованному ребру, затем опять по любому еще не использованному ребру и так далее. На каждом шаге мы выбираем любое из рёбер, еще не включённых в путь. Мы запнёмся только когда путь придёт в вершину v_0 (если удалить все рёбра текущей части пути, то единственной вершиной, у которой входная степень больше выходной, будет v_0). При этом все выходные ребра из v_0 будут исчерпаны.

Построение циклов $D_1, \ldots, D_k$: если путь P не исчерпывает все ребра, то выбираем первое не пройденное ребро (u_1, t_1), далее строим некоторый путь из u_1 в u_1 по оставшимся свободными ребрам, пока не исчерпаем все выходящие из u_1 ребра. Если после этого еще не все ребра пройдены, то берем первое не пройденное ребро (u_2, t_2). И так далее.

Соединение путей и циклов в эйлеров путь: сначала идем вдоль пути P, затем из v_0 идем вдоль цикла C_1 в u_1, затем проходим весь цикл D_1, затем по неиспользованным ребрам цикла C_1 возвращаемся в v_0, затем идем вдоль цикла C_2 в u_2 и так далее. После того, как все циклы $D_1, \ldots, D_k$ будут пройдены, мы проходим все неиспользованные циклы из числа $C_1, \ldots, C_n$. (Конец описания жадного алгоритма.)

При каждой последовательности выборов жадного алгоритма будет построен уникальный эйлеров путь: просматривая ребра из пути, мы можем понять, какие выборы были сделаны алгоритмом и также понять, когда он запнулся, найти первое не пройденное ребро и так далее.

Поэтому число эйлеровых путей из u_0 в v_0 не меньше количества способов осуществления жадным алгоритмом своих выборов. Последнее очевидно равно произведению по всем вершинам σ_j биномиального коэффициента $C^{c_{j1},\ldots,c_{jn}}_{c_{j1}+\ldots+c_{jn}}$, где c_{ji} есть количество ребер из σ_j в σ_i. С точностью до слагаемого $O(\log c)$ логарифм биномиального коэффици-

ента $C_c^{c_1,\ldots,c_n}$ равен $ch(c_1/c,\ldots,c_n/c)$. Удаление из графа ребер циклов $C_1,\ldots,C_n$ уменьшает выходные степени всех вершин лишь на константу (зависящую от n). Поэтому c_{ji} равно $p(\sigma_j\sigma_i)m+O(1)$. Обозначим через $p(\sigma_j)$ сумму $\sum_i p(\sigma_j\sigma_i)$. Тогда с точностью до слагаемого $O(\log m)$ логарифм количества способов выбора жадного алгоритма равен сумме по всем j величины

$$\sum_j mp(\sigma_j)h\left(\frac{p(\sigma_j\sigma_1)}{p(\sigma_j)},\ldots,\frac{p(\sigma_j\sigma_n)}{p(\sigma_j)}\right).$$

Эта сумма, как нетрудно понять, и есть $mH(\beta|\alpha)$.

Итак, осталось доказать существование циклов $C_1,\ldots,C_n$. Как уже отмечалось, наш орграф слабо связен (любая пара различных вершин u,v связана неориентированным путем). Любой слабо связный граф, в котором входная степень любой вершины равна ее выходной степени, является и сильно связным. Действительно, иначе отношение «u достижимо из v ориентированным путем» задавало бы предпорядок, в котором более одного класса эквивалентности (u эквивалентно v, если они достижимы друг из друга ориентированным путем). Тогда в вершины любого наибольшего класса эквивалентности входит в совокупности больше ребер, чем выходит. Какое отношение имеет это рассуждение к нашему графу? В нём условие равенства входной и выходной степени выполнено с точностью до добавления 1. Поскольку между любыми соседними вершинами линейное (по m) количество ребер, на рассуждения эта разница не влияет (при достаточно большом m).

Итак, граф сильно связен, а поэтому существует цикл C, проходящий через все вершины. Поскольку между любыми соседними вершинами линейное количество ребер, мы можем считать, что все ребра C попарно различны. Рассмотрим такой цикл C, имеющий наименьшую длину, и докажем, что его длина меньше n^2. Будем двигаться по циклу, начиная с некоторой его вершины, и следить за множеством пройденных вершин. Отметим в цикле те моменты, когда это множество увеличивается. Между любыми такими моментами не более n вершин, поскольку не может быть двух повторяющихся вершин (всю часть цикла между двумя повторениями можно выбросить).

Итак, существует рёберно простой цикл длины не более n^2, проходящий по всем вершинам. Выбросим из графа все ребра этого цикла. Если m достаточно большое, то между любыми смежными вершина-

ми не меньше n^3 рёбер. Значит, мы можем повторить эту операцию не менее n раз. □

Напомним, что длина кода на основе частот букв примерно равна $mH(\beta) \approx mH(\alpha)$. Поскольку $H(\beta|\alpha) \leqslant H(\beta)$, построенный код не длиннее кода на основе частот букв, а при больших m и при наличии зависимости между соседними буквами он и строго короче. Например, новый код значительно лучше старого для слова $x = 01010101\ldots$. Частоты обеих букв в этом слове равны $1/2$, поэтому кодирование на основе частот букв дает кодовое слово длины m. С другой стороны $H(\beta|\alpha)$ равно нулю, поскольку каждая позиция в этом слове однозначно определяется предыдущей. Поэтому новый код будет иметь длину, пренебрежимо малую по сравнению с m.

Как и раньше, если частоты диграмм неизвестны, то их можно написать в начало кода — длина кодового слова увеличится незначительно, если m достаточно велико. Аналогичным образом можно учитывать зависимости между тройками букв. Для троек получаемый код имеет длину примерно $mH(\gamma|\beta\alpha) \leqslant mH(\beta|\alpha)$, где $\alpha\beta\gamma$ обозначает случайно выбранное подслово длины 3 в исходном слове x. Точно так же можно учитывать зависимости между l подряд идущими буквами для произвольного l. Чем больше l, тем более экономичный код мы получим. Только надо иметь в виду, что если частоты комбинаций заранее не известны, то при большом l дополнительная информация о них в размере $O(n^l \log n)$ бит может стать больше получаемой экономии. Чтобы в этом случае в самом деле получалась экономия, m должно быть значительно больше n^l.

11.3.2 Кодирования с ошибками

Пусть фиксировано неотрицательное целое l и для каждого слова w длины $l+1$ в n-буквенном алфавите задано неотрицательное число $\pi(w)$, сумма всех $\pi(w)$ по всем словам длины $l+1$ равна 1. Пусть эти числа удовлетворяют условию *стационарности*:

$$\sum_a \pi(ua) = \sum_b \pi(bu).$$

Число, задаваемое этой формулой, мы будем обозначать через $\pi(u)$. Заметим, что частоты вхождения $(l+1)$-буквенных блоков в любую

фиксированную последовательность «почти» удовлетворяют этому равенству. Точнее, левая часть равенства есть частота вхождения слова w во всех позициях, кроме последней, а правая часть — частота вхождения u во всех позициях, кроме первой. Поэтому левая и правая часть могут отличаться максимум на $1/(m-l-2)$.

Определение 1. Рассмотрим марковскую цепь, состояниями которой являются слова w длины $l+1$ с положительным $\pi(w)$, а переходы имеют вид $au \to ub$, где a, b — произвольные буквы, а u — l-буквенное слово. Вероятность такого перехода по определению есть

$$\frac{\pi(ub)}{\pi(u)}$$

(она не зависит от a).

Как нетрудно видеть, распределение π является стационарным для этой марковской цепи:

$$\sum_a \pi(au)\frac{\pi(ub)}{\pi(u)} = \pi(ub)$$

для всех $(l+1)$-буквенных блоков ub. Эта марковская цепь задает следующий процесс порождения случайного слова длины $m \geqslant l+1$. Первые $l+1$ букв берутся случайным образом с распределением π. А затем мы применяем марковскую цепь для получения $l+2$-ой буквы, потом опять применяем марковскую цепь и так $m-l-1$ раз. Обозначим так определенное распределение на словах длины m через μ. Будем предполагать еще, что марковская цепь *эргодична*. Это означает, что найдется такое N, что для всех состояний u, v марковская цепь за N шагов с положительной вероятностью переходит из u в v.

Будем называть пару функций E, D m, k, ε-кодированием, если E отображает слова длины m над n-буквенным алфавитом в двоичные слова длины k, а D, наоборот, двоичные слова длины k в слова длины m над n-буквенным алфавитом и при этом μ-вероятность события $D(E(x)) \neq x$ не превосходит ε. Рассмотрим случайные величины α, β, получаемые в результате следующего процесса. Генерируем случайное слово длины $l+1$ с распределением π. Последняя его буква есть β, а слово, полученное отрезанием последней буквы есть α.

Теорема 11.5. *Для всех положительных ε найдется такое c, что для почти всех m существует m, k, ε-кодирование с $k \leqslant mH(\beta|\alpha) + c\sqrt{m}$. Обратно, для всех $\varepsilon < 1$ найдётся c такое, что для всех достаточно больших m и всех m, k, ε-кодирований выполнено $k \geqslant mH(\beta|\alpha) - c\sqrt{m}$.*

Эта теорема обобщает теорему Шеннона (последняя получится, если положить $l = 0$) и ее доказательство есть прямое обобщение доказательства теоремы Шеннона.

Доказательство. Утверждение теоремы не изменится, если в качестве случайной величины β взять взять целиком случайное слово длины $l + 1$ (а α оставить прежним). В такой форме мы и будем доказывать теорему.

Первое утверждение. Назовем слово x длины m над n-буквенным алфавитом *сбалансированным*, если для всех слов w длины $l+1$ частота вхождения w в x отличается от $\pi(w)$ не более чем на $d/\sqrt{m}$. Величину d выберем из условия, чтобы вероятность несбалансированности была менее ε (при всех достаточно больших m). Почему найдется такое d? Нам понадобится центральная предельная теорема для эргодических марковских цепей [7]. В простейшем варианте теорема гласит, что для любой эргодической марковской цепи, любого ее начального состояния и любого состояния q, распределение случайной величины ((количество посещений состояния q в первые m моментов времени) $-\pi(q))/\sqrt{m}$ стремится с ростом m к нормальному распределению. Здесь π обозначает стационарное распределение марковской цепи. То есть для любого b вероятность того, что частота вхождений диграммы w отличается от $\pi(w)$ более чем на $d/\sqrt{m}$, стремится к интегралу

$$\frac{1}{\sigma\sqrt{2\pi}} \int_{-d}^{d} e^{-u^2/2\sigma^2}\, du,$$

где σ не зависит от m. Выберем настолько большое d, чтобы этот интеграл был больше $1 - \varepsilon/2n^2$ (для всех диграмм). Для такого d при всех достаточно больших m вероятность того, что отклонение частоты каждой диграммы w от $\pi(w)$ превысит $d/\sqrt{m}$, будет меньше ε/n^2. Поэтому, вероятность несбалансированности меньше ε.

В любом сбалансированном слове частота вхождения слова w длины $l+1$ равна $\pi(w)+O(1/\sqrt{m})$. Значит, для любого слова u длины l и любой

буквы b частота вхождения u равна

$$\pi(u) + O(1/\sqrt{m}),$$

а частота вхождения b после u равна

$$\frac{\pi(ub)}{\pi(u)} + O(1/\sqrt{m}).$$

Следовательно, вероятность каждого сбалансированного слова равна

$$\prod_{u,b} \Big(\frac{\pi(ub)}{\pi(u)}\Big)^{m\pi(ub)+O(\sqrt{m})} = 2^{-mH(\beta|\alpha)+O(\sqrt{m})}.$$

Поэтому количество сбалансированных слов не больше $2^{mH(\beta|\alpha)+O(\sqrt{m})}$, а значит, их можно закодировать словами длины $k = mH(\beta|\alpha) + O(\sqrt{m})$.

Второе утверждение доказывается так: опять рассмотрим сбалансированные слова. Но на этот раз выберем b таким большим, чтобы суммарная вероятность несбалансированных слов была меньше $(1-\varepsilon)/2$ при всех достаточно больших m. Тогда суммарная вероятность сбалансированных слов x, для которых $D(E(x)) = x$, будет не меньше $(1-\varepsilon)/2$. Действительно, по условию доля слов, для которых $D(E(x)) = x$, не меньше $1-\varepsilon$. Причем не более чем $(1-\varepsilon)/2$ из них являются несбалансированными.

Теперь оценим сверху суммарную вероятность сбалансированных слов x, для которых $D(E(x)) = x$. Как мы видели, вероятность каждого сбалансированного слова не больше $2^{-mH+O(\sqrt{m})}$. По условию, слов, для которых $D(E(x)) = x$, не больше 2^k (действительно, для каждого двоичного слова u длины k существует не более одного x, для которого $D(u) = x$, $E(x) = u$). Поэтому суммарная вероятность сбалансированных слов x, для которых $D(E(x)) = x$, не превосходит $2^{k-mH+O(\sqrt{m})}$.

Соединив верхнюю и нижнюю оценки, мы получаем

$$(1-\varepsilon)/2 \leqslant 2^{k-mH+O(\sqrt{m})},$$

откуда и следует утверждение теоремы. □

Итак, резюмируем сказанное выше. Пусть имеется текст x, частота вхождений $(l+1)$-буквенных блоков в который известна. Пусть X_i обозначает случайную величину, равную i-ой букве случайно выбранного

подслова в x длины $l+1$. Тогда x (как и все другие слова длины m с теми же частотами вхождений блоков длины $l+1$) можно закодировать двоичным словом длины примерно $mH(X_{l+1}|X_1\ldots X_l)$. Кроме того, если слово x выбирается случайным образом так, что вероятность $(l+1)$-ой буквы при известных l предыдущих определяется данными нам частотами, то существует двоичное кодирование с небольшой вероятностью ошибки и длиной кода примерно $mH(X_{l+1}|X_1\ldots X_l)$, причем длину кода существенно понизить нельзя.

11.3.3 Учет частот $(l+1)$-буквенных сочетаний с помощью разбиения на блоки длины $l+1$

Частоты $(l+1)$-буквенных сочетаний можно учитывать и другим образом: разбить слово x на блоки длины $l+1$, рассмотреть каждый из них как букву расширенного алфавита и применить кодирование на основе частот букв к полученному слову $\overline{x}$.

Естественно предположить, что для каждой буквы w расширенного алфавита её частота в $\overline{x}$ примерно равна доли позиций, начиная с которых слово w входит в x. Иными словами, две следующих случайных величины имеют примерно одинаковое распределение: блок длины $l+1$ в x, начальная позиция которого выбирается случайно среди чисел, кратных $l+1$, и блок длины $l+1$ в x, начальная позиция которого выбирается случайно среди всех чисел от 1 до $m-l$. При этом предположении длина кода на основе частот букв слова $\overline{x}$ будет примерно равна $mH(X_1\ldots X_{l+1})/(l+1)$. Возникает вопрос: какой из двух способов выгоднее? То есть что больше — $H(X_{l+1}|X_1\ldots X_l)$ или $H(X_1\ldots X_{l+1})/(l+1)$? Оказывается, при условии стационарности первый способ не хуже второго.

Поставим вопрос более формально. Пусть дана бесконечная последовательность случайных величин $X_1, X_2, \ldots$ со значениями в данном алфавите такая, что распределение случайной величины $X_i, X_{i+1}, \ldots, X_{i+l}$ зависит только от l (но не от i). Такие последовательности называются *стационарными источниками* букв данного алфавита.

Задача 72. Пусть $X_0, X_1, X_2, \ldots$ — состояния марковской цепи из определения 1 в моменты времени $0, 1, 2, \ldots$. Докажите, что $X_0, X_1, X_2, \ldots$ является стационарным источником.

Итак, пусть дан стационарный источник $X_0, X_1, X_2, \ldots$. Что можно сказать о соотношениях между величинами

$$mH(X_{l+1}|X_1 \ldots X_l) \quad \text{и} \quad \frac{H(X_1 \ldots X_{l+1})}{l+1}\,?$$

Ответ дается в следующих задачах.

Задача 73. Докажите, что

$$H(X_l|X_1, X_2, \ldots, X_{l-1}) \geqslant H(X_{l+1}|X_1, X_2, \ldots, X_l).$$

Это означает, что с ростом l экономность первого способа улучшается (если частоты известны).

Задача 74. Докажите, что

$$\frac{H(X_1, X_2, \ldots, X_l)}{l} \geqslant H(X_l|X_1, X_2, \ldots, X_{l-1}).$$

Это означает, что первый способ учета частот l-буквенных сочетаний не хуже второго.

Задача 75. Докажите, что

$$\frac{H(X_1, X_2, \ldots, X_l)}{l} \leqslant \frac{H(X_1, X_2, \ldots, X_{l+1})}{l+1}.$$

Это означает, что с ростом l экономность второго способа улучшается (если частоты известны).

Задача 76. Докажите, что

$$\lim_{l\to\infty} \frac{H(X_1, X_2, \ldots, X_l)}{l} = \lim_{l\to\infty} H(X_l|X_1, X_2, \ldots, X_{l-1}).$$

Таким образом, при очень больших l оба способа дают код примерно одной длины.

11.4 Передача информации при наличии дополнительной информации у принимающей стороны. Теорема Вольфа–Слепяна

Пусть имеются две совместно распределенные случайные величины α, β, принимающие значения в алфавитах Γ, Σ. Рассмотрим следую-

щую ситуацию. У передающей стороны имеется слово α^k длины k в алфавите Γ, а у принимающей — слово β^k длины k в алфавите Σ. Эти слова получены в результате k независимых испытаний пары (α, β). Сколько битов необходимо передать, чтобы принимающая сторона смогла найти α^k, если допустить небольшую вероятность ошибки ε?

Сначала рассмотрим упрощенную задачу, в которой передающая сторона знает не только α^k, но и β^k (при этом обе стороны знают Γ, Σ, k и совместное распределение α, β). Тогда можно действовать следующим образом. Зафиксируем некоторую константу c и будем называть пару строк $(a, b) \in \Gamma^k \times \Sigma^k$ *сбалансированной*, если для любой пары букв $(\gamma, \sigma) \in \Gamma \times \Sigma$ частота вхождений этой пары в слово (a, b) отличается от вероятности $p^k(a, b) = \Pr(\alpha^k = a, \beta^k = b)$ этой пары не более, чем на $c/\sqrt{k}$. При достаточно большом c с вероятностью не менее $1 - \varepsilon$ случайно взятая пара слов α^k, β^k является сбалансированной, а значит, $p^k(\alpha^k|\beta^k) = 2^{-H(\alpha|\beta)k + O(\sqrt{k})}$. Зная исход b случайной величины β^k, передающий кодирует свое слово a кодом Шеннона — Фано длины

$$n = -\log p^k(\alpha^k|\beta^k) = H(\alpha|\beta)k + O(\sqrt{k}),$$

если пара (a, b) сбалансирована (и кодирует чем угодно иначе). Вероятность ошибки будет не больше ε, поскольку суммарная вероятность несбалансированных пар не превосходит ε.

Итак, мы доказали следующее утверждение: *для любого положительного ε и случайных величин α, β найдется такое c, что для всех k существует пара функций*

$$E : \Gamma^k \times \Sigma^k \to \{0,1\}^n, \qquad D : \{0,1\}^n \times \Sigma^k \to \Gamma^k$$

(кодер и декодер), где $n = H(\alpha|\beta)k + c\sqrt{k}$, для которых

$$D(E(\alpha^k, \beta^k), \beta^k) = \alpha^k$$

с вероятностью не менее $1 - \varepsilon$

Задача 77. Докажите, что значение n в этом утверждении нельзя существенно уменьшить: если существуют функции с указанными свойствами, то $n \geqslant H(\alpha|\beta)k - c\sqrt{k}$, где c не зависит от k, но зависит от $\varepsilon, \alpha, \beta$. [Указание. Используйте рассуждения из доказательства теоремы Шеннона о бесшумном канале.]

Оказывается, аналогичное утверждение справедливо и в том случае, когда передающий не знает исхода β^k. Это утверждение называется теоремой Вольфа — Слепяна.

Теорема 11.6. *Для любого положительного ε и случайных величин α, β найдется такое c, что для всех k существует пара функций*

$$E : \Gamma^k \to \{0,1\}^n, \qquad D : \{0,1\}^n \times \Sigma^k \to \Gamma^k$$

(кодер и декодер), где $n = H(\alpha|\beta)k + c\sqrt{k}$, для которых

$$D(E(\alpha^k), \beta^k) = \alpha^k$$

с вероятностью не менее $1 - \varepsilon$.

Отличие от предыдущего утверждения в том, что теперь E имеет только один аргумент.

Доказательство. Выберем в определении сбалансированности константу c таким образом, чтобы случайная пара (α^k, β^k) была близкой к единице. Насколько близкой, мы поймём позднее (это будет зависеть от ε). Через T будем обозначать множество всех сбалансированных пар. Для любого $a \in \Gamma^k$ через $T(a)$ будем обозначать множество всех слов b, для которых пара (a, b) сбалансирована. Аналогично определим $T(b)$ для $b \in \Sigma^k$. Множество $T(a)$ будем называть *тенью* a. Через $T(X)$ (для некоторого множества слов $X \subset \Sigma^k$) будем обозначать объединение теней всех слов из X.

Наш план таков. Мы выделим в множестве всех возможных сообщений Γ^k попарно непересекающихся части $X_1, \ldots, X_N$ со следующими свойствами. Во-первых, $\Pr[\alpha^k \in X_1 \cup \ldots \cup X_N] \geqslant 1 - \varepsilon/2$. Во-вторых, для любого $i \leqslant N$ тень любого слова $a \in X_i$ слабо пересекается с объединением теней всех предшествующих a слов из X_i (относительно упорядочения X_i, которое мы тоже определим). Точнее, при условии $\alpha^k = a$ условная вероятность того, что β^k принадлежит тени a, но не принадлежит тени ни одного из предшествующих a слов из X_i, не меньше $1 - \varepsilon/2$. Кодирующая функция E будет для данного слова a выдавать то i, для которого $a \in X_i$ (если таких i нет, то значение E равно любому i). Декодирующая функция D на данной паре входов (b, i) будет выдавать первый элемент a из X_i, для которого b принадлежит тени a. Если b не принадлежит тени никакого слова из X_i, то

D выдает что угодно. Из свойств $X_1, \ldots, X_N$ следует, что вероятность неправильного декодирования не больше ε. Действительно, эта вероятность складывается из двух вероятностей: вероятности того, что α^k не принадлежит ни одному из $X_1, \ldots, X_N$ (по условию эта вероятность меньше $\varepsilon/2$) и суммы по всем a из объединения $X_1, \ldots, X_N$ вероятности того, что $\alpha_k = a$ и β^k не принадлежит тени a или принадлежит тени одного из предшествующих a слов. По условию при любом фиксированном a эта вероятность не превосходит $\varepsilon/2$, поэтому и второе слагаемое не больше $\varepsilon/2$. Количество частей N не будет превосходить $2^{H(\alpha|\beta)k+O(\sqrt{k})}$, что и обеспечит требуемую верхнюю оценку длины кода.

Далее переходим к реализации плана.

Итак, осталось построить множества сообщений $X_1, \ldots, X_N$ с требуемыми свойствами. Мы будем строить их по очереди. Сначала построим X_1, начав с пустого множества и добавляя в него слова по очереди так, чтобы тень очередного элемента мало пересекалась с объединением теней предыдущих элементов. Добавлять новые элементы с сохранением этого свойства мы сможем до тех пор, пока $\Pr[\beta^k \in T(X_1)]$ не станет достаточно большим. После этого мы перейдем к X_2 и будем формировать его таким же образом. Но на этот раз добавляемые к X_2 сообщения будем выбирать из дополнения к X_1. Опять же, процесс добавления мы сможем продолжать до тех пор, пока $\Pr[\beta^k \in T(X_2)]$ не станет достаточно большим. И так далее. Мы закончим в тот момент, когда станет выполнено неравенство $\Pr[\alpha^k \in X_1 \cup \ldots \cup X_N] \geqslant 1 - \varepsilon/2$. Поскольку по построению $X_1, \ldots, X_N$ дизъюнктны и каждое из них довольно большое, общее их количество будет небольшим.

Для реализации этого плана нам понадобится следующая лемма. В ее формулировке под X следует понимать уже построенную часть очередного множества X_i, а под Y — дополнение к $X_1 \cup \ldots \cup X_{i-1}$.

Лемма 11.1. *Для любой пары множеств $X \subset Y \subset \Gamma^k$, если*

$$\Pr[\alpha^k \in Y] \geqslant \varepsilon/2$$

(Y не слишком мало) и

$$\Pr[\beta^k \in T(X)] \leqslant \varepsilon^2/8$$

(тень X мала), то существует $a \in Y$ такое, что

$$\Pr[\beta^k \in T(a) \setminus T(X) | \alpha^k = a] \geqslant 1 - \varepsilon/2$$

(тень a мало пересекается с объединением теней слов из X, а значит его можно добавить к X).

Доказательство леммы. Выберем случайным образом пару слов (a, b), используя для этого случайную величину $(\alpha^k, \beta^k)|(\alpha^k \in Y)$. Выбор будем считать удачным, если оказалось, что пара (a, b) сбалансирована и b не принадлежит $T(X)$. Оценим сверху вероятность неудачи.

Выберем константу c в определении сбалансированности так, чтобы вероятность несбалансированности случайной пары (α^k, β^k) была меньшей $\varepsilon^2/8$. Тогда безусловная вероятность неудачи (вероятность события «$(\alpha^k, \beta^k) \notin T$ или $\beta^k \in T(X)$») не превосходит $\varepsilon^2/8 + \varepsilon^2/8 = \varepsilon^2/4$. Условная вероятность неудачи не больше безусловной вероятности, деленной на вероятность условия, то есть не больше $(\varepsilon^2/4)/(\varepsilon/2) = \varepsilon/2$. Отсюда следует, что для некоторого фиксированного $a \in Y$ условная вероятность неудачи не больше этого числа. □

Сначала построим X_1. Для этого положим $Y = \Gamma^k$ и применим лемму к $X = \emptyset$. Мы найдем элемент a_1 со свойством

$$\Pr[\beta^k \in T(a_1) | \alpha^k = a_1] \geqslant 1 - \varepsilon/2$$

и добавим его в X. Вторично применив лемму, мы найдем a_2 со свойством

$$\Pr[\beta^k \in T(a_2) \setminus T(a_1) | \alpha^k = a_2] \geqslant 1 - \varepsilon/2$$

и опять добавим его в X. И так будем добавлять элементы в X до тех пор, пока $\Pr[\beta^k \in T(X)] \leqslant \varepsilon^2/8$. Как только это условие станет ложным, закончим добавлять слова в X и положим $X_1 = X$.

Теперь положим $Y = \Gamma^k \setminus X_1$ и таким же образом построим X_2. Затем положим $Y = \Gamma^k \setminus (X_1 \cup X_2)$ и построим X_3 и так далее. Мы остановимся, когда очередное $Y = \Gamma^k \setminus (X_1 \cup X_2 \cup \ldots \cup X_N)$ перестанет удовлетворять условию леммы, то есть $\Pr[\alpha^k \in Y] < \varepsilon/2$. В этот момент построенные множества обладают требуемыми свойствами и нам осталось только оценить сверху N.

Верхняя оценка для N получается из следующего соображения. Весом пары (a, b) будем называть $\Pr(\beta^k = b)$. По построению, для каждого i сумма весов всех пар, у которых первая компонента принадлежит X_i, не меньше $\varepsilon^2/8$. Поэтому N не превосходит общего веса всех сбалансированных пар, деленного на $\varepsilon^2/8$. Итак, нам нужно оценить сумму

весов всех сбалансированных пар:

$$\sum_{(a,b)\in T} \Pr(\beta^k = b) = \sum_{b\in B} \Pr(\beta^k = b) \times |T(b)|.$$

Здесь B обозначает множество всех вторых компонент сбалансированных пар. Оценим сверху $|T(b)|$ (для произвольного $b \in B$). Из сбалансированности для всех $a \in T(b)$ следует, что вероятность $\Pr[\alpha^k = a|\beta^k = b]$ равна $2^{-H(\alpha|\beta)k+O(\sqrt{k})}$, поэтому $|T(b)|$ не превосходит обратного к этому числа. Итак, мы получаем оценку

$$\sum_{b\in B} \Pr(\beta^k = b) \times |T(b)| = 2^{H(\alpha|\beta)k+O(\sqrt{k})} \sum_{b\in B} \Pr(\beta^k = b) \leqslant$$
$$\leqslant 2^{H(\alpha|\beta)k+O(\sqrt{k})}.$$

Следовательно, N не превосходит

$$2^{H(\alpha|\beta)k+O(\sqrt{k})}/(\varepsilon^2/8) = 2^{H(\alpha|\beta)k+O(\sqrt{k})},$$

что и требовалось доказать. □

11.5 Каналы с помехами

11.5.1 Пропускная способность канала с помехами

Канал с помехами задается своим входным и выходным алфавитами $\Sigma = \{\sigma_1, \ldots, \sigma_l\}, T = \{\tau_1, \ldots, \tau_m\}$ и стохастической матрицей $W \in \mathbb{R}^{m\times l}$ (все элементы неотрицательны и сумма элементов в каждой строке равна единице). Столбцы матрицы соответствуют символам входного алфавита, а строки — символам выходного алфавита. Элемент $W_{\sigma\tau}$ матрицы есть вероятность того, что на выходе канала будет буква τ, при условии, что на вход канала подается буква σ. Случайную величину со значениями в выходном алфавите, задаваемую строчкой номер σ матрицы W мы будем обозначать через $W(\sigma)$.

Никакого распределения на входных буквах в определении канала нет. Однако пропускная способность канала определяется с помощью таких распределений. Пусть α — произвольная случайная величина со значениями во входном алфавите канала. «Применив канал» к случайной величине α, мы получим некоторую новую случайную величину, которую мы будем обозначать через $\beta = W(\alpha)$.

По определению совместное распределение α, β задается равенством $\Pr[\alpha = i, \beta = j] = W_{ij} \cdot \Pr[\alpha = i]$.

Определение. Пропускная способность $I(W)$ канала с матрицей W определяется как максимум по всем α общей информации α и $W(\alpha)$:

$$I(W) = \max_{\alpha} I(\alpha : W(\alpha)).$$

Примеры. Канал с матрицей $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ имеет пропускную способность 1 (в качестве α надо взять равномерно распределенную случайную величину). Этот канал не имеет помех и просто копирует вход на выход. Канал с матрицей $\begin{pmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{pmatrix}$ имеет пропускную способность 0. Выходной символ в этом канале никак не зависит от входа. Канал с матрицей $\begin{pmatrix} 1/2 & 1/2 \\ 1/3 & 2/3 \end{pmatrix}$ имеет пропускную способность между нулем и единицей. Выходной символ в этом канале зависит от входа, но не сохраняет полную информацию о входе.

11.5.2 Использование каналов с помехами

Если передать один бит по третьему каналу из предыдущего примера, то выходной бит не даст почти никакой информации о входном бите. Тем не менее, канал можно использовать для надежной передачи одного бита информации, если сначала закодировать каждый из двух битов достаточно длинной последовательностью нулей и единиц и передавать по каналу коды вместо самих битов. Например, в качестве кода нуля можно использовать достаточно длинную последовательность из одних нулей, а в качестве кода единицы — достаточно длинную последовательность из единиц. В самом деле, если передать много нулей подряд, то на выходе с большой вероятностью будет примерно поровну единиц и нулей, а если передавать много единиц подряд, то на выходе с большой вероятностью единиц будет примерно вдвое больше, чем нулей. Таким образом по этому каналу можно довольно надежно передать один бит информации.

Если нам нужно передать не один бит, а достаточно длинную последовательность битов, то можно разбить ее на блоки и кодировать каждый блок по отдельности. Требование состоит в том, чтобы после внесения помех каналом каждый блок можно было декодировать с

ничтожной вероятностью ошибки. Теорема о канале с помехами оценивает сколько символов необходимо и достаточно передавать по каналу в расчете на один бит исходной последовательности.

Итак, будем передавать исходную последовательность по каналу с помехами следующим образом. Сначала разрезаем ее на блоки некоторой длины n, затем кодируем каждый блок $x = b_1 \dots b_n$ несколькими буквами входного алфавита канала, затем каждую из кодирующих букв передаем по каналу и полученную последовательность выходных букв канала декодируем, восстанавливая исходный блок. То есть передача одного блока x осуществляется следующим образом

$$x \to \boxed{\text{E}} \to \text{канал} \to \boxed{\text{D}} \to x.$$

Здесь E — отображение из $\{0,1\}^n$ в Σ^k, а D — отображение из T^k в $\{0,1\}^n$.

Такой способ использования канала называется блочным n,k-кодированием (n — длина кодируемого блока, а k — длина его кода), а отношение n/k называется скоростью передачи информации. Мы предполагаем, что каждую из k букв канал передает независимо, то есть при подаче на вход k-буквенного слова $a = \sigma_1 \dots \sigma_k$, на выходе появится слово $b = \tau_1 \dots \tau_k$ с вероятностью, равной произведению чисел $W_{\sigma_1\tau_1}, \dots, W_{\sigma_k\tau_k}$. Будем случайную величину с этим распределением обозначать через $W^k(a)$.

Пусть ε — действительное число на интервале $(0,1)$. Будем блочное кодирование, задаваемое функциями E, D, называть $(1-\varepsilon)$-надежным, если для любого входного слова $x \in \{0,1\}^n$ вероятность события

$$D(W^k(E(x))) = x$$

не меньше $1-\varepsilon$. При данных k, ε мы хотим найти наибольшее n, для которого существует $(1-\varepsilon)$-надежное n,k-кодирование, то есть существует $(1-\varepsilon)$-надежная пара отображений $E : \{0,1\}^n \to \Sigma^k$ и $D : T^k \to \{0,1\}^n$. Отношение n/k для максимального такого n называется максимальной скоростью $(1-\varepsilon)$-надежной передачи информации при k-кратном использовании канала. Оказывается, при достаточно больших k максимально возможная скорость передачи информации примерно равна пропускной способности канала и слабо зависит от ε. Точнее, при любом фиксированном $0 < \varepsilon < 1$ максимально возможная скорость при k-кратном использовании канала стремится к пропускной

способности канала, когда k стремится к бесконечности. Более точные оценки даются в следующей теореме.

Пусть фиксирован канал с пропускной способностью I.

Теорема 11.7. *(1) Для любых $0 < \varepsilon < 1$ и $\delta > 0$ при всех достаточно больших k, для любого $(1-\varepsilon)$-надежного n,k-кодирования выполнено $n/k < I + \delta$. (2) Для любого $\varepsilon > 0$ существует такое c, что для всех $k > 0$ существует $(1-\varepsilon)$-надежное n,k-кодирование с $n > Ik - c\sqrt{k}$.*

Доказательство. (1) Мы докажем первое утверждение в ослабленной формулировке: *для любых $\varepsilon > 0$, $k > 0$ для любого $(1-\varepsilon)$-надежного n,k-кодирования выполнено*

$$\frac{n}{k} \leqslant \frac{I + 1/k}{1-\varepsilon}. \tag{11.2}$$

Из ослабленной формулировки следует лишь, что предел при $\varepsilon \to 0$ и $k \to \infty$ максимальной скорости передачи не превосходит пропускной способности канала.

Итак, пусть даны $\varepsilon > 0$ и k и $(1-\varepsilon)$-надежное n,k-кодирование E, D. Рассмотрим случайную величину α', которая равномерно распределена среди всех n-битовых слов. Рассмотрим также случайные величины, которые получаются при кодировании, передаче по каналу и декодировании этой случайной величины: $\alpha = E(\alpha')$, $\beta = W^k(\alpha)$, $\beta' = D(\beta)$:

$$\alpha' \to \boxed{\text{E}} \to \alpha \to \text{канал} \to \beta \to \boxed{\text{D}} \to \beta'.$$

Докажем, что общая информация α и β не меньше $n(1-\varepsilon) - 1$. Сначала заметим, что поскольку $\alpha' \to \alpha \to \beta \to \beta'$ есть марковская цепь, мы можем заключить, что

$$I(\alpha : \beta) \geqslant I(\alpha' : \beta) \geqslant I(\alpha' : \beta').$$

Теперь применим неравенство Фано к случайным величинам α', β'. По условию, эти случайные величины различаются с вероятностью не более ε, а значит, $H(\alpha'|\beta') \leqslant n\varepsilon + 1$, следовательно

$$I(\alpha' : \beta') = H(\alpha') - H(\alpha'|\beta) = n - H(\alpha'|\beta') \geqslant n - \varepsilon n - 1.$$

Поскольку $I(\alpha : \beta)$ не меньше $I(\alpha' : \beta')$, это неравенство выполнено и для $I(\alpha : \beta)$. Напомним, что значениями α являются k-буквенные слова над входным алфавитом канала, а значениями β —

k-буквенные слова над выходным алфавитом канала. Поэтому пару случайных величин (α, β) можно рассматривать как последовательность пар $(\alpha_1, \beta_1), \ldots, (\alpha_k, \beta_k)$. Докажем, что для хотя бы одной пары в этой последовательности общая информация первой и второй ее компонент не меньше $I(\alpha : \beta)/k$, что и завершит доказательство, поскольку $\beta_i = W(\alpha_i)$ для всех i. Для этого выпишем неравенство

$$\begin{aligned} I(\alpha : \beta) = H(\beta) - H(\beta|\alpha) \leqslant H(\beta_1) + \ldots + H(\beta_k) - H(\beta|\alpha) = \\ = H(\beta_1) + \ldots + H(\beta_k) - \big(H(\beta_1|\alpha) + \ldots + H(\beta_k|\alpha)\big). \end{aligned}$$

Последнее равенство здесь выполнено, поскольку $\beta_1, \ldots, \beta_k$ независимы при известном α (после фиксации всех k входных букв, выходные буквы определяются каналом независимо друг от друга). Осталось понять, что $H(\beta_i|\alpha) = H(\beta_i|\alpha_i)$ при всех i. Действительно, обозначим через α^i кортеж $(\alpha_1 \ldots \alpha_k)$ с вычеркнутым i-ым членом. Тогда случайные величины $\alpha^i, \alpha_i, \beta_i$ образуют марковскую цепь, следовательно $H(\beta_i|\alpha_i, \alpha^i) = H(\beta_i|\alpha_i)$. Итак, мы установили, что

$$I(\alpha : \beta) \leqslant I(\alpha_1 : \beta_1) + \ldots + I(\alpha_k : \beta_k).$$

Как уже установлено, левая часть этого неравенства не меньше $n - n\varepsilon - 1$. С другой стороны, правая часть не больше Ik, а значит $Ik \geqslant n - n\varepsilon - 1$, что и требовалось доказать.

(2) Зафиксируем случайную величину α такую, что общая информация α и $\beta = W(\alpha)$ равна пропускной способности канала I. Зафиксируем также $\varepsilon > 0$ и $k > 0$. Через (α^k, β^k) мы будем обозначать k независимых копий случайной величины (α, β).

Нам нужно построить множество X из $2^n = 2^{H(\alpha:\beta)k - O(\sqrt{k})}$ слов $x_1, \ldots, x_{2^n} \in \Sigma^k$, которые будут использованы как коды исходных 2^n сообщений. Кодирующая функция E будет отображать i-ое в лексикографическом порядке бинарное слово длины n в слово x_i. Что нам нужно от слов $x_1, \ldots, x_{2^n}$? Нам требуется хорошая отличимость случайных величин $W^k(x_i)$ друг от друга. Точнее, нам нужно, чтобы существовала декодирующая функция D', которая по $W^k(x_i)$ с вероятностью не менее $1 - \varepsilon$ восстанавливает x_i (декодирующая функция D будет композицией D' и E^{-1}).

Для построения X и D' мы будем использовать сбалансированные слова (как и в теореме Вольфа-Слепяна). Точнее, для каждой пары

букв (σ, τ) обозначим через $p(\sigma, \tau)$ вероятность этой пары, то есть вероятность события $\alpha = \sigma, \beta = \tau$. Среднее количество вхождений пары (σ, τ) в случайное слово $(\alpha_1, \beta_1) \dots (\alpha_k, \beta_k)$ равно $kp(\sigma, \tau)$. Зафиксируем некоторое неотрицательное действительное число c. Назовем пару слов $(a, b) \in \Sigma^k \times T^k$ *сбалансированной*, если для любой пары букв (σ, τ) положительной вероятности количество вхождений этой пары букв в слово $(a_1, b_1) \dots (a_k, b_k)$ отличается от ожидаемого количества $kp(\sigma, \tau)$ не более чем на $c\sqrt{k}$. Используя неравенство Чебышёва, мы можем подобрать c настолько большим, что при всех k с вероятностью не менее $1 - \varepsilon/2$ пара слов (α^k, β^k) является сбалансированной. Зафиксируем любое такое c.

Вероятности всех сбалансированных пар слов близки: для любой сбалансированной пары слов (a, b) вероятность события $\alpha^k = a, \beta^k = b$ равна $2^{-H(\alpha,\beta)k+O(\sqrt{k})}$. Действительно, эта вероятность равна $\prod p(\sigma, \tau)^{p(\sigma,\tau)k+O(\sqrt{k})}$, где произведение берется по всем парам положительной вероятности. То же самое относится и к маргинальным и условным вероятностям: $\Pr(\alpha^k = a) = 2^{-H(\alpha)k+O(\sqrt{k})}$, $\Pr(\beta^k = b | \alpha^k = a) = 2^{-H(\beta|\alpha)k+O(\sqrt{k})}$ и так далее.

Пусть T обозначает множество всех сбалансированных пар, A — множество всех первых слов сбалансированных пар, а B — множество всех вторых слов сбалансированных пар. Для $a \in A$ обозначим через $T(a)$ *тень* a, то есть множество всех слов b, для которых пара $(a, b) \in T$. Мы будем использовать оценку

$$|T(a)| \leqslant 2^{H(\beta|\alpha)k+O(\sqrt{k})}.$$

Это неравенство выполнено, поскольку для каждой сбалансированной пары (a, b) условная вероятность $\Pr(\beta^k = b | \alpha^k = a)$ не меньше $2^{-H(\beta|\alpha)k-O(\sqrt{k})}$, поэтому $|T(a)|$ не превосходит величины, обратной к этой.

Значение декодирующей функции D' на слове $b \in T^k$ будет равно первому такому x_i, для которого $b \in T(x_i)$. Если b не принадлежит объединению теней слов из X, то определим $D'(b)$ произвольным образом. Нам нужно подобрать слова $x_1, x_2, \dots$ так, чтобы для любого i вероятность события $W(x_i) \in T(x_i) \setminus (T(x_1) \cup \dots \cup T(x_{i-1}))$ была не меньше $1 - \varepsilon$. Будем подбирать их по очереди. Очередное x_i будем искать, используя следующую лемму.

Лемма 11.2. *Пусть $X \subset T^k$ любое множество такое, что*

$$\Pr[\beta^k \in T(X)] \leqslant \varepsilon/2.$$

Тогда существует $a \in A \setminus X$ такое, что

$$W(a) \in T(a) \setminus T(X)$$

с вероятностью не менее $1 - \varepsilon$.

Доказательство леммы. Выберем случайным образом пару слов (a, b), используя случайную величину (α^k, β^k). Выбор будем считать удачным, если оказалось, что пара (a, b) сбалансирована и b не принадлежит $T(X)$. Эксперимент может быть неудачным по двум причинам: (a, b) не сбалансирована, вероятность этого (в силу выбора константы в определении сбалансированности) не превосходит $\varepsilon/2$, и $b \in T(X)$, вероятность этого по условию не больше $\Pr[\beta^k \in T(X)] \leqslant \varepsilon/2$. Таким образом, выбор удачен с вероятностью не менее $1 - \varepsilon/2 - \varepsilon/2 = 1 - \varepsilon$. Отсюда следует, что для некоторого фиксированного $a \in A$ условная вероятность удачи не меньше этого числа. (Конец доказательства леммы.)

Будем по этой лемме увеличивать X до тех пор, пока выполнено условие леммы, то есть $\Pr[\beta^k \in T(X)] \leqslant \varepsilon/2$. В результате мы построим множество X для которого $\Pr[\beta^k \in T(X)] > \varepsilon/2$. Докажем, что отсюда следует, что X настолько большое, как нам нужно, то есть $|X| \geqslant 2^{I(\beta:\alpha)k - O(\sqrt{k})}$. Действительно, вероятность события $\beta^k \in T(X)$ не превосходит суммы $\Pr(\beta^k = b)$ по всем $b \in T(X)$. Суммируемые величины не превосходят $2^{-H(\beta)k + O(\sqrt{k})}$, а мощность $T(X)$ не больше $|X| 2^{H(\beta|\alpha)k + O(\sqrt{k})}$. Следовательно,

$$\begin{aligned} \varepsilon/2 < \Pr[\beta^k \in T(X)] &\leqslant |X| 2^{H(\beta|\alpha)k + O(\sqrt{k})} 2^{-H(\beta)k + O(\sqrt{k})} = \\ &= |X| 2^{-I(\alpha:\beta)k + O(\sqrt{k})}, \end{aligned}$$

откуда $|X| \geqslant 2^{I(\beta:\alpha)k - O(\sqrt{k})} / (\varepsilon/2) = 2^{I(\beta:\alpha)k - O(\sqrt{k})}$. □

Задача 78. *Параллельным соединением* двух каналов с матрицами W^1, W^2 и алфавитами $\Sigma_1, \Sigma_2, \Delta_1, \Delta_2$ называется канал с входным алфавитом $\Sigma_1 \times \Sigma_2$, выходным алфавитом $\Delta_1 \times \Delta_2$, матрица которого есть тензорное произведение W^1, W^2 (то есть вероятность преобразования

пары (σ_1, σ_2) в пару (δ_1, δ_2) равна произведению $W^1_{\sigma_1\delta_1} W^2_{\sigma_2\delta_2}$). Докажите, что пропускная способность параллельного соединения равна сумме пропускных способностей исходных каналов.

Задача 79. Доказать, что пропускная способность канала равна нулю тогда и только тогда, когда все строки матрицы канала одинаковы.

Задача 80. Пусть первый канал имеет матрицу W и алфавиты Σ, Δ, а второй канал — матрицу V и алфавиты Δ, Γ. Их *последовательным соединением* называется канал с алфавитами Σ, Γ и матрицей VW (произведение матриц). Докажите, что пропускная способность последовательного соединения каналов не больше пропускной способности каждого из исходных каналов.

Задача 81. Докажите, что пропускная способность последовательного соединения каналов не определяется однозначно пропускными способностями исходных каналов.

Задача 82. Пусть даны два канала с матрицами W^1, W^2 и алфавитами $\Sigma_1, \Sigma_2, \Delta_1, \Delta_2$, причем Σ_1 не пересекается с Σ_2, а Δ_1 не пересекается с Δ_2. Их *прямой суммой* называется канал с входным алфавитом $\Sigma_1 \cup \Sigma_2$, выходным алфавитом $\Delta_1 \cup \Delta_2$, матрица которого получится, если разместить матрицы W^1, W^2 по диагонали, а в незаполненные места записать нули (то есть вероятность преобразования букв из Σ_1 в буквы из Δ_1 такая же, как у первого канала, вероятность преобразования букв из Σ_2 в буквы из Δ_2 такая же, как у второго канала, а вероятность преобразования букв из Σ_1 в буквы из Δ_2 и букв из Σ_2 в буквы из Δ_1 нулевая). Обозначим через c_1, c_2, c пропускные способности первого, второго и полученного каналов, соответственно. Докажите формулу $2^c = 2^{c_1} + 2^{c_2}$.

Глава 12. Предсказания и игры

В этом разделе изучается следующий сценарий. Игрок в казино должен предсказывать результаты бросаний, выполненных крупье. В простейшей ситуации (которой мы и ограничимся) каждое бросание имеет два исхода, обозначаемых -1 и $+1$, а предсказанием является некоторое действительное число на отрезке $[-1, 1]$, понимаемое, как оценка шансов появления $+1$ в результате очередного бросания. Игроку известны результаты всех выполненных к текущему моменту бросаний. Таким образом, возможными стратегиями действий игрока являются функции, которые сопоставляют каждой последовательности из -1 и $+1$ (составленной из исходов уже совершённых бросаний) некоторое число на отрезке $[-1, 1]$.

Пусть $x_1, \ldots, x_k$ обозначает произвольную последовательность исходов бросаний, сделанных крупье, а $p_1, \ldots, p_k$ обозначает произвольную последовательность предсказаний, сделанных игроком. Как оценить качество этих предсказаний для этой последовательности исходов? Один из наиболее естественных способов состоит в том, что за предсказания игрок отвечает своими деньгами, как и происходит в реальном казино. А именно, будем считать, что у игрока в каждый момент имеется капитал (начальный капитал примем за 1), и предсказание p означает, что $\frac{1+p}{2}$-ая часть текущего капитала ставится на 1, а остаток (то есть, $\frac{1-p}{2}$-ая часть текущего капитала) ставится на -1. Ставка, сделанная на выпавший исход, возвращается игроку в удвоенном размере, а ставка, сделанная на противоположный исход, отбирается в пользу казино. Качество предсказаний будем оценивать текущим капиталом игрока (чем он больше, тем предсказания лучше). Нетрудно убедиться, что после выпадения исходов $x_1, \ldots, x_k$ и сделанных предсказаний $p_1, \ldots, p_k$ капитал игрока выражается формулой

$$(1 + x_1 p_1) \cdot (1 + x_2 p_2) \cdot \cdots \cdot (1 + x_k p_k). \tag{12.1}$$

Пусть игрок руководствуется некоторой стратегией игры S (которая сообщает, какую надо сделать ставку, исходя из последовательности исходов, выпавших к данному моменту). И пусть x — последовательность плюс/минус единиц длины k. Обозначим через $C_S(x)$ текущий капитал игрока, руководствующегося стратегией S, после k бросаний с исходами $x_1, \ldots, x_k$. В соответствии с формулой (12.1),

$$C_S(x) = (1 + x_1 S(\Lambda)) \cdot (1 + x_2 S(x_1)) \cdot \cdots \cdot (1 + x_k S(x_1 \ldots x_{k-1}))$$

(через Λ мы обозначаем пустую последовательность). Что можно сказать о функции $x \mapsto C_S(x)$?

Теорема 12.1. *Для любой стратегии S функция $C_S(x)$ принимает неотрицательные значения, причем*

$$C_S(\Lambda) = 1 \qquad \text{и} \qquad C_S(x) = \frac{C_S(x(-1)) + C_S(x1)}{2}$$

для всех x. Обратно, для любой функции $C(x)$, удовлетворяющей этим условиям (такие функции называются мартингалами*) существует стратегия игры S, для которой $C = C_S$.*

Доказательство. Равенство $C_S(\Lambda) = 1$ выполнено по условию. Неотрицательность функции C_S очевидна.

Равенство $C_S(x) = \frac{C_S(x(-1)) + C_S(x1)}{2}$ доказывается так. Пусть в текущий момент капитал игрока равен $C_S(x)$, а предсказание равно p. По правилам выплаты,

$$C(x(-1)) = 2 \cdot (C(x)(1-p)/2) = C(x)(1-p),$$
$$C(x1) = 2 \cdot (C(x)(1+p)/2) = C(x)(1+p).$$

Отсюда сразу следует доказываемое равенство.

Обратно, пусть имеется произвольный мартингал $C(x)$. Тогда

$$C(x1) - C(x) = C(x) - C(x(-1))$$

для всех x. Рассмотрим следующую стратегию игры S: после исходов $x = x_1 \ldots x_k$ делаем предсказание

$$\frac{C(x1)}{C(x)} - 1 = 1 - \frac{C(x(-1))}{C(x)}.$$

Эта величина находится на отрезке $[-1, 1]$, поскольку, с одной стороны, она получена добавлением неотрицательного числа к -1, а с другой стороны, она получена вычитанием неотрицательного числа из 1. В случае исхода 1 капитал станет равным

$$C(x)\Big(1 + \Big(\frac{C(x1)}{C(x)} - 1\Big)\Big) = C(x1),$$

как мы и хотели. Аналогичным образом, в случае исхода -1 капитал будет равен

$$C(x)\Big(1 - \Big(1 - \frac{C(x(-1))}{C(x)}\Big)\Big) = C(x(-1)),$$

как и требуется. □

Пусть всего казино делает n бросаний. Какие значения может принимать функция $C_S(x)$ на словах длины n (то есть в конце игры)?

Теорема 12.2. *Среднее значение функции $C_S(x)$ по всем словам из ± 1 длины n равно 1. Обратно, для любой неотрицательной функции $C(x)$, определенной на словах длины n, среднее значение которой равно 1, существует стратегия игры S, для которой $C(x) = C_S(x)$ для всех строк x из ± 1 длины n.*

Доказательство. По предыдущей теореме, функция $C(x)$ является мартингалом. Среднее значение мартингала на словах любой длины n равно 1, что легко доказать индукцией по n.

Обратно, пусть $C(x)$ удовлетворяет условию. Продолжим ее на все слова длины не более n, положив $C(y)$ равным среднему арифметическому $C(x)$ по всем продолжениям x длины n слова y. Очевидно, C является мартингалом и по предыдущей теореме есть C_S для некоторой стратегии S. □

Задача 83. Пусть казино делает n бросаний так, что последовательность исходов x заведомо принадлежит некоторому множеству $A \subset \{-1, 1\}^n$. Пусть игроку стало известно множество A. Докажите, что у игрока есть стратегия, гарантирующая заключительный капитал не менее $2^n/|A|$, какая бы последовательность исходов $x \in A$ ни выпала.

Напомним, что мы оцениваем качество предсказаний с помощью формулы (12.1), отражающей игру в казино со ставками на два равновероятных исхода в пределах текущего капитала. На практике используются и другие формулы. Говорят, например, что американское бюро погоды ($+1$ соответствует наличию дождя, -1 — его отсутствию) за неточные предсказания подвергается штрафу, пропорциональному величине

$$(x_1 - p_1)^2 + (x_2 - p_2)^2 + \cdots + (x_k - p_k)^2.$$

(чем выше эта величина, тем предсказание хуже). Легко видеть, что при идеально точном предсказании штраф, вычисленный по этой формуле, равен нулю, а иначе положителен. Можно вместо возведения в квадрат в этой формуле использовать взятие абсолютной величины:

$$|x_1 - p_1| + |x_2 - p_2| + \cdots + |x_k - p_k|.$$

Наконец, для вычисления штрафа можно использовать минус логарифм формулы (12.1):

$$-\log(1 + x_1 p_1) - \log(1 + x_2 p_2) - \cdots - \log(1 + x_k p_k) \qquad (12.2)$$

(минус нужен, чтобы сохранить монотонность — чем хуже предсказание, тем больше штраф, а логарифм нужен, чтобы общий штраф стал равен сумме отдельных штрафов).

Все три рассмотренных формулы для вычисления штрафа имеют следующий общий вид. Зафиксирована некоторая функция $\rho : [-1, 1] \times \{-1, 1\} \to \mathbb{R}$ для вычисления штрафа в одном акте предсказания: если очередное предсказание равно $p \in [-1, 1]$, а исход равен $q \in \{-1, 1\}$, то начисляется штраф в размере $\rho(p, q)$, который добавляется к уже имеющемуся штрафу. То есть общий штраф при предсказаниях $p_1, \ldots, p_k$ и исходах $x_1, \ldots, x_k$ вычисляется по формуле

$$\rho(p_1, x_1) + \rho(p_2, x_2) + \cdots + \rho(p_k, x_k).$$

Возникает естественный вопрос. Пусть фиксирована функция штрафа ρ. Можно ли построить какую-нибудь хорошую стратегию предсказаний, то есть стратегию, которая в том или ином смысле хорошо предсказывает *все* последовательности? В идеале мы хотели бы построить стратегию, которая *любую* последовательность предсказывает не хуже, чем *любая* другая стратегия. К сожалению, этот идеал недостижим (для любой из трёх рассмотренных выше функций

штрафа). Действительно, для любой конкретной последовательности $x_1, \ldots, x_k$ оптимальной будет стратегия, выдающая в качестве предсказаний символы этой конкретной последовательности, а все другие стратегии предсказывают эту стратегию строго хуже этой. Поскольку при любом $k > 0$ существует более одной последовательности длины k, то и оптимальной стратегии не существует.

Поэтому умерим наши запросы и поставим вопрос следующим образом: пусть дано конечное или даже счетное семейство стратегий $S_1, S_2, \ldots$, с которыми нам нужно конкурировать. Существует ли стратегия S со следующим свойством: для любой стратегии-конкурента S_i для некоторой константы c_i выполнено неравенство

$$\text{штраф}_S(x) \leqslant \text{штраф}_{S_i}(x) + c_i$$

для *всех* последовательностей x?

Оказывается, для многих функций ρ ответ на этот вопрос положительный. Более того, можно явно описать класс функций ρ, для которых это верно. Это было сделано в работах Вовка и его учеников. Более подробно о способах построения хороших стратегий предсказания можно прочитать в статье [11] и в монографии [5]. Мы ограничимся только одной конструкцией, изложенной в работе Р. Соломонова [10], работающей для функции штрафа $\rho(p, q) = -\log(1+pq)$ (общий штраф, следовательно, задаётся формулой (12.2)).

Теорема 12.3. *Пусть задано произвольное счётное семейство стратегий $S_1, S_2, \ldots$. Существует стратегия S такая, что для любого i найдется константа c_i, для которой*

$$\textit{штраф}_S(x) \leqslant \textit{штраф}_{S_i}(x) + c_i$$

для всех x.

Доказательство. Обозначим через $C_i(x)$ капитал игрока, полученный после исходов x при игре в соответствии со стратегией S_i. По теореме 12.1 (первая часть) функция $x \mapsto C_i(x)$ является мартингалом. Рассмотрим функцию $x \mapsto \sum_{i=1}^{\infty} 2^{-i} C_i(x)$. Нетрудно проверить, что она также является мартингалом. По второй части теоремы 12.1 найдётся стратегия S, для которой

$$C_S(x) = \sum_i 2^{-i} C_i(x)$$

для всех x. Эта стратегия и есть искомая. В самом деле, для любого i и всех x выполнено неравенство

$$C_S(x) \geqslant 2^{-i} C_i(x),$$

а значит

$$\text{штраф}_S(x) = -\log C_S(x) \leqslant i - \log C_i(x) = \text{штраф}_{S_i}(x) + i.$$

Теорема доказана.

Можно изложить доказательство в игровых терминах: стратегия S мысленно делит свой начальный капитал 1 между стратегиями $S_1, S_2, \ldots$ так, что i-ая стратегия получает капитал 2^{-i}. Затем стратегия S мысленно запускает все стратегии параллельно и вычисляет предсказание каждой из них. Выдаваемое ей предсказание есть взвешенная сумма всех вычисленных предсказаний, при этом мнение каждой стратегии учитывается с весом, пропорциональным текущему капиталу этой стратегии. Это довольно естественная линия поведения — если у Вас есть несколько советчиков, рекомендующих, куда вложить деньги, то мнение каждого из них естественно учитывать с весом, пропорциональным количеству уже заработанных им денег. □

В следующих задачах мы считаем, что казино генерирует исходы бросаний случайным образом, в соответствии с некоторым распределением вероятностей (необязательно равномерным).

Задача 84. Пусть казино делает n бросаний, используя некоторое распределение вероятностей $x \mapsto \mu(x)$ на строках длины n (известное игроку).

(а) Докажите, что у игрока есть стратегия со средним капиталом $2^{n-H_{\min}}$ в конце игры, где $H_{\min}$ обозначает минимальную энтропию распределения p: $H_{\min} = \min_x(-\log \mu(x))$.

(б) Докажите, что у игрока есть стратегия, средний логарифм капитала которой (в конце игры) равен $n - H$, где H обозначает энтропию Шеннона распределения μ.

До сих пор мы рассматривали случай, когда выигранная ставка удваивается. При такой выплате по теореме 12.2 средний капитал игрока равен начальному. Это можно интерпретировать следующим образом: действия казино являются «честными», если выбор исходов осуществляется с помощью симметричной монетки, гарантирующей равномерное распределение на исходах.

Представим теперь, что выбор исходов в казино осуществляется, с помощью некоторого (возможно, неравномерного) распределения $x \mapsto \mu(x)$ на словах длины n. Чтобы избежать чисто технических проблем, будем считать, что $\mu(x)$ положительно для всех x. При какой выплате на сделанную ставку игра останется честной в этом случае? Ответ: если после исходов $x = (x_1, \ldots, x_k)$ игрок сделал ставку -1 и угадал, то ему нужно возвратить ставку, умноженную на $\mu(x)/\mu(x(-1))$. Аналогично, если игрок сделал ставку на 1 и угадал, то ему должна возвращаться ставка с коэффициентом $\mu(x)/\mu(x1)$.

Правильность такой стратегии выплат подтверждается тем, что выполняются аналоги теорем 12.1 и 12.2.

Теорема 12.4. *Для любой стратегии S игрока функция $C_S(x) = C(x)$ принимает неотрицательные значения, причем*

$$C(\Lambda) = 1 \qquad \text{и} \qquad C(x) = \frac{\mu(x(-1))}{\mu(x)} C(x(-1)) + \frac{\mu(x1)}{\mu(x)} C(x1).$$

Обратно, для любой функции $C(x)$, удовлетворяющей этим условиям (такие функции называются мартингалами *относительно распределения μ) существует стратегия игры S, для которой $C = C_S$.*

Теорема 12.5. *Для любой стратегии S среднее значение функции $C_S(x) = C(x)$ на словах длины n по распределению μ равно 1. Обратно, для любой неотрицательной функции $C(x)$, определенной на словах длины n, среднее значение которой по распределению μ равно 1, существует стратегия игры S, для которой $C(x) = C_S(x)$ для всех строк длины n.*

Доказательство этих теорем совершенно аналогично доказательству теорем 12.1 и 12.2.

Задача 85. Пусть казино делает n бросаний, используя некоторое распределение вероятностей $x \mapsto \nu(x)$ на строках длины n (известное игроку). При этом казино производит выплаты так, как если бы оно использовало распределение μ (то есть, выигранная ставка на -1, сделанная после исходов x, увеличивается в $\mu(x)/\mu(x(-1))$ раз, а ставка на 1 — в $\mu(x)/\mu(x1)$ раз.

(а) Докажите, что у игрока есть стратегия со средним капиталом $\max_x\{\nu(x)/\mu(x)\}$ в конце игры.

(б) Докажите, что у игрока есть стратегия, средний логарифм капитала которой равен расстоянию Кульбака–Лейблера $\sum_x \nu(x) \log \frac{\nu(x)}{\mu(x)}$ между распределениями ν, μ.

Глава 13. Колмогоровская сложность

13.1 Что такое колмогоровская сложность?

«На пальцах» это проще всего объяснить так. Существуют программы, которые сжимают файлы для экономии места: наиболее распространённые называются `zip`, `gzip`, `compress`, `rar`, `arj`.

Применив такую программу к некоторому файлу (с текстом, данными, программой), мы получаем его сжатую версию (которая, как правило, короче исходного файла). По ней можно восстановить исходный файл (с помощью парной программы «декомпрессии»; часто сжатие и восстановление объединены в одну программу).

Так вот, в первом приближении колмогоровскую сложность файла можно описать как длину его сжатой версии. Тем самым файл с регулярной структурой и хорошо сжимаемый имеет малую колмогоровскую сложность (в сравнении с его длиной). Напротив, плохо сжимаемый файл имеет сложность, близкую к длине.

Это описание весьма приблизительно и нуждается в уточнениях — как технических, так и принципиальных. Начнём с технического: вместо файлов (последовательностей байтов) мы будем рассматривать двоичные слова. Более существенны другие отличия:

- Программы сжатия нет — мы рассматриваем лишь программу восстановления. Точнее, назовём *декомпрессором* произвольный алгоритм (программу), который получает на вход двоичные слова и выдаёт на выход также двоичные слова. Если декомпрессор D на входе x даёт y, мы пишем $D(x) = y$ и говорим, что x является *описанием* y при данном D (относительно данного D). Декомпрессоры мы также будем называть *способами описания*.
- Мы не требуем, чтобы декомпрессор был определён на всех словах. При некоторых x вычисление $D(x)$ может не останавливаться и не давать результата. Мы также не накладываем никаких огра-

ничений на время работы D: на некоторых входах программа D может дать ответ лишь после очень долгой работы.

Пусть фиксирован некоторый способ описания (декомпрессор) D. Для данного слова x рассмотрим все его описания, то есть все слова y, для которых $D(y)$ определено и равно x. Длину самого короткого из них и называют *колмогоровской сложностью* слова x при данном способе описания D:

$$KS_D(x) = \min\{\, l(y) \mid D(y) = x\}.$$

Здесь и далее $l(y)$ обозначает длину слова y. Индекс D подчёркивает, что определение зависит от выбора способа D. Минимум пустого множества, как обычно, считается равным $+\infty$, так что $KS_D(x)$ бесконечно для тех x, которые не входят в область значений функции D (не могут быть результатами декомпрессии, не имеют описаний).

Это определение кажется бессодержательным, поскольку для разных D получаются разные определения, в том числе абсурдные. Например, если D нигде не определён, то KS_D всегда бесконечно. Если $D(y) = \Lambda$ (пустое слово) при всех y, то сложность пустого слова равна нулю (поскольку $D(\Lambda) = \Lambda$ и $l(\Lambda) = 0$), а сложность всех остальных слов бесконечна.

Более осмысленный пример: если программа-декомпрессор копирует вход на выход и $D(x) = x$ при всех x, то каждое слово имеет единственное описание (самого себя) и $KS_D(x) = l(x)$.

Естественно, для любого слова x можно подобрать способ описания D, при котором x имеет малую сложность. Достаточно положить $D(\Lambda) = x$, и тогда $KS_D(x)$ будет равно нулю. Можно также подобрать способ описания, благоприятствующий целому классу слов: например, для слов из одних нулей хорош такой способ описания:

$$D(\mathrm{bin}(n)) = 000\ldots000 \quad (n \text{ нулей}),$$

где $\mathrm{bin}(n)$ — двоичная запись числа n. Легко проверить, что длина слова $\mathrm{bin}(n)$ примерно равна $\log_2 n$ (не превосходит $\log_2 n + 1$). Мы получаем, что для построенного способа описания сложность слова из n нулей близка к $\log_2 n$ (и много меньше длины слова, то есть n). Зато все другие слова (содержащие хотя бы одну единицу) имеют бесконечную сложность.

На первый взгляд кажется, что определение сложности настолько сильно зависит от выбора способа описания, что никакая общая теория невозможна.

13.2 Оптимальные способы описания

Способ описания тем лучше, чем короче описания. Поэтому естественно дать такое определение: способ D_1 *не хуже* способа D_2, если

$$KS_{D_1}(x) \leqslant KS_{D_2}(x) + c$$

при некотором c и при всех x.

В этом определении требует пояснения константа c. Мы соглашаемся пренебрегать увеличением сложности не более чем на константу. Конечно, можно сказать, что это лишает определение практического смысла, так как константа c может быть сколь угодно велика. Однако без такого «огрубления» обойтись не удаётся.

Пример. Пусть даны два произвольных способа описания D_1 и D_2. Покажем, что существует способ описания D, который не хуже их обоих.

В самом деле, положим

$$D(0y) = D_1(y),$$
$$D(1y) = D_2(y).$$

Другими словами, первый бит описания мы воспринимаем как номер способа описания, а остальную часть — как собственно описание. Ясно, что если y является описанием x при D_1 (или D_2), то $0y$ (соответственно $1y$) является описанием x при D, и это описание всего лишь на один бит длиннее. Поэтому

$$KS_D(x) \leqslant KS_{D_1}(x) + 1,$$
$$KS_D(x) \leqslant KS_{D_2}(x) + 1,$$

при всех x, так что способ D не хуже обоих способов D_1 и D_2.

Этот приём используется на практике. Например, формат `zip`-архива предусматривает заголовок, в котором указывается номер используемого метода сжатия, а затем идёт сам сжатый файл. При этом использование N различных способов описания приводит к тому, что начальные $\log_2 N$ битов (или около того) приходится зарезервировать для указания используемого способа.

Несколько обобщив эту же идею, можно доказать такую теорему:

Теорема 13.1 (Соломонова–Колмогорова)**.** *Существует способ описания D, который не хуже любого другого: для всякого способа описания D' найдётся такая константа c, что*

$$KS_D(x) \leqslant KS_{D'}(x) + c$$

для любого слова x.

Способ описания, обладающий указанным в теореме свойством, называют *оптимальным*.

Доказательство. Напомним, что по нашему определению способами описания являются вычислимые функции. Программы, их вычисляющие, можно считать двоичными словами. Будем при этом предполагать, что по программе можно определить, где она кончается (такой способ записи по-английски называется self–delimiting; подходящего русского слова, пожалуй, нет, и мы будем называть такие программы само-ограниченными). Если выбранный способ записи программ не таков, то его можно модифицировать. Например, можно продублировать каждый знак в записи программы (заменив 0 на 00 и 1 на 11), а в конце программы добавить группу 01.

Теперь определим новый способ описания D, положив

$$D(py) = p(y),$$

где p — произвольная программа (при выбранном способе записи), а y — любое двоичное слово. Другими словами, D читает слово слева направо, выделяя из него начало–программу. (Если таковой не оказывается, D делает что угодно, например, зацикливается.) Далее D применяет найденную программу (p) к остатку входа (y) и выдаёт полученный результат.

Покажем, что D не хуже любого другого способа описания P. Пусть p — программа, вычисляющая функцию P, причём записанная в выбранной нами форме. Если слово y является кратчайшим описанием слова x относительно P, то py будет описанием x относительно D (не обязательно кратчайшим). Поэтому при переходе от P к D длина описания увеличится не более чем на $l(p)$, и

$$KS_D(x) \leqslant KS_P(x) + l(p).$$

Видно, что константа (то есть $l(p)$) зависит лишь от выбора способа описания P, но не от x. □

По существу здесь используется тот же приём, что и в предыдущем примере, только вместо двух способов описания соединяются все мыслимые способы — каждый со своим префиксом. Именно так устроены так называемые «само-разархивирующиеся» архивы (self-extracting archives). Такой архив представляет собой исполняемый файл, в котором, однако, собственно программа занимает лишь небольшой начальный кусок. Эта программа загружается в память, после чего читает и разархивирует остаток архива.

Заметим, что построенный нами оптимальный способ описания на некоторых входах работает очень долго (поскольку бывают долго работающие программы), а на некоторых входах и вовсе не определён.

Фиксируем некоторый оптимальный способ описания D и будем называть *колмогоровской сложностью* слова x значение $KS_D(x)$. В обозначении $KS_D(x)$ будем опускать индекс D и писать просто $KS(x)$.

Замена оптимального способа на другой оптимальный способ приводит к изменению сложности не более чем на константу: для любых оптимальных способов D_1 и D_2 найдётся такая константа $c(D_1, D_2)$, что

$$|KS_{D_1}(x) - KS_{D_2}(x)| \leqslant c(D_1, D_2)$$

при всех x. Это неравенство записывают как

$$KS_{D_1}(x) = KS_{D_2}(x) + O(1),$$

понимая под $O(1)$ произвольную ограниченную функцию от x.

Имеет ли смысл говорить о колмогоровской сложности конкретного слова x согласно этому определению, не указывая, какой оптимальный способ мы используем? Нет — легко понять, что подбором подходящего оптимального способа можно сделать сложность данного слова любой. Точно так же нельзя говорить, что слово x проще слова y (имеет меньшую сложность): выбирая тот или иной оптимальный способ, можно сделать любое из двух слов более простым.

Каков же тогда смысл колмогоровской сложности, если ни про какое конкретное слово ничего нельзя сказать?

Свойства оптимального способа, построенного при доказательстве теоремы Соломонова–Колмогорова, зависят от использованного способа записи программ (то есть от выбора языка программирования). Два таких способа приводят к сложностям, отличающимся не более чем на

константу, которая есть длина интерпретатора одного языка программирования, написанного на другом языке. Можно надеяться, что при естественном выборе языков эта константа будет измеряться тысячами или даже сотнями. Тем самым, если мы говорим о сложностях порядка сотен тысяч (например, для текста романа) или миллионов (например, для ДНК), то уже не так важно, какой именно язык программирования мы выбрали.

Но, тем не менее, надо всегда помнить, что все утверждения о колмогоровской сложности носят асимптотический характер. Чтобы утверждение имело смысл, в нём должна идти речь о сложности не изолированного слова, а последовательности слов. Для теории сложности вычислений такого рода ситуация типична: например, оценки на сложность решения какой-либо задачи обычно также имеют асимптотический характер.

13.3 Сложность и случайность

Вернёмся к неравенству $KS(x) \leqslant l(x) + O(1)$ (теорема 13.5). Для большинства слов данной длины это неравенство близко к равенству. В самом деле, справедливо такое утверждение:

Теорема 13.2. *Пусть n — произвольное число. Тогда существует менее 2^n слов x, для которых $KS(x) < n$.*

Доказательство. Пусть D — оптимальный способ описания, фиксированный при определении колмогоровской сложности. Тогда все слова вида $D(y)$ при $l(y) < n$ (и только они) имеют сложность менее n. А таких слов не больше, чем различных слов y, имеющих длину меньше n, которых имеется

$$1 + 2 + 4 + 8 + \ldots + 2^{n-1} = 2^n - 1$$

штук (слов длины k ровно 2^k). □

Отсюда легко заключить, что доля слов сложности меньше $n - c$ среди всех слов длины n меньше $2^{n-c}/2^n = 2^{-c}$. Например, доля слов сложности менее 90 среди всех слов длины 100 меньше 2^{-10}.

Таким образом, большинство слов несжимаемы или почти несжимаемы. Это можно выразить и так: почти наверняка случайно взятое

слово данной длины окажется (почти) несжимаемым. Рассмотрим следующий мысленный (или даже реальный) эксперимент. Бросим монету, скажем, 80000 раз и сделаем из результатов бросаний файл длиной в 10000 байтов (8 битов образуют один байт). Можно смело держать пари, что ни один существовавший до момента бросания архиватор не сумеет сжать этот файл более чем на 10 байтов. (В самом деле, вероятность этого меньше 2^{-80} для каждого конкретного архиватора, а число различных архиваторов не так уж велико.)

Вообще естественно считать случайными несжимаемые слова: случайность есть отсутствие закономерностей, которые позволяют сжать слово. Конечно, чёткой границы между случайными и неслучайными объектами провести нельзя. Глупо интересоваться, какие именно из восьми слов длины 3 (то есть из слов $000, 001, 010, 011, 100, 101, 110, 111$) случайны, а какие нет. Другой пример: начав со «случайного» слова длиной 10000, будем заменять в нём по очереди единицы на нули. В конце получится явно неслучайное слово (из одних нулей), но имеет ли смысл спрашивать, в какой именно момент слово перестало быть случайным? Вряд ли.

Естественно определить *дефект случайности* двоичного слова x как разность $l(x) - KS(x)$. Используя эту терминологию, можно сказать так: теорема 13.5 утверждает, что дефект случайности почти что неотрицателен (не меньше некоторой константы), а теорема 13.2 гарантирует, что для случайно взятого слова данной длины n (мы считаем все слова длины n равновероятными) вероятность иметь дефект больше d оценивается сверху числом $1/2^d$.

Теперь закон больших чисел (утверждающий, что у большинства двоичных слов данной длины n доля единиц близка к $1/2$) можно попытаться перевести на язык колмогоровской сложности так: у любого слова с малым дефектом случайности частота единиц близка к $1/2$. Из этого «перевода» вытекает исходная формулировка закона (поскольку мы уже знаем, что слова с малым дефектом случайности составляют большинство); как мы впоследствии убедимся, в некотором смысле эти формулировки равносильны.

Если мы хотим провести чёткую границу между случайными и неслучайными объектами, мы должны от конечных объектов перейти к бесконечным. Определение случайности для бесконечных последовательностей нулей и единиц было дано учеником Колмогорова шведским математиком Мартин-Лёфом. Впоследствии Левин и Шнорр на-

шли критерий случайности в терминах сложности: последовательность случайна тогда и только тогда, когда дефект случайности её начальных отрезков ограничен. (Правда, нужно использовать другой вариант колмогоровской сложности, так называемую *монотонную* сложность.)

Задача 86. Докажите, что среднее арифметическое $KS(x)$ по всем словам длины n есть $n + O(1)$. [Указание. Доля слов x, для которых $KS(x) < n - l$, меньше 2^{-l}, и ряд $\sum l2^{-l}$ сходится. Это доказывает нижнюю оценку для среднего арифметического.]

13.4 Невычислимость KS и парадокс Берри

Прежде чем говорить о применениях колмогоровской сложности, скажем о препятствии, с которым сталкивается любое применение. Увы, функция KS невычислима: не существует алгоритма, который по данному слову x определяет его колмогоровскую сложность. Более того, не существует никакой нетривиальной (неограниченной) вычислимой нижней оценки для KS, как показывает следующая теорема.

Теорема 13.3. *Пусть k — произвольная (не обязательно всюду определённая) вычислимая функция из Ξ в $\mathbb{N}$. (Другими словами, вычисляющий её алгоритм применим к некоторым двоичным словам и даёт в качестве результатов натуральные числа.) Если k является нижней оценкой для колмогоровской сложности (то есть $k(x) \leqslant KS(x)$ для тех x, для которых $k(x)$ определено), то k ограничена: все её значения не превосходят некоторой константы.*

Доказательство. Доказательство этой теоремы повторяет так называемый «парадокс Берри». Этот парадокс состоит в предложении рассмотреть

> *наименьшее число, которое нельзя определить фразой из не более чем тринадцати русских слов.*

Эта фраза как раз содержит тринадцать слов и определяет то самое число, которое нельзя определить, так что получается противоречие.

Следуя этой идее, мы можем искать *первое попавшееся двоичное слово, колмогоровская сложность которого больше некоторого N.* С одной стороны, это слово по построению будет иметь сложность больше N, с другой стороны, приведённое его описание будет коротким (оно включает в себя число N, но двоичная запись числа N гораздо короче

самого́ N). Но как искать это слово? Для этого-то и нужна вычислимая нижняя оценка колмогоровской сложности.

Перейдём к формальному изложению доказательства. По условию функция k является вычислимой нижней оценкой колмогоровской сложности. Рассмотрим функцию $B(N)$, аргументом которой является натуральное число N, вычисляемую следующим алгоритмом: «Развернуть параллельно вычисления $k(0), k(1), k(2), \ldots$ и проводить их до тех пор, пока не обнаружится некоторое x, для которого $k(x) > N$. Первое из таких x и будет результатом».

Если функция k ограничена, то теорема доказана. В противном случае функция B определена для всех N, и $k(B(N)) > N$ по построению. По предположению k является нижней оценкой сложности, так что и $KS(B(N)) > N$. С другой стороны, $B(N)$ эффективно получается по двоичной записи $\mathrm{bin}(N)$ числа N, поэтому

$$KS(B(N)) \leqslant KS(\mathrm{bin}(N)) + O(1) \leqslant l(\mathrm{bin}(N)) + O(1) \leqslant \log_2 N + O(1)$$

(первое неравенство следует из теоремы 13.6, а второе — из теоремы 13.5; $O(1)$ обозначает ограниченное слагаемое). Получается, что

$$N < KS(B(N)) \leqslant \log_2 N + O(1),$$

что при больши́х N приводит к противоречию. □

13.5 Перечислимость сверху колмогоровской сложности

Множество натуральных чисел называется *перечислимым*, если существует алгоритм без входа, который печатает список всех элементов множества в некотором порядке, разделяя элементы, например, запятыми. Временны́е интервалы между печатью последовательных элементов могут быть сколь угодно большими: после печати очередного элемента мы не знаем, появится ли когда-нибудь следующий. Если множество бесконечно, то алгоритм, разумеется, не останавливается. Но он может не останавливаться даже в случае конечного множества.

Перечислимыми являются

- множество четных чисел: алгоритм делает счетное число шагов, на шаге i он печатает $2i$,

- множество простых чисел: на шаге i алгоритм проверяет, является ли i простым, и печатает его, если оно простое, и просто переходит к $i+1$ иначе,
- множество n, для которых в десятичной записи числа π найдется n троек подряд, окруженных не-тройками (алгоритм вычисляет по очереди все цифры десятичной записи π и, как только, находит очередной массив троек, окруженных не-тройками, печатает его длину.

Аналогично определяется перечислимость множества двоичных слов, множества пар натуральных чисел и так далее.

Пусть функция $f(x)$ определена на всех двоичных словах и принимает в качестве значений натуральные числа, а также специальный символ $+\infty$. Функция f называется *перечислимой сверху*, если множество

$$G_f = \{\langle x, n\rangle \mid f(x) < n\},$$

называемое иногда «надграфиком» функции f, перечислимо. Это объясняет несколько странное название «перечислимая сверху».

Теорема 13.4. (а) *Функция KS перечислима сверху, причём $|\{x \mid KS(x) < n\}| < 2^n$ при всех n.*

(б) *Если функция KS' перечислима сверху и $|\{x \mid KS'(x) < n\}| < 2^n$ при всех n, то найдётся такое c, что $KS(x) < KS'(x) + c$ для всех x.*

Доказательство. (а) Мы уже проводили оценку количества слов сложности меньше n, поэтому нам нужно доказать только перечислимость сверху. Зафиксируем оптимальный декомпрессор D из определения KS.

Алгоритм перечисления надграфика функции KS: Запустим D параллельно на всех входах: на n-ом этапе дадим D проработать n шагов на первых n двоичных словах. Когда в ходе этого процесса мы обнаружим, что $D(p) = x$ для некоторых p, x, ставим в очередь на печать пары $\langle x, n\rangle$ для всех $n > l(p)$. При этом оставляем в очереди счетное количество свободных мест для будущих пар. Кроме того, на каждом этапе выводим на печать первую пару из очереди.

(б) Пусть имеется функция KS' с указанными в теореме свойствами. Определим отображение $x \mapsto D'(x)$ следующим образом. Запустим перечислитель для надграфика KS'. При появлении очередной пары

$\langle x, n\rangle$ берем первое слово p длины n, на котором D' не было определено ранее, и определяем $D'(p) = x$. Неравенство $|\{x \mid KS'(x) < n\}| < 2^n$ гарантирует, что такое p обязательно найдется.

Определенное нами отображение вычисляется следующим алгоритмом: на входе p запускаем описанный выше процесс и дожидаемся, когда $D'(p)$ станет определенным (если этого никогда не произойдет, то алгоритм не завершит свою работу на входе p). Будем этот алгоритм обозначать той же буквой. По построению $KS_{D'}(x) = KS'(x) + 1$. Отсюда следует, что $KS(x) \leqslant KS'(x) + O(1)$. □

Теорему 13.4 можно воспринимать в качестве альтернативного определения колмогоровской сложности: $KS(x)$ есть наименьшая с точностью до постоянного слагаемого перечислимая сверху функция, удовлетворяющая неравенству $|\{x \mid KS(x) < n\}| < 2^n$ при всех n. Обычно это альтернативное определение используется в следующем виде. Пусть для некоторой перечислимой сверху функции $f(x)$ нам нужно выяснить, верно ли неравенство $KS(x) < f(x) + O(1)$ (то есть существует ли константа c такая, что для $KS(x) < f(x) + c$ для всех x). По теореме 13.4 это неравенство имеет место тогда и только тогда, когда $|\{x \mid f(x) < n\}| = O(2^n)$.

13.6 Сложность и информация

Колмогоровскую сложность слова x можно назвать также *количеством информации* в слове x. В самом деле, слово из одних нулей, которое может быть описано коротко, содержит мало информации, а какое-то сложное слово, которое не поддаётся сжатию, содержит много информации (пусть даже и бесполезной — мы не пытаемся отделить осмысленную информацию от бессмысленной, так что любая абракадабра для нас содержит много информации, если её нельзя задать коротко).

Если слово x имеет сложность k, мы говорим, что x содержит k битов информации. Естественно ожидать, что число битов информации в слове не превосходит его длины, то есть что $KS(x) \leqslant l(x)$. Так и оказывается (но только надо добавить константу, о чём мы уже говорили).

Теорема 13.5. *Существует такая константа c, что*

$$KS(x) \leqslant l(x) + c$$

для любого слова x.

Доказательство. Мы уже говорили, что если $P(y) = y$ при всех y, то $KS_P(x) = l(x)$. В силу оптимальности найдётся такое c, что

$$KS(x) \leqslant KS_P(x) + c = l(x) + c$$

при всех x. □

Утверждение этой теоремы обычно записывают так: $KS(x) \leqslant l(x) + O(1)$. Из него вытекает, в частности, что колмогоровская сложность любого слова конечна (то есть что любое слово имеет описание).

Вот ещё один пример свойства, которого естественно ожидать от понятия «количество информации»: при алгоритмических преобразованиях количество информации не увеличивается (точнее, увеличивается не более чем на константу, зависящую от алгоритма).

Теорема 13.6. *Для любого алгоритма A существует такая константа c, что*

$$KS(A(x)) \leqslant KS(x) + c$$

для всех x, при которых $A(x)$ определено.

Доказательство. Пусть D — оптимальный декомпрессор, используемый при определении колмогоровской сложности. Рассмотрим другой декомпрессор D':

$$D'(p) = A(D(p))$$

(D' применяет сначала D, а затем A). Если p является описанием некоторого x относительно D, причём $A(x)$ определено, то p является описанием $A(x)$ относительно D'. Взяв в качестве p кратчайшее описание x относительно D, находим, что

$$KS_{D'}(A(x)) \leqslant l(p) = KS_D(x) = KS(x),$$

а в силу оптимальности

$$KS(A(x)) \leqslant KS_{D'}(A(x)) + c \leqslant KS(x) + c$$

при некотором c и при всех x, что и требовалось доказать. □

Эта теорема гарантирует, что количество информации «не зависит от кодировки» — если мы, скажем, заменим в каком-то слове все нули на единицы и наоборот (или разбавим это слово нулями, добавив

после каждой цифры по нулю), то полученное слово будет иметь ту же сложность, что и исходное (с точностью до константы), поскольку преобразования в ту и другую сторону выполняются некоторым алгоритмом.

По этой причине колмогоровская сложность определена не только для двоичных слов, но и для других «конструктивных объектов». Например, сложность натурального числа n определяется, как $KS(n) = KS(f(n))$, где f — произвольная вычислимая инъективная функция, отображающая натуральные числа в двоичные слова (кодирование). По теореме 13.6 при смене f на другое $KS(n)$ изменится не более чем на константу. Аналогичным образом определяется сложность пары слов $KS(\langle x, y\rangle)$, как $KS([x, y])$ для некоторого кодирования $\langle x, y\rangle \mapsto [x, y]$, троек слов и так далее.

13.7 Сложность пары слов

Пусть x и y — два слова. Сколько битов информации содержится в паре, состоящей из x, y? Естественно ожидать, что количество информации в ней не превосходит суммы количеств информации в x и в y. И действительно это так, правда, с некоторой поправкой.

Теорема 13.7. *Существует такая константа c, что при любых x и y выполнено неравенство*

$$KS(\langle x, y\rangle) \leqslant KS(x) + 2\log KS(x) + KS(y) + c.$$

Доказательство. Попробуем для начала доказать теорему без добавочного члена $2\log KS(x)$, её ослабляющего. Это естественно делать следующим образом. Пусть D — оптимальный способ описания, используемый при определении колмогоровской сложности. Рассмотрим новый способ описания D'. Именно, если $D(p) = x$ и $D(q) = y$, будем считать pq описанием кода $[x, y]$ пары $\langle x, y\rangle$, то есть положим $D'(pq) = [x, y]$. Тогда сложность $[x, y]$ относительно D' не превосходит длины слова pq, то есть $l(p) + l(q)$. Если в качестве p и q взять кратчайшие описания, то получится $KS_{D'}([x, y]) \leqslant KS_D(x) + KS_D(y)$, и остаётся воспользоваться оптимальностью D и перейти от D' к D (при этом возникает константа c).

Что неверно в этом рассуждении? Дело в том, что определение D' некорректно: мы полагаем $D'(pq) = [D(p), D(q)]$, но D' не имеет средств

разделить p и q. Вполне может оказаться, что есть два разбиения слова на части, дающие разные результаты: $p_1q_1 = p_2q_2$, но $D(p_1) \neq D(p_2)$ или $D(q_1) \neq D(q_2)$.

Для исправления этой ошибки напишем перед словом pq длину слова p в двоичной записи. При этом в этой двоичной записи мы удвоим каждый бит, а после неё напишем 01, чтобы алгоритм мог понять, где кончается двоичная запись и начинается само p. Более точно, обозначим через $\mathrm{bin}(k)$ двоичную запись числа k, а через $\overline{x}$ результат удвоения каждого бита в слове x. (Например, $\mathrm{bin}(5) = 101$, а $\overline{\mathrm{bin}(5)} = 110011$.) Теперь положим

$$D'(\overline{\mathrm{bin}(l(p))}\,01pq) = D(p)D(q).$$

Это определение корректно: алгоритм D' сначала читает $\overline{\mathrm{bin}(l(p))}$, пока цифры идут парами. Когда появляется группа 01, он определяет $l(p)$, затем отсчитывает $l(p)$ цифр и получает p. Остаток входа есть q, после чего уже можно вычислить $[D(p), D(q)]$.

Величина $KS_{D'}(\langle x, y\rangle)$ оценивается теперь числом $2l(\mathrm{bin}(l(p))) + 2 + l(p) + l(q)$; двоичная запись числа $l(p)$ имеет длину не больше $\log_2 l(p)+1$, поэтому D'-описание слова xy имеет длину не более $2\log_2 l(p)+4+l(p)+l(q)$, откуда и вытекает утверждение теоремы. □

Заметим, что в теореме можно поменять местами p и q и доказать, что $KS(\langle x, y\rangle) \leqslant KS(x) + KS(y) + 2\log_2 KS(y) + c$. Будем в дальнейшем писать $KS(x, y)$ вместо $KS(\langle x, y\rangle)$. Константу 2 при логарифме в этой теореме можно понизить:

Теорема 13.8.

$$KS(x, y) \leqslant KS(x) + \log KS(x) + 2\log\log KS(x) + KS(y) + O(1);$$
$$KS(x, y) \leqslant KS(x) + \log KS(x) + \log\log KS(x) + {}$$
$${} + 2\log\log\log KS(x) + KS(y) + O(1);$$
$$\ldots$$

(Последовательность утверждений теоремы можно продолжать неограниченно. Кроме того, можно поменять местами x и y.)

Доказательство. Назовём вычислимое отображение $x \mapsto \widehat{x}$ множества двоичных слов в себя *беспрефиксным кодированием*, если ни для каких двух различных слов x и y слово $\widehat{x}$ не является началом слова $\widehat{y}$.

(Отсюда, в частности, следует, что $\widehat{x} \neq \widehat{y}$ при $x \neq y$.) Смысл этого определения таков: любое слово вида $\widehat{x}z$ можно однозначно разрезать на части и найти слова x и z.

Пример беспрефиксного кодирования: $x \mapsto \overline{x}01$, где $\overline{x}$ означает строку x с удвоенными битами. Здесь признаком окончания слова являются биты 01. Это кодирование не очень экономно (увеличивает длину вдвое). Более экономно такое кодирование:

$$x \mapsto \widehat{x} = \overline{\mathrm{bin}(l(x))}\,01x$$

(здесь $\mathrm{bin}(l(x))$ — двоичная запись длины слова x). Для него

$$l(\widehat{x}) = l(x) + 2\log l(x) + O(1).$$

Этот приём можно «итерировать»: начав с произвольного беспрефиксного кодирования $x \mapsto \widehat{x}$, можно построить новое (также беспрефиксное) кодирование

$$x \mapsto \widehat{\mathrm{bin}(l(x))}x.$$

В самом деле, если слово $\widehat{\mathrm{bin}(l(x))}x$ является началом слова $\widehat{\mathrm{bin}(l(y))}y$, то одно из слов $\widehat{\mathrm{bin}(l(x))}$ и $\widehat{\mathrm{bin}(l(y))}$ является началом другого, и потому $\mathrm{bin}(l(x)) = \mathrm{bin}(l(y))$. Отсюда мы заключаем, что x есть начало y, а потом — что $x = y$. (Другими словами, сначала мы однозначно декодируем длину слова, пользуясь тем, что она записана в беспрефиксном коде, а потом уже однозначно определяем само слово.)

Такая итерация даёт беспрефиксное кодирование, при котором

$$l(\widehat{x}) = l(x) + \log l(x) + 2\log\log l(x) + O(1),$$

затем

$$l(\widehat{x}) = l(x) + \log l(x) + \log\log l(x) + 2\log\log\log l(x) + O(1),$$

и так далее.

Вернёмся к доказательству теоремы. Пусть D — оптимальный способ описания, используемый при определении сложности. Рассмотрим способ описания D', задаваемый так:

$$D'(\widehat{p}q) = [D(p), D(q)],$$

где $\widehat{p}$ — беспрефиксный код слова p, а квадратные скобки означают кодирование пар, использованное при определении сложности пары. беспрефиксность кодирования $p \mapsto \widehat{p}$ гарантирует корректность этого определения ($\widehat{p}$ однозначно вычленяется из $\widehat{p}q$).

Если p и q — кратчайшие описания слов x и y, то слово $\widehat{p}q$ будет описанием слова $[x, y]$, и его длина как раз даёт оценку, указанную в теореме. (Чем более экономно беспрефиксное кодирование, тем лучше получается оценка; описанный выше итеративный метод построения беспрефиксных кодов даёт необходимые оценки.) □

Из теоремы 13.8 следует, что

$$KS(x, y) \leqslant KS(x) + KS(y) + O(\log n)$$

для слов x и y длины не более n. Как говорят, сложность пары не превосходит суммы сложностей её членов с точностью до логарифма (длин).

Возникает естественный вопрос: нельзя ли усилить оценку и доказать, что $KS(x, y) \leqslant KS(x) + KS(y) + O(1)$?

Следующее простое рассуждение показывает, что это невозможно. В самом деле, из этой оценки вытекало бы, что $KS(x, y) \leqslant l(x) + l(y) + O(1)$. Рассмотрим какое-то N и всевозможные $n = 0, 1, 2, \ldots, N$. Для каждого такого n имеется 2^n слов x длины n, а также 2^{N-n} слов y длины $N - n$. Комбинируя такие x и y, мы для данного n получаем 2^N пар $\langle x, y\rangle$, а всего (для всех n от 0 до N) получаем $(N+1)2^N$ пар. Если бы сложность всех этих пар $\langle x, y\rangle$ не превосходила $l(x) + l(y) + O(1) = N + O(1)$, то получилось бы $(N+1)2^N$ различных слов $[x, y]$, имеющих сложность не более $N + O(1)$, а таких слов, как мы знаем (теорема 13.2, с. 178), лишь $O(2^N)$.

Задача 87. Докажите, что не найдётся такого c, что

$$KS(x, y) \leqslant KS(x) + \log KS(x) + KS(y) + c$$

при всех x и y. [Указание. Замените в правой части неравенства сложности на длины и подсчитайте количество пар.]

Задача 88. Дайте естественное определение сложности тройки слов. Покажите, что $KS(x, y, z) \leqslant KS(x) + KS(y) + KS(z) + O(\log n)$, если слова x, y, z имеют длину не больше n.

Задача 89. (а) Покажите, что если $x \mapsto \widehat{x}$ — беспрефиксное кодирование, то

$$\sum_{x\in\Xi} 2^{-l(\widehat{x})} \leqslant 1$$

(здесь Ξ — множество всех двоичных слов).

(б) Покажите, что если беспрефиксное кодирование увеличивает длину слова не более чем на $f(n)$ (где n — исходная длина), то есть $l(\widehat{x}) \leqslant l(x) + f(l(x))$, то ряд $\sum_n 2^{-f(n)}$ сходится.

Насколько неравенство теоремы 13.7 близко к равенству? Может ли $KS(x,y)$ быть существенно меньше суммы $KS(x) + KS(y)$? В полном согласии с нашей интуицией это возможно, если x и y содержат много общего. Например, при $x = y$ мы имеем $KS(x,x) = KS(x) + O(1)$, поскольку $[x,x]$ алгоритмически получается из x и обратно (теорема 13.6).

Чтобы уточнить это наблюдение, мы определим понятие количества информации, которая содержится в x, но не содержится в y (для произвольных слов x и y). Эту величину называют *условной колмогоровской сложностью x при условии y* и обозначают через $KS(x|y)$ (говорят также «при известном y», «относительно y»). Её определение аналогично определению обычной (безусловной) сложности, но декомпрессор D имеет доступ не только к сжатому описанию, но и к слову y.

Если провести аналогию между энтропией Шеннона и колмогоровской сложностью, то $KS(x|y)$ аналогична условной энтропии $H(\xi|\eta)$ (а $KS(x)$ аналогична обычной безусловной энтропии $H(\xi)$).

13.8 Условная колмогоровская сложность

Посылая файл по электронной почте, можно сэкономить, если послать не сам файл, а его сжатый вариант (кратчайшее описание). Экономия может быть ещё больше, если получатель уже имеет старую версию файла — в таком случае достаточно описать внесённые изменения. Эти соображения приводят к следующему определению *условной сложности слова x при известном слове y*.

Назовём *способом условного описания* произвольную вычислимую функцию D двух аргументов (аргументы и значения функции D являются двоичными словами). Первый аргумент мы будем называть *описанием*, второй — *условием*. Если $D(y,z) = x$, мы говорим, что слово y является *описанием* слова x *при известном* z (ещё говорят «при

условии z», «относительно z»). Сложность $KS_D(x|z)$ определяется как длина кратчайшего описания слова x при известном слове z:

$$KS_D(x|z) = \min\{l(y) \mid D(y,z) = x\}.$$

Говорят, что способ (условного) описания D_1 *не хуже* способа D_2, если найдётся такая константа c, что

$$KS_{D_1}(x|z) \leqslant KS_{D_2}(x|z) + c$$

для любых слов x и z. Способ (условного) описания D называется *оптимальным*, если он не хуже любого другого способа (условного) описания.

Теорема 13.9. *Существует оптимальный способ условного описания.*

Доказательство. Этот вариант теоремы Колмогорова–Соломонова доказывается точно так же, как и безусловный (см. теорему 13.1, с. 176). А именно, фиксируем некоторый способ программирования для функций двух аргументов, при котором программы записываются в виде двоичных слов, и положим

$$D(\widehat{p}y, z) = p(y, z),$$

где $p(y,z)$ обозначает результат применения программы p ко входам y и z, а $\widehat{p}$ есть беспрефиксный код слова p. Теперь легко проверить, что если D' — произвольный способ условного описания, а p — соответствующая ему программа, то

$$KS_D(x|z) \leqslant KS_{D'}(x|z) + l(\widehat{p}).$$

□

Как и прежде, мы фиксируем некоторый оптимальный способ D условного описания и опускаем индекс D в KS_D. Соответствующую функцию мы называем *условной колмогоровской сложностью*; как и безусловная, она определена с точностью до ограниченного слагаемого.

Вот несколько простых свойств условной колмогоровской сложности:

Теорема 13.10.

$$KS(x|y) \leqslant KS(x) + O(1);$$
$$KS(x|x) = O(1);$$
$$KS(f(x,y)|y) \leqslant KS(x|y) + O(1);$$
$$KS(x|y) \leqslant KS(x|g(y)) + O(1).$$

Здесь g и f — произвольные вычислимые функции (одного и двух аргументов); имеется в виду, что указанные в теореме неравенства выполнены, если $f(x,y)$ (соответственно $g(y)$) определено.

Доказательство. Безусловный способ описания можно рассматривать и как условный (не зависящий от второго аргумента), отсюда получаем первое неравенство. Второе утверждение получается, если рассмотреть способ описания $D(p,z) = z$. Третье неравенство доказывается так: пусть D — оптимальный способ условного описания; рассмотрим другой способ

$$D'(p,y) = f(D(p,y),y)$$

и сравним его с оптимальным. Доказательство последнего неравенства аналогично, только нужно определить способ описания так:

$$D'(p,y) = D(p,g(y)).$$

□

Задача 90. Покажите, что условная сложность «непрерывна по второму аргументу»: $KS(x|y0) = KS(x|y) + O(1)$; $KS(x|y1) = KS(x|y) + O(1)$.

Задача 91. Покажите, что при фиксированном y функция $x \mapsto KS(x|y)$ отличается от KS не более чем на константу, зависящую от y (и не превосходящую $2KS(y) + O(1)$).

Задача 92. Докажите, что $KS([x,z]|[y,z]) \leqslant KS(x|y) + O(1)$ для любых x, y, z (квадратные скобки означают вычислимое кодирование пар).

Задача 93. Пусть фиксирован некоторый разумный язык программирования. (Формально говоря, нужно, чтобы соответствующая ему нумерация была главной, то есть чтобы была возможна эффективная трансляция программ с других языков [1].) Покажите, что условная сложность $KS(x|y)$ равна (с точностью до $O(1)$) минимальной сложности программы, которая даёт x на входе y. [Указание. Если D —

оптимальный способ условного описания, то сложность программы, которая получается из D фиксацией первого аргумента p, не превосходит $l(p) + O(1)$. С другой стороны, если программа p переводит y в x, то $KS(x|y) = KS(p(y)|y) \leqslant KS(p) + O(1)$.]

Многие свойства безусловной сложности легко переносятся и на условную. Вот некоторые из них (доказательства повторяют соответствующие рассуждения для безусловной сложности):

- Функция $KS(x|y)$ перечислима сверху (это означает, что множество троек $\langle x, y, n\rangle$, для которых $KS(x|y) < n$, перечислимо).
- При данных y и n множество тех слов x, для которых $KS(x|y) < n$, содержит менее 2^n элементов. Отсюда следствие:
- Для всякого y и для всякого n найдётся слово x длины n, сложность которого при известном y не меньше n.

Задача 94. Докажите, что для любых слов y и z и для любого n найдётся слово x длины n, у которого $KS(x|y) \geqslant n - 1$ и $KS(x|z) \geqslant n - 1$. [Указание: плохиех слов каждого типа менее половины.]

Теорема 13.11. *Если $\langle x, y\rangle \mapsto K(x|y)$ — произвольная перечислимая сверху функция, причём для любых y и n множество*

$$\{x \mid K(x|y) < n\}$$

содержит менее 2^n элементов, то $KS(x|y) \leqslant K(x|y) + c$ при некотором c и при любых x и y.

Доказательство этой теоремы почти дословно повторяет доказательство теоремы 13.4, поэтому мы его опускаем.

Понятие условной сложности позволяет придать смысл вопросу о том, сколько новой информации в ДНК одного организма по сравнению с ДНК другого: если d_1 — двоичное слово, кодирующее ДНК первого организма, а d_2 — двоичное слово, кодирующее ДНК второго, то искомая величина есть $KS(d_1|d_2)$. Аналогичным образом можно спрашивать, какой процент информации был потерян при переводе романа на другой язык: в этом случае нас интересует отношение

$$KS(\text{оригинал}|\text{перевод})/KS(\text{оригинал}).$$

Разность $KS(x) - KS(x|y)$ естественно назвать количеством информации об x в y: эта разность показывает, насколько знание слова y упрощает описание слова x.

В теореме о сложности пары (теорема 13.8, с. 186) также можно заменить сложность на условную:

Теорема 13.12.

$$KS(x, y) \leqslant KS(x) + 2\log KS(x) + KS(y|x) + O(1).$$

Доказательство. Пусть D_1 — оптимальный способ безусловного описания, а D_2 — оптимальный способ условного описания. Построим способ описания D', положив

$$D'(\widehat{p}q) = [D_1(p), D_2(q, D_1(p))].$$

Здесь $\widehat{p}$ — беспрефиксный код слова p, а квадратные скобки обозначают вычислимое кодирование пар, используемое при определении сложности пары. Если p — кратчайшее описание слова x, а q — кратчайшее описание слова y при известном x, то слово $\widehat{p}q$ будет описанием слова $[x, y]$ относительно D', откуда

$$KS(x, y) \leqslant KS_{D'}(x, y) + O(1) \leqslant l(\widehat{p}) + l(q) + O(1).$$

Осталось заметить, что можно выбрать беспрефиксное кодирование так, чтобы $l(\widehat{p})$ не превосходило $l(p) + 2\log l(p) + O(1)$ (см. доказательство теоремы 13.8 о сложности пары, с. 186). □

В этой теореме, как и раньше, можно заменить $2\log KS(x)$ на $\log KS(x) + 2\log\log KS(x)$ и так далее. Можно также добавочный член перенести на условную сложность, написав

$$KS(x, y) \leqslant KS(x) + KS(y|x) + 2\log KS(y|x) + O(1).$$

В доказательстве при этом следует заменить $D'(\widehat{p}q)$ на $D'(\widehat{q}p)$.

Задача 95. Докажите «неравенство треугольника»:

$$KS(x|z) \leqslant KS(x|y) + 2\log KS(x|y) + KS(y|z) + O(1)$$

для любых трёх слов x, y, z.

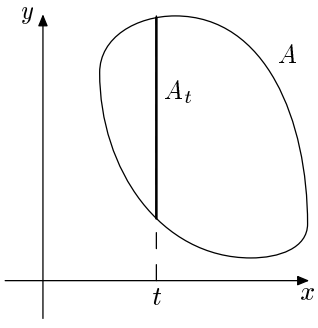

Рис. 3. Сечение A_t множества A простых пар.

Если не вдаваться в тонкости различных вариантов оценки добавочного члена с логарифмом, результат теоремы 13.12 можно сформулировать так: для слов x и y длины не более n имеет место неравенство

$$KS(x,y) \leqslant KS(x) + KS(y|x) + O(\log n).$$

Оказывается, что это неравенство на самом деле представляет собой равенство (с той же логарифмической точностью).

Теорема 13.13 (Колмогорова–Левина)**.**

$$KS(x,y) = KS(x) + KS(y|x) + O(\log n).$$

для слов x, y длины не больше n.

Доказательство. В одну сторону неравенство уже доказано. Осталось доказать, что $KS(x,y) \geqslant KS(x) + KS(y|x) + O(\log n)$, если x и y — слова длины не более n.

Пусть x, y — произвольные слова длины не более n. Обозначим сложность $KS(x,y)$ пары $\langle x,y\rangle$ через a. Пусть A — множество всех пар слов, у которых сложность не больше a. Число элементов в множестве A не больше $O(2^a)$ (точнее, меньше 2^{a+1}), и пара $\langle x,y\rangle$ — один из них.

Для каждого слова t рассмотрим сечение A_t множества A:

$$A_t = \{u \mid \langle t,u\rangle \in A\}$$

(рис. 3). Сумма мощностей множеств A_t при всех t равна мощности множества A, то есть не превосходит $O(2^a)$. Поэтому больши́х сечений у множества A немного, что мы сейчас и используем.

Пусть $m = \lfloor \log_2 |A_x| \rfloor$, где x — первая компонента исходной пары. Другими словами, пусть число элементов в A_x заключено между 2^m и 2^{m+1}. Докажем, что $KS(y|x)$ не сильно превосходит m, а $KS(x)$ не сильно превосходит $a - m$. Начнём с первого.

Зная a, можно перечислять множество A. Если мы к тому же знаем x, то можно оставлять от A только пары, у которых первая координата равна x, и получить перечисление множества A_x. Чтобы задать y, помимо a и условия x, достаточно указать порядковый номер элемента y в этом перечислении. Для этого достаточно $m + O(1)$ битов, и вместе с a получается $m + O(\log n)$ битов. Заметим, что $a = KS(x, y)$ не превосходит $O(n)$, если x и y — слова длины не более n, а потому для задания a достаточно $O(\log n)$ битов. Итак,

$$KS(y|x) \leqslant m + O(\log n).$$

Перейдём ко второй оценке. Множество B всех t, для которых $|A_t| \geqslant 2^m$, содержит не более $2^{a+1}/2^m$ элементов (иначе сумма $|A| = \sum |A_t|$ была бы больше 2^{a+1}). Множество B можно перечислять, зная a и m. (В самом деле, надо перечислять пары в множестве A; как только найдётся 2^m пар с одинаковой первой координатой, эта координата помещается в перечисление множества B.) Тем самым исходное слово x (как и любой другой элемент множества B) можно задать, указав $(a - m) + O(\log n)$ битов ($a - m$ битов уходят на порядковый номер слова x в B, а $O(\log n)$ битов позволяют дополнительно указать a и m). Отсюда

$$KS(x) \leqslant (a - m) + O(\log n),$$

и остаётся сложить два полученных неравенства. □

Доказанную только что теорему можно рассматривать как перевод на язык колмогоровской сложности такого комбинаторного факта. Пусть дано множество A пар слов. Его мощность не превосходит произведения мощности проекции на первую координату и мощности наибольшего сечения A_t (первая координата равна t, вторая произвольна). Это соответствует неравенству $KS(x, y) \leqslant KS(x) + KS(y|x) + O(\log n)$. Обратное неравенство интерпретируется сложнее. Пусть дано множество A пар слов, а также числа p и q, для которых $|A| \leqslant pq$. Тогда можно разбить A на две части P и Q с такими свойствами: проекция P на первую координату содержит не более p элементов, а все сечения Q_x

множества Q (первая координата равна x, вторая произвольна) содержат не более q элементов. (В самом деле, если отнести к P те сечения, которые содержат больше q элементов, то их число не превосходит p.)

Заметим, что на самом деле для доказательства важны не длины слов x и y, а их сложности. По существу мы доказали такой факт:

Теорема 13.14 (Колмогорова–Левина, вариант со сложностью)**.**

$$KS(x,y) = KS(x) + KS(y|x) + O(\log KS(x,y))$$

для любых слов x, y.

Задача 96. Оцените более точно константы в проведённом доказательстве и покажите, что

$$KS(x) + KS(y|x) \leqslant KS(x,y) + 3\log KS(x,y) + O(\log\log KS(x,y)).$$

Задача 97. Покажите, что в теореме Колмогорова– Левина члены порядка $O(\log n)$ неизбежны (причём в обе стороны): при любом n найдутся слова x и y длины не более n, для которых

$$KS(x,y) \geqslant KS(x) + KS(y|x) + \log n - O(1),$$

а также слова x и y длины не более n, для которых

$$KS(x,y) \leqslant KS(x) + KS(y|x) - \log n + O(1).$$

[Указание. В первом случае можно воспользоваться замечанием после теоремы 13.8 (с. 188). Во втором случае можно взять в качестве x число между $n/2$ и n, для которого $KS(x) = \log n + O(1)$, а затем в качестве y взять слово длины x, для которого $KS(y|x) = x + O(1)$.]

Задача 98. Покажите, что изменение одного бита в слове длины n меняет его сложность не более чем на $\log n + O(\log\log n)$.

Задача 99. Фиксируем некоторый (безусловный) способ описания D. Докажите, что для некоторой константы c и для всех n и k выполнено такое свойство: если какое-то слово x имеет 2^k описаний длины не более n, то $KS(x|k) \leqslant n-k+c$. [Указание. Пусть k фиксировано. Для каждого n рассмотрим слова x, имеющие не менее 2^k описаний длины не более n. Число таких слов (при данном k) не превосходит 2^{n-k}, и можно воспользоваться теоремой 13.11, с. 192.]

С помощью этой задачи можно доказать такое утверждение о безусловной сложности:

Задача 100. Пусть D — фиксированный (безусловный) способ описания. Тогда найдётся такое число c, что для любого слова x число кратчайших D-описаний слова x не превосходит c. [Указание. В условиях предыдущей задачи $KS(x) \leqslant n - k + 2\log k + O(1)$, поэтому при $KS(x) = n$ значение k ограничено.]

Задача 101. Докажите, что найдётся константа c с таким свойством: если для данных x и n вероятность события $KS(x|y) \leqslant k$ (для случайно взятого слова y длины n; все такие слова считаем равновероятными) не меньше 2^{-l}, то $KS(x|n,l) \leqslant k + l + c$. [Указание. Соединим каждое слово y длины n со всеми словами x, сложность которых относительно y не превосходит k, получим двудольный граф с $O(2^{n+k})$ рёбрами, и в нём число вершин x, из которых выходит не менее 2^{n-l} рёбер, есть $O(2^{k+l})$. Обратите внимание, что в $KS(x|n,l)$ не входит k — это не опечатка!]

Эта задача позволяет ответить на такой вопрос: чему в среднем равна сложность $KS(x|y)$ для данного x и случайно выбранного слова y данной длины n? Ясно, что $KS(x|y) \leqslant K(x|n) + O(1)$ (поскольку $n = l(y)$ восстанавливается по y). Оказывается, что для большинства слов y данной длины эта оценка точная:

Задача 102. Докажите, что найдётся такая константа c, что для любого слова x и для любых натуральных чисел n и d доля тех слов y длины n, у которых $KS(x|y) < KS(x|n) - d$ (среди всех слов длины n), не превосходит $cd^2/2^d$. Выведите отсюда, что среднее арифметическое $KS(x|y)$ по всем словам y данной длины n равно $KS(x|n) + O(1)$ (константа в $O(1)$ не зависит от x и n).

Задача 103. Докажите, что $KS(x) = KS(x|KS(x)) + O(1)$. [Указание. Пусть x имеет короткое описание q при известном $KS(x)$. Тогда для восстановления x достаточно задать q и разницу длин $KS(x) - l(q)$, и получается более короткое описание слова x, чем это возможно.]

13.8.1 Количество информации

Мы знаем (теорема 13.10), что условная сложность $KS(y|x)$ не превосходит безусловной сложности $KS(y)$ (с точностью до константы). Разность $KS(y) - KS(y|x)$ показывает, насколько знание слова x упрощает

описание слова y. Поэтому её называют *количеством информации в слове x о слове y*. Обозначение: $I(x:y)$.

Теорема 13.10 показывает, что информация $I(x:y)$ неотрицательна (с точностью до константы): существует такое c, что $I(x:y) \geqslant c$ при всех x и y.

Вспомнив, что

$$KS(x,y) = KS(x) + KS(y|x) + O(\log KS(x,y)),$$

(теорема 13.14, с. 196), можно выразить условную сложность через безусловные: $KS(y|x) = KS(x,y) - KS(x) + O(\log KS(x,y))$. Тогда для информации получается выражение:

$$I(x:y) = KS(y) - KS(y|x) = KS(x) + KS(y) - KS(x,y) + O(\log KS(x,y)).$$

Отсюда сразу вытекает

Теорема 13.15 (симметрия информации)**.**

$$I(x:y) = I(y:x) + O(\log KS(x,y))$$

Эта теорема гарантирует, что разница между $I(x:y)$ и $I(y:x)$ логарифмически мала по сравнению с $KS(x,y)$. Как показывает следующая задача, эта разница может быть сравнима с самими значениями $I(x:y)$ и $I(y:x)$, если те много меньше $KS(x,y)$.

Задача 104. Покажите, что если x — слово длины n, для которого $KS(x|n) \geqslant n$, то $I(x:n) = KS(n) + O(1)$, в то время как $I(n:x) = O(1)$.

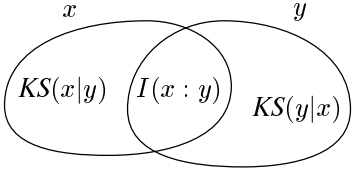

Рис. 4. Взаимная информация и условная сложность

Симметрия (пусть и не полная, а лишь с точностью до логарифмического слагаемого) позволяет называть $I(x:y)$ (или $I(y:x)$) *взаимной информацией* слов x и y. Соотношения между взаимной информацией, условными сложностями и сложностью пары можно изобразить на символической картинке (рис. 4).

На ней показано, что слова x и y имеют $I(x:y) \approx I(y:x)$ битов общей информации. Добавив $KS(x|y)$ битов (информация, которая есть в x, но не в y, левая область), мы получаем

$$I(y:x) + KS(x|y) \approx (KS(x) - KS(x|y)) + KS(x|y) \approx KS(x)$$

битов (как и должно быть в слове x). Аналогичным образом центральная часть вместе с $KS(y|x)$ справа дают $KS(y)$. Наконец, все три области вместе складываются в

$$KS(x|y) + I(x:y) + KS(y|x) = KS(x) + KS(y|x) = \\ = KS(x|y) + KS(y) = KS(x,y)$$

(все равенства верны с точностью $O(\log n)$, если слова x и y имеют длину не больше n).

13.9 Сложность и энтропия

Рассмотренные нами свойства шенноновской энтропии, условной энтропии и взаимной информации (для случайных величин) сходны с аналогичными утверждениями о колмогоровской сложности. Можно ли каким-либо образом уточнить и формализовать эти аналогии?

13.9.1 Колмогоровская сложность и энтропия частот

Будем рассматривать произвольный (не обязательно двухбуквенный) конечный алфавит A и слова в этом алфавите. Колмогоровская сложность для них определяется обычным образом.

Пусть фиксирован алфавит A, содержащий k букв. Рассмотрим произвольное слово x некоторой длины N в этом алфавите. Пусть $p_1, \ldots, p_k$ — частоты появления букв в слове x. Все они являются дробями со знаменателем N; сумма частот равна единице. Пусть $h(p_1, \ldots, p_k)$ — шенноновская энтропия соответствующего распределения.

Теорема 13.16.

$$KS(x) \leqslant h(p_1, \ldots, p_k) \cdot N + O(\log N).$$

Имеется в виду, что $O(\log N)$ не больше $c \log N$, причём константа c не зависит ни от N, ни от слова x, ни от частот $p_1, \ldots, p_k$, но может зависеть от k, так что в этой теореме размер алфавита считается фиксированным.

Доказательство. Применим к x кодирование на основе частот букв (с. 141). Длина полученного кода есть $Nh(p_1, \ldots, p_k) + O(\log N)$. Длина

задания N и частот $p_1, \ldots, p_n$ входят в $O(\log N)$. Поскольку для кодирования на основе частот букв кодирующая и декодирующая функции вычислимы, мы и получаем неравенство теоремы. □

Заметим, что неравенство теоремы 13.16 может быть как угодно далеко от равенства: если, скажем, алфавит состоит из двух букв, и в слове x они чередуются, то правая часть равна 1, а левая имеет порядок $(\log N)/N$. Что и не удивительно — правая часть учитывает только частоты, а не другие (не частотные) закономерности. В следующем разделе мы покажем, что при случайном порождении слова сложность близка к энтропии.

13.9.2 Математическое ожидание сложности

Пусть фиксирован k-буквенный алфавит A и k неотрицательных чисел $p_1, \ldots, p_k$ с суммой 1 (которые мы для простоты считаем рациональными).

Рассмотрим случайную величину ξ, значениями которой являются буквы алфавита A, принимаемые с вероятностями $p_1, \ldots, p_k$. Для каждого N рассмотрим случайную величину ξ^N, соответствующую N независимым копиям случайной величины ξ. Её значениями являются слова длины N в алфавите A (каждая буква порождается независимо, причём вероятность появления i-ой буквы равна p_i). Нас будет интересовать математическое ожидание сложности значений случайной величины ξ^N (взвешенное среднее с весами, равными вероятностям).

Теорема 13.17. *Математическое ожидание величины $KS(\xi^N)$ равно $NH(\xi) + O(\log N)$ (константа в $O(\log N)$ может зависеть от ξ, но не зависит от N).*

Заметим, что (при положительных p_i) среди значений величины ξ^N будут встречаться все слова длины N в алфавите A; некоторые из них имеют сложность много больше NH (если только распределение вероятностей не равномерное, поскольку в последнем случае таких слов нет), а другие имеют сложность много меньше NH.

Доказательство. Для каждого слова длины N (то есть для каждого значения случайной величины ξ^N) рассмотрим его кратчайшее описание (относительно оптимального способа описания). Полученные слова образуют инъективный код. Средняя длина этого кода как раз равна

математическому ожиданию сложности значений величины ξ^N. Поэтому по задаче 41 она не может быть меньше $H(\xi^N) - 2\log H(\xi^N) - 2$. Поскольку $H(\xi^N) = NH(\xi) = O(N)$, нижняя оценка доказана.

А теорема 10.2 позволяет получить верхнюю оценку. В самом деле, она утверждает существование префиксного кода, имеющего среднюю длину кодового слова не выше $H+1$. Такой код можно алгоритмически построить, зная число N (и числа p_i, предполагаемые фиксированными). Например, можно использовать конструкцию из доказательства теоремы 10.2, или применить код Хаффмана, или даже просто перебирать все коды, пока не найдётся подходящий.

Так или иначе построенный код можно рассматривать как некоторый способ условного описания (при известном N), для которого средняя префиксная сложность значения величины ξ^N не превосходит $H(\xi^N) + 1 = NH(\xi) + 1$. Переход к оптимальному способу описания увеличит среднюю сложность не более чем на константу. □

Теорема 13.17 показывает, что *в среднем* сложность равна энтропии, хотя она бывает и больше, и меньше её. На самом деле верно и более сильное утверждение: с почти единичной вероятностью сложность близка к энтропии, и вероятность того, что случайно выбранное значение величины ξ^N будет иметь сложность, сильно отличающуюся от энтропии, мала. Это утверждение является алгоритмическим аналогом теоремы Шеннона о пропускной способности канала без шума и следует из нее.

С одной стороны, в качестве декодирующей функции в теореме Шеннона можно рассмотреть оптимальный способ описания, а в качестве кодирующей функции — (невычислимый) компрессор, находящий кратчайшее описание. Поэтому для каждого $\varepsilon > 0$ существует c_ε такое, что вероятность события $KS(\xi^N) < NH(\xi) - c_\varepsilon\sqrt{N}$ меньше ε. С другой стороны, декодирующая функция, примененная в доказательстве теоремы Шеннона, вычислима, поэтому с вероятностью не менее $1-\varepsilon$ выполнено $KS(\xi^N|N, p_1, \ldots, p_k, \varepsilon) \leqslant NH(\xi) + c_\varepsilon\sqrt{N}$, следовательно, $KS(\xi^N) \leqslant NH(\xi) + c_\varepsilon\sqrt{N} + O(\log N)$.

13.10 Применения колмогоровской сложности

Прежде всего оговоримся: речь пойдёт не о практических применениях, а о связи колмогоровской сложности с другими вопросами.

13.10.1 Бесконечность множества простых чисел

Начнём с совсем «игрушечного» применения: докажем, что множество простых чисел бесконечно.

Пусть это не так и существует всего m различных простых чисел $p_1, \ldots, p_m$. Тогда любое целое число x, разлагаясь на простые множители, представляется в виде

$$x = p_1^{k_1} p_2^{k_2} \ldots p_m^{k_m}$$

и тем самым может быть задано набором степеней $k_1, \ldots, k_m$. Каждое из чисел k_i не превосходит $\log x$ (основания степеней не меньше 2) и потому имеет сложность не более $O(\log \log x)$ (двоичная запись содержит $O(\log \log x)$ битов). Поскольку m фиксировано, то сложность набора $\langle k_1, k_2, \ldots, k_m \rangle$ есть $O(\log \log x)$ и потому сложность любого числа x (которое алгоритмически получается из этого набора) есть $O(\log \log x)$. А для «случайного» n-битового числа x сложность примерно равна n, а не $O(\log n)$, как получается по этой формуле (логарифм n-битового числа не превосходит n).

Можно ли считать это «честным» применением колмогоровской сложности? Скептик скажет, что здесь всего лишь производится обычный подсчёт числа возможностей (counting argument, как говорят): если существует лишь m различных простых чисел, то различных чисел от 1 до x существует не более $(\log x)^m$, поскольку каждое такое число задаётся показателями степеней при простых множителях, и каждый показатель меньше $\log x$. После чего мы приходим к противоречию, поскольку $x > (\log x)^m$ при больших x.

Возразить на это нечего — именно это рассуждение и пересказано с помощью колмогоровской сложности (и стало немного более громоздким за счёт различных асимптотических обозначений). Тем не менее в подобном переводе может быть некоторый смысл, поскольку новый язык формирует новую интуицию, и она может быть полезна, даже если потом можно всё пересказать на старом языке.

13.10.2 Перенос информации по ленте

Следующий пример тоже модельный — мы докажем, что копирование слова длины n на одноленточной машине Тьюринга требует (в худшем случае) не менее εn^2 шагов. Этот классический результат был получен

в 1960-е годы с помощью так называемого «метода следов»; наше доказательство является переводом классического на язык колмогоровской сложности. (Мы предполагаем известными основные понятия, связанные с машинами Тьюринга; см., например, [1]).

Пусть на ленте машины Тьюринга выделена «буферная зона» некоторого размера b; нас интересует скорость переноса информации через эту буферную зону, скажем, слева направо, из области L в область R (рис. 5)

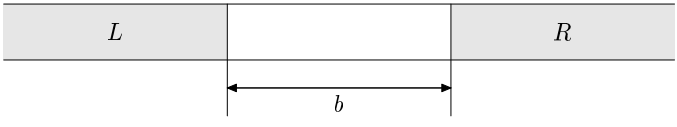

Рис. 5. Буферная зона размера b.

В некоторый момент ленты буферная область и R пусты, а область L содержит произвольную информацию. Нас интересует, какова может быть сложность слова R через t шагов после этого. Мы докажем, что она не больше $(t \log m)/b + O(\log t)$, где m — число состояний машины Тьюринга, а b — ширина буферной зоны. В самом деле, каждое из m состояний машины несёт $\log m$ битов информации, и за шаг информация переносится на одну клетку, а перенос её на b клеток требует в b раз больше времени.

Осталось лишь уточнить формулировку и доказательство.

Теорема 13.18. *Пусть фиксирована машина Тьюринга с m состояниями. Тогда существует такая константа c, что для любого b и для любого вычисления с буферной зоной b (вначале эта зона и лента справа от неё пусты, головка машины слева от зоны) сложность правой части ленты $R(t)$ после t шагов вычисления не превосходит*

$$\frac{t \log m}{b} + 4 \log t + c.$$

Доказательство. Проведём границу где-нибудь посередине буферной зоны, и при каждом пересечении границы головкой машины Тьюринга слева направо (при «выезде за границу») будем записывать, в каком состоянии она её пересекла. Записанная последовательность состояний

называется *следом* машины. Зная след, можно восстановить работу машины справа от границы (без использования информации о ленте слева от границы). В самом деле, надо искусственно поместить машину в первое состояние, указанное в следе, и выпустить её за границу; по возвращении перевести её во второе состояние следа и снова выпустить и так далее. При этом за границей машина будет вести себя так же, как и раньше (ведь ничего, кроме состояния, при пересечении границы у ней нет). В частности, в некоторый момент t' состояние ленты будет содержать $R(t)$. При этом t' не превосходит t, хотя может быть и меньше (поскольку время, проведённое машиной слева от границы, теперь не учитывается). Таким образом, можно алгоритмически восстановить $R(t)$, зная след, t', а также расстояние b' от границы до начала зоны R. Поэтому найдётся такая константа c (зависящая от машины, но не от b и t), что для любого следа S

$$KS(R(t)) \leqslant l(S)\log m + 4\log t + c$$

(мы умножили длину $l(S)$ следа S на $\log m$, поскольку след является словом в m-буквенном алфавите и на каждую букву приходится $\log m$ битов; приписывание к нему чисел b' и t' (в само-ограниченной записи) требует не более $2\log b + 2\log t$ битов, причём можно считать, что $t > b$ (иначе R пуста, головка туда не дошла); константа c возникает при переходе к оптимальному способу описания).

Это неравенство верно для любого начального содержимого части L и для любого положения границы: если теперь для данного L взять самый короткий из следов, то его длина меньше t/b (всего разных положений границы $b+1$, и в каждый момент пересечение происходит лишь в одном месте, так что сумма длин следов не больше t). Получаем требуемое неравенство. □

Задача 105. Покажите, что оценку теоремы можно улучшить, заменив b в знаменателе на $2b$. [Указание. Въезд суммарно потребует почти столько же шагов, сколько и выезд (не более чем на b меньше).]

Теперь сразу же получается квадратичная оценка для задачи копирования.

Пусть одноленточная машина M копирует любое входное слово: если вначале на ленте написано слово x из нулей и единиц, а дальше лента пуста, то в конце работы справа от x появляется его копия, и на ленте написано xx.

Теорема 13.19. *Существует такая константа $\varepsilon > 0$, что для любого n существует слово длины n, копирование которого с помощью M занимает не менее εn^2 шагов.*

Доказательство. Будем для простоты считать, что n чётно, и возьмём в качестве x слово, у которого вторая половина u имеет сложность не меньше длины (то есть $n/2$). Применим теорему о скорости переноса информации, считая буферной зоной участок длины $n/2$ справа от x (рис. 6).

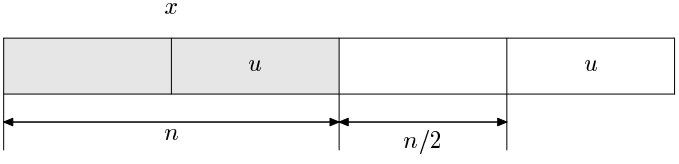

Рис. 6. Буферная зона при копировании.

Пусть копирование заняло t шагов, тогда сложность зоны R после этого не меньше $n/2$; с другой стороны, она не превосходит $t \log m/b + 4 \log t + c$ по доказанной теореме, где b — ширина буферной зоны, то есть $n/2$. Получаем, что

$$\frac{n}{2} \leqslant \frac{t \log m}{n/2} + 4 \log t + c.$$

Без ограничения общности считаем, что $t < n^2$ (иначе всё и так хорошо) и заменяем $4 \log t$ на $8 \log n$. Получаем, что

$$t \geqslant \frac{n^2}{4 \log m} - O(n \log n).$$

Второй член мал по сравнению с первым при больших n (и формально можно распространить неравенство на все n за счёт уменьшения коэффициента при n^2). □

Насколько существенна в этом доказательстве колмогоровская сложность? Скептик может вновь сказать, что по существу мы просто оценивали число слов, которые можно скопировать за данное время,

используя тот факт, что разные слова должны давать разные следы (иначе справа от границы поведение машины было бы одно и то же). Действительно, именно так и выглядело первоначальное доказательство этой оценки (уточним, что в первоначальных работах рассматривалось не копирование, а распознавание симметрии, но вся техника та же самая). Насколько идея этого доказательства становится понятнее при переводе его на язык колмогоровской сложности — вопрос вкуса.

Многие оценки в теории сложности вычислений могут быть доказаны аналогичным образом, когда в качестве «трудного случая» рассматривается слово максимальной сложности в некотором классе и затем показывается, что если бы оценка не соблюдалась, то это слово имело бы меньшую сложность. Много таких приложений (со ссылками на оригинальные работы) приведено в книге Ли и Витаньи [9], которые сами сыграли большую роль в развитии этого метода (называемого *Incompressibility method*). Отметим, что во многих случаях оценка впервые была доказана как раз с использованием колмогоровской сложности.

13.10.3 Бритва Оккама

Мы закончим следующим философским вопросом: что значит, что теория хорошо объясняет результаты эксперимента? Пусть имеется какой-то набор экспериментальных данных и разные теории, его объясняющие. Например, экспериментальными данными могут быть положения планет на небесной сфере. Их можно объяснять в духе Птолемея, рассматривая эпициклы и дифференты (и внося дополнительные поправки при необходимости), а можно ссылаться на законы современной механики. Как объяснить, чем современная теория лучше? Можно сказать так: современная теория позволяет вычислять положения планет с той же (и даже лучшей) точностью, имея меньше параметров. Условно говоря, достижение Кеплера состоит в том, что он нашёл более короткое описание для экспериментальных данных. Совсем грубо можно сказать, что экспериментаторы получают двоичные слова, после чего теоретики ищут для этих слов короткие описания (и тем самым верхние оценки на сложность), и лучше тот теоретик, у которого описание короче (оценка меньше).

Этот подход называют иногда «бритвой Оккама» по имени философа, который говорил, что не следует множить сущности без необхо-

димости. Насколько такое толкование одобрил бы сам Оккам, сказать трудно.

Можно использовать ту же идею в более практической ситуации. Представим себе, что мы собираемся автоматически читать надписи на конвертах и ищем правило, отделяющее изображения нулей от изображений единиц. (Будем считать, что изображение дано в виде матрицы битов, записанной как двоичное слово.) У нас есть несколько тысяч образцов — картинок, про которые мы знаем, нуль это или единица. По этим образцам мы хотим сформулировать какое-то разумное разделяющее правило. Что означает в этом контексте слово «разумное»? Если мы просто перечислим все образцы того или другого типа, получится вполне действующее правило — действующее до тех пор, пока нам не принесут новый образец, — но оно будет слишком длинным. Естественно считать, что разумное правило должно иметь короткое описание, то есть по возможности меньшую колмогоровскую сложность.

Дополнительные задачи

Задача 106. Сколькими способами можно упорядочить по весу 6 различных камешков?

Задача 107. Сколькими способами можно упорядочить 6 различных камешков так, чтобы первый камешек был больше второго?

Задача 108. Сколькими способами можно упорядочить 7 различных камешков так, чтобы первый камешек был больше второго, а второй больше третьего?

Задача 109. Сколькими способами можно упорядочить 7 различных камешков так, чтобы первый камешек был больше второго, а третий больше четвертого?

Задача 110. Дано шесть монет разного веса и известно, что первая монета легче второй, а вторая легче третьей. Имеются весы, позволяющие сравнить по весу любые две монеты. Доказать, что для того, чтобы упорядочить по весу все монеты, необходимо 7 взвешиваний.

Задача 111. Доказать, что, более того, в предыдущей задаче необходимо 8 взвешиваний и выполнить упорядочение за 8 взвешиваний.

Задача 112. Дано шесть монет разного веса и известно, что первая монета легче второй, третья легче четвертой, а четвертая легче пятой. Имеются весы, позволяющие сравнить по весу любые две монеты. Доказать, что для того, чтобы упорядочить по весу все монеты, необходимо 6 взвешиваний.

Задача 113. Доказать, что, более того, в предыдущей задаче необходимо 7 взвешиваний и выполнить упорядочение за 7 взвешиваний.

Задача 114. Имеются две карточки, на каждой из которых написано целое число от 1 до 9. Алёне сообщена сумма этих чисел, а Боре — разность первого и второго числа. Что написано на карточках, если в

этой ситуации Боря знает число на первой карточке? При каких числах на карточках Алёна знает, что Боря знает число на первой карточке?

Задача 115. Имеются две карточки, на каждой из которых написано целое число от 1 до 9. Алёне сообщена сумма этих чисел. При каких числах на карточках Алёне необходимо и достаточно задать два вопроса с ответами ДА/НЕТ, чтобы узнать оба числа?

Задача 116. У Алисы имеется последовательность x из n битов, а у Боба последовательность y из n битов. Они хотят вычислить $f(x,y) = \sum_{i=1}^{n}(x_i + y_i) \pmod 2$. Докажите, что для этого достаточно передать 2 бита, то есть коммуникационная сложность функции f не больше 2.

Задача 117. Докажите, что коммуникационная сложность функции f из предыдущей задачи в точности равна 2.

Задача 118. У Алисы имеется последовательность x из n битов, а у Боба последовательность y из n битов. Они хотят вычислить $f(x,y) = \sum_{i=1}^{n}(x_i + y_i)$. Докажите, что для этого достаточно передать $2\lceil \log_2 n \rceil$ бит.

Задача 119. Докажите, что коммуникационная сложность функции f из предыдущей задачи не меньше $\lceil \log_2 n \rceil$.

Задача 120. Пусть вероятности букв a,b,c,d,e,f,g равны, соответственно, 0.05, 0.1, 0.15, 0.15, 0.15, 0.2, 0.2. Найдите среднюю длину оптимального префиксного хода (построив код Хаффмана) и сравните ее с энтропией Шеннона.

Задача 121. Пусть вероятности букв a,b,c,d,e,f,g равны, соответственно, 0.05, 0.1, 0.1, 0.15, 0.2, 0.2, 0.2. Найдите среднюю длину оптимального префиксного хода (построив код Хаффмана) и сравните ее с энтропией Шеннона.

Задача 122. Пусть случайная величина α имеет распределение $p(0) = 1/9$, $p(1) = 1/6$, $p(2) = 1/6$, $p(3) = 5/9$. Положим $\beta = \alpha \pmod 2$. Найти $H(\alpha)$, $H(\beta)$, $I(\beta : \alpha)$, $H(\alpha|\beta)$, $H(\beta|\alpha)$, $H(\alpha, \beta)$.

Задача 123. Веса двух монеток выбираются случайно и независимо среди чисел $1, 2, 3, 4$. Какова энтропия Шеннона случайной величины, равной результату сравнения на чашечных весах весов первой и второй монетки? Какова энтропия случайной величины, равной разности весов первой и второй монетки? Какова взаимная информация этих случайных величин?

Задача 124. Доказать, что функция $I(\alpha : \beta) - I(\alpha : \beta|\gamma)$ является симметрической функцией случайных величин α, β, γ.

Задача 125. Алена выбирает число $x = 1, \ldots, n$ случайным образом так, что вероятность выбора числа i пропорциональна $1/i^2$ (коэффициент пропорциональности подобран так, чтобы сумма всех вероятностей была равна 1). Докажите, что Боря может, задав Алене в среднем $O(1)$ вопросов с ответами да/нет, узнать выбранное число.

Задача 126. Алена выбирает число $x = 1, \ldots, n$ случайным образом так, что вероятность выбора числа i пропорциональна $1/i$ (коэффициент пропорциональности подобран так, чтобы сумма всех вероятностей была равна 1). Докажите, что Боря может, задав Алене в среднем $(\log_2 n)/2 + o(\log n)$ вопросов с ответами да/нет, узнать выбранное число. Докажите, что любой алгоритм Бори в среднем задает не менее $(\log_2 n)/2 + o(\log n)$ вопросов.

Задача 127. Алена выбирает число $x = 1, \ldots, n$ случайным образом так, что вероятность выбора числа i пропорциональна $1/(i \ln i)$ (коэффициент пропорциональности подобран так, чтобы сумма всех вероятностей была равна 1). Докажите, что Боря может, задав Алене в среднем $o(\log n)$ вопросов с ответами да/нет, узнать выбранное число.

Задача 128. Пусть случайная величина α имеет распределение $1/3, 2/3$, а случайная величина β имеет распределение $1/2, 1/2$. В каких пределах могут изменяться $I(\beta : \alpha)$, $H(\alpha|\beta)$, $H(\beta|\alpha)$, $H(\alpha, \beta)$?

Задача 129. Алисе сообщено значение случайной величины α, а Бобу — значение некоторой функции f от α. Придумайте алгоритм, который позволит Алисе сообщить Бобу значение α, передав в среднем не более $H(\alpha|f(\alpha)) + 1$ битов.

Задача 130. Алисе сообщено значение случайной величины α, а Бобу — значение некоторой функции f от α. Докажите, что не существует алгоритма, который позволит Алисе сообщить Бобу значение α, передав в среднем менее $H(\alpha|f(\alpha))$ битов.

Задача 131. Доказать, что для любых случайных величин $\alpha_1, \ldots, \alpha_n$ выполнено следующее неравенство: $(n-1)H(\alpha_1, \ldots, \alpha_n)$ не превосходит суммы энтропий n кортежей, получающихся из кортежа $\langle \alpha_1, \ldots, \alpha_n \rangle$ вычеркиванием его компонент.

Задача 132. Доказать, что для любых случайных величин α, β, γ выполнено неравенство:

$$H(\alpha) \leqslant H(\alpha|\beta) + H(\alpha|\gamma) + I(\beta : \gamma).$$

Задача 133. Построить совместно распределённые случайные величины ξ, η, β, γ, для которых не выполнено неравенство

$$I(\xi : \eta) \leqslant I(\xi : \eta|\beta) + I(\xi : \eta|\gamma) + I(\beta : \gamma).$$

Задача 134. Пусть $\alpha_1, \alpha_2, \alpha_3, \ldots$ произвольные совместно распределенные случайные величины с одним и тем же конечным множеством исходов. Верно ли, что они образуют марковскую цепь тогда и только тогда, когда $I(\alpha_{i-1} : \alpha_{i+1}|\alpha_i) = 0$ для всех $i > 1$. (Последовательность случайных величин $\alpha_1, \alpha_2, \alpha_3, \ldots$ образует марковскую цепь, если для некоторой марковской цепи, α_i есть состояние этой цепи в момент i).

Задача 135. Алена выбирает число $x = 1, \ldots, n$ случайным образом по известному Боре распределению вероятностей. Известно, что Боря может, задав Алене в среднем q вопросов с ответами да/нет, узнать выбранное число. Докажите, что тогда некоторое число x выбирается Аленой с вероятностью не меньше 2^{-q}.

Задача 136. Приведите пример совместно распределенных случайных величин α, β, γ для которых неравенство

$$H(\alpha, \gamma) + H(\beta, \gamma) \leqslant H(\alpha, \beta, \gamma) + H(\gamma)$$

неверно.

Задача 137. Пусть дано линейное неравенство от $H(\alpha)$, $H(\beta)$, $H(\gamma)$, $H(\alpha, \gamma)$, $H(\beta, \gamma)$, которое истинно для всех троек α, β, γ таких, что $I(\alpha : \beta|\gamma) = 0$. Докажите, что тогда неравенство верно и для всех вообще троек совместно распределённых случайных величин α, β, γ.

Задача 138. Алёна выбирает число $x = 1, \ldots, n$ случайным образом. Задача Бори — задавая Алёне вопросы с ответами да/нет, узнать выбранное число. Боре разрешается ошибаться с вероятностью не более $1/2$. Постройте алгоритм Бори, который задает в худшем случае не более $\log n - 1$ вопросов (n — степень двойки). Докажите, любой алгоритм Бори задает в худшем случае не менее $\log n - 1$ вопросов.

Задача 139. Игрок в казино может поставить на любой из двух исходов (0,1) бросания симметричной монетки любую ставку в пределах его текущего капитала. Сделанная ставка удваивается в случае выигрыша и теряется иначе. Существует ли стратегия игрока, которая гарантированно утраивает начальный капитал после шести игр при условии, что в этих шести играх выпало чётное количество 1?

Задача 140. Игрок в казино может поставить на любой из двух исходов (0,1) бросания монетки любую ставку в пределах его текущего капитала. Сделанная ставка удваивается в случае выигрыша и теряется иначе. Пусть игроку стало известно, что в первых шести бросаниях будет ровно 3 единицы. Придумайте стратегию игрока, гарантирующую возрастание начального капитала в 3.2 раза. Какую ставку эта стратегия делает после того, как в первом бросании выпал 0? Какую ставку эта стратегия делает после того, как в первых двух бросаниях выпали нули? Докажите, что число 3.2 в этой задаче нельзя увеличить.

Задача 141. Игрок в казино может ставить на любой из шести исходов бросания кубика в пределах своего текущего капитала (можно ставить одновременно на несколько исходов различные ставки). Различные бросания кубика независимы и все грани равновероятны. При выпадении исхода i все сделанные ставки отбираются, но выплачивается выигрыш, в c_i раз больший ставки, сделанной на этот исход, где $c_1 = c_2 = c_3 = 4$, $c_4 = 8$, $c_5 = c_6 = 16$. Придумайте стратегию игрока, которая гарантирует с вероятностью не менее $7/8$ увеличение начального капитала в полтора раза в течение 3 игр.

Задача 142. Игрок в казино может ставить на любой из шести исходов бросания кубика в пределах своего текущего капитала (можно ставить сразу на несколько исходов различные ставки). Различные бросания кубика независимы и вероятности выпадения исходов $1, \ldots, 6$ равны $1/2, 1/4, 1/16, 1/16, 1/16, 1/16$ (они известны игроку). При выпадении любого исхода i все сделанные ставки отбираются, но выплачивается ставка, сделанная на этот исход, умноженная на 6. Придумайте стратегию игрока, которая гарантирует с вероятностью не менее $7/8$ увеличение капитала вдвое в течение трех игр.

Задача 143. (а) Найти пропускную способность следующего двоичного канала с шумом. Если входная буква равна 0, то на выходе канала появляются 0 и 1 с вероятностями $1/3$ и $2/3$, соответственно. Если входная буква равна 1, то на выходе канала появляются 0 и 1 с веро-

ятностями $1/6$ и $5/6$, соответственно. (б) Найти наименьшую возможную вероятность ошибки при передаче по этому каналу одного бита при кодировании его двумя буквами. (Вероятность ошибки при данной схеме кодирования/декодирования определяется, как максимум по всем исходным сообщениям вероятности неправильного декодирования данного сообщения.)

Задача 144. На множестве слов длины 4 в двухбуквенном алфавите 0,1 задана бернуллиева мера с вероятностями 0 и 1 равными $1/3$ и $2/3$, соответственно. Какова минимально возможная вероятность ошибки при кодировании элементов этого множества трех-битовыми последовательностями. (Вероятность ошибки при данной схеме кодирования/декодирования определяется, как максимум по всем исходным сообщениям вероятности неправильного декодирования данного сообщения.)

Задача 145. Постройте кодовые таблицы кодов Хаффмана, Шеннона–Фано и арифметического кода для последовательности `ebdecaefbcffcbcaddbaacbccedcffbb`.

Задача 146. Доказать, что колмогоровская сложность слова, составленного из n знаков после запятой числа $\sqrt{2}$, равна $KS(n) + O(1)$.

Задача 147. Зафиксируем некоторое вычислимое однозначное кодирование конечных множеств (двоичных) слов (двоичными) словами. Колмогоровской сложностью конечного множества слов назовем колмогоровскую сложность его кода. Положим $A_n = \{x : l(x) = n, KS(x) < n\}$. Докажите, что $KS(A_n) \leqslant n + O(\log n)$.

Задача 148. Зафиксируем некоторый оптимальный декомпрессор D. Для каждого слова x будем рассматривать кратчайшие описания x^* слова x (слово p называется описанием x, если $D(p) = x$). Докажите, что $KS(x^*) = KS(x, KS(x))$ для некоторого такого x^*.

Задача 149. Доказать, что для в любой строке длины n из нулей и единиц можно так изменить один бит, что колмогоровская сложность полученной строки станет не меньше $\log_2 n$.

Задача 150. Известно, что колмогоровская сложность неориентированного графа (без петель) с вершинами $1, 2, \ldots, n$ не меньше $n(n-1)/2$. (а) Докажите, что отсюда следует, что граф связен, если n достаточно велико. (б) Докажите, что отсюда следует, что граф содержит цикл,

если n достаточно велико. (в) Докажите, что отсюда следует, что входная и выходная степени любой вершины не меньше εn для некоторого $\varepsilon > 0$ и всех достаточно больших n.

Задача 151. Известно, что сложность строки x длины n из нулей и единиц не меньше n. (а) Докажите, что отсюда следует, что x содержит не меньше чем $n/2 - O(\sqrt{n})$ нулей. (б) Докажите, что для всех достаточно больших n в x не встречается $5\log_2 n$ нулей подряд. (в) Докажите, что для всех достаточно больших n в строке x встречается $(\log n)/2$ нулей подряд. (г) Докажите, что $KS(x|n) \geqslant n - O(1)$. (д) Сотрём в x все биты с нечётными номерами и обозначим полученное слово через x'. Докажите, что $KS(x') \geqslant n/2 - O(1)$.

Задача 152. Зафиксируем некоторый оптимальный декомпрессор D. Для каждого n рассмотрим самое «долгоиграющее» описание длины не более n, то есть такое p_n, для которого $D(p_n)$ определено и время вычисления $D(p_n)$ максимально (среди всех $p \in \mathrm{Dom}D$ длины не более n). Докажите, что $KS(p_n) \geqslant n - O(1)$.

Задача 153. Зафиксируем некоторый оптимальный декомпрессор D. Для каждого n рассмотрим максимальное время работы D на входах p длины не более n, на которых D останавливается. Обозначим полученное число через t_n. (а) Докажите, что функция $n \mapsto t_n$ растет быстрее любой вычислимой функции. (б) Докажите, что $KS(t_n) = n + O(\log n)$.

Задача 154. Пусть $m(n)$ обозначает минимальную колмогоровская сложность натуральных чисел в интервале $[n, +\infty)$. Докажите, что функция $m(n)$ невычислима.

Задача 155. Известно, что всюду определенная вычислимая функция f по любому натуральному n дает некоторое такое натуральное k, что колмогоровская сложность всех натуральных чисел $n, n+1, \ldots$ больше k. Докажите, что f ограничена сверху.

Задача 156. Известно, что колмогоровская сложность целого числа x в интервале от 1 до n не меньше $\lfloor \log_2 n \rfloor$. Докажите, что число x составное, если n достаточно велико.

Задача 157. Докажите, что не существует константы c, для которой для всех x, y выполнено $KS(x, y) \leqslant KS(x) + KS(y) + \log_2 KS(y) + c$.

Задача 158. Докажите, что количество слов длины n, для которых $KS(x|n) < KS(x) - c$, меньше $2^{n-c+O(1)}$.

Задача 159. Докажите неравенство $KS(x, y|x, z) \leqslant KS(y|z) + O(1)$. Докажите, что это неравенство является строгим (то, есть обратное неравенство $KS(x, y|x, z) \geqslant KS(y|z) - O(1)$ не является верным для всех x, y, z).

Задача 160. Пусть бесконечная последовательность $x_1, x_2, \ldots$ обладает следующим свойством: для всех n выполнено $KS(x_1 \ldots x_n|n) \leqslant c$. (а) Докажите, что количество таких последовательностей не превосходит $O(2^c)$. (б) Докажите, что то же самое верно и для последовательностей, удовлетворяющих более слабому неравенству $KS(x_1 \ldots x_n) \leqslant \log n + c$.

Задача 161. Докажите, что для любых x, y, n найдется слово z длины n, для которого $KS(z|x) \geqslant n - 1$ и $KS(z|y) \geqslant n - 1$.

Библиография

[1] Верещагин Н. К., Шень А. Лекции по математической логике и теории алгоритмов. Часть 3. Вычислимые функции. – М.: МЦНМО, 2008. (`ftp://ftp.mccme.ru/users/shen/logic/comput`)

[2] Успенский В. А., Верещагин Н. К., Шень А. Колмогоровская сложность. (Рукопись незаконченной книги.) (`ftp://ftp.mccme.ru/users/shen/kolmbook`)

[3] Чисар И., Кёрнер Я. Теория информации. – М.: «Мир», 1985. 397 с.

[4] Яглом А. М., Яглом И. М. Вероятность и информация. – М.: «Наука», 1973. 512 с.

[5] Nicolo Cesa-Bianchi and Gabor Lugosi. Prediction, Learning, and Games. Cambridge University Press, Cambridge, England, 2006.

[6] Cover M., Thomas J. A. Elements of information theory. – John Wiley & Sons, 1991.

[7] Galin L. Jones. On the Markov chain central limit theorem. // *Probab. Surveys* 1 (2004) 299-320.

[8] Kushilevitz E., Nisan N. Communication Complexity. – Camridge UP. 1997.

[9] Li M., Vitányi P. An Introduction to Kolmogorov Complexity and Its Applications, Second Edition. – Springer, 1997, 638 pp.

[10] Solomonoff R. J., A formal theory of inductive inference, part 1, part 2. *Information and Control* (now *Information and Computation*), v. 7 (1964), p. 1–22, p. 224–254.

[11] V.Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences* 56 (1998), p. 153-173.

Дополнение

Два приложения к книге М. М. Бонгарда «Проблема узнавания»

Приложение 1. Гипотезы, содержащие только истину, и оптимальная гипотеза

Мы часто описываем различные ситуации словами: «правда, но не вся правда», «не только правда» и т. п. Каким формальным соотношением соответствуют эти интуитивные представления? Начнем с простейшего случая.

Предположим, некто Z должен найти, на какой странице книги, содержащей 100 страниц, имеется ссылка на работу интересующего его автора. Пусть в действительности эта ссылка находится на 45 странице. Допустим, по мнению Z, ссылка находится в первой половине книги. Очевидно, в этом случае мы скажем, что гипотеза Z — истинна. Правда, гипотеза эта не очень сильная. Более сильной мы бы признали, например, гипотезу, что ссылка находится между 40 и 50 стр. Гипотеза о том, что ссылка находится между 47 и 50 стр., будет еще более сильной, но уже ложной.

Во всех рассмотренных примерах было очень легко определить, является ли гипотеза истинной или ложной. Нужно было только проверить, попадает ли ответ задачи (номер страницы) в область, ограниченную гипотезой.

Существуют, однако, более сложные ситуации. Допустим, книга печаталась двумя изданиями с тиражами 10 000 и 30 000 экземпляров. В первом издании ссылка попала на 45 стр., а во втором — на 48. Мы не знаем, к какому изданию относится имеющийся у нас экземпляр.

Что можно сказать про гипотезу о том, что ссылка с вероятностью 0.3 находится между 30 и 46 стр. и с вероятностью 0.7 — между 47 и 50 стр.? По-видимому, про нее не имеет смысла говорить, что она истинна или ложна. Для описания соотношения между гипотезой и задачей в подобном случае необходимо какое-то новое понятие. При этом хочется, чтобы в простых ситуациях новое понятие не противоречило бы обычной истинности или ложности.

Обратим внимание, что неопределенность задачи при некоторой гипотезе сама по себе не является мерилом истинности гипотезы. Вернемся к примеру с книгой. Рассмотрим две гипотезы:

а) ссылка находится между 35 и 50 стр., причем вероятность найти ее между 35 и 42 стр. больше, чем между 43 и 50 стр.;
б) ссылка находится между 1 и 100 страницами.

Неопределенность задачи при гипотезе а) может быть меньше, чем при гипотезе б)[6]. В то же время мы интуитивно чувствуем, что гипотеза а) «содержит не только истину». А гипотезу б) мы признаем истинной, хотя она и приводит к большей неопределенности задачи. Почему? По-видимому, при оценке истинности для нас существенно не только то, что дает гипотеза, но и то, что она «обещает» (сила гипотезы). Гипотеза б) мало дает, но она ведь ничего и не обещает. Поэтому не возникает ощущения, что нас обманули.

Мы употребили термин «гипотеза обещает». А как измерить силу гипотезы (много или мало она обещает)? В простейшем примере, с которого начался этот параграф, гипотеза была тем сильнее, чем больше она ограничивала область поисков. В случае, когда гипотеза заключается в распределении вероятностей $q_1, q_2, \ldots, q_n$ того, что объект принадлежит $1, 2, \ldots, n$ классу, естественно считать мерой силы гипотезы энтропию этого распределения вероятностей $H(\mathbf{q})$ [7]. В соответствии с этим мы будем считать, что гипотеза $\mathbf{q}'$ сильнее, чем гипотеза $\mathbf{q}''$, если $H(\mathbf{q}') < H(\mathbf{q}'')$.

Теперь появилась возможность сравнить то, что гипотеза обещает, с тем, что она реально дает. Сделаем это для более общего случая, когда решающий алгоритм имеет дело не с отдельной задачей, а с некоторой

[6] Если по гипотезе а) вероятность области 35-42 стр. не намного больше, чем области 43-50 стр.

[7] Легко заметить, что гипотеза $\mathbf{q}$ обещает нам именно неопределенность, равную $H(\mathbf{q})$. В самом деле, если гипотеза абсолютно точна, то $p_i = q_i$ $(i = 1, 2, \ldots, n)$, а значит, $N(\mathbf{p}|\mathbf{q}) = H(\mathbf{q})$.

совокупностью задач. В этом случае имеется не какое-то определенное распределение вероятностей ответов $\mathbf{p}$, а лишь некоторые соотношения типа:

$$\left.\begin{array}{c} f_1(p_1, p_2, \ldots, p_n) \geqslant 0, \\ f_2(p_1, p_2, \ldots, p_n) \geqslant 0, \\ \ldots \\ f_n(p_1, p_2, \ldots, p_n) \geqslant 0, \end{array}\right\} \tag{1}$$

ограничивающие область возможных значений $\mathbf{p}$. К системе ограничений, конечно, всегда добавляются соотношения

$$p_i \geqslant 0 \quad \text{и} \quad \sum_{i=1}^{n} p_i = 1.$$

Будем считать, что система ограничений такова, что существует по крайней мере одно распределение $\mathbf{p}$, удовлетворяющее всем неравенствам (совокупность содержит хоть одну задачу).

Назовем распределение вероятностей $\tilde{\mathbf{q}}$ *гипотезой, содержащей только истину относительно ограничений 1*, если

$$\max_{\mathbf{p}} N(\mathbf{p}|\tilde{\mathbf{q}}) \leqslant H(\tilde{\mathbf{q}}), \tag{2}$$

где $\max_{\mathbf{p}}$ есть точная верхняя граница, взятая по всем распределениям $\mathbf{p}$, удовлетворяющим ограничениям 1.

Другими словами, мы будем называть гипотезу *содержащей только истину*, если для всех задач из совокупности она дает не большую неопределенность, чем обещает.

Гипотезы, не удовлетворяющие соотношению 2, будут называться *содержащими не только истину*. Заметим, что при любой системе ограничений существует по крайней мере одна содержащая только истину гипотеза $\tilde{\mathbf{q}}_i = \frac{1}{n}$ $(i = 1, 2, \ldots, n)$. Действительно, в этом случае

$$N(\mathbf{p}|\tilde{\mathbf{q}}) = \log n = H(\tilde{\mathbf{q}}).$$

Назовем $\tilde{\mathbf{q}}^0$ *сильнейшей содержащей только истину гипотезой*, если $\tilde{\mathbf{q}}^0$ — гипотеза, содержащая только истину, и имеет место соотношение

$$H(\tilde{\mathbf{q}}^0) \leqslant H(\tilde{\mathbf{q}}), \tag{3}$$

где $\tilde{\mathbf{q}}$ — произвольная содержащая только истину гипотеза.

Можно показать[8], что множество всех гипотез, содержащих только истину, является замкнутым, и так как $H(\tilde{\mathbf{q}}) \geqslant 0$, то $H(\tilde{\mathbf{q}})$ достигает на этом множестве точной нижней границы, а следовательно, сильнейшая содержащая только истину гипотеза всегда существует.

Если решающий алгоритм без обратной связи имеет дело с отдельной задачей, то наилучшей гипотезой для него будет $\mathbf{q} = \mathbf{p}$[9]. Обобщим понятие наилучшей гипотезы для случая, когда имеется совокупность задач (действуют ограничения 1).

Назовем распределение вероятностей $\mathbf{q}^0$ оптимальной гипотезой относительно ограничений 1, если

$$\max_{\mathbf{p}} N(\mathbf{p}|\mathbf{q}^0) \leqslant \max_{\mathbf{p}} N(\mathbf{p}|\mathbf{q}), \tag{4}$$

где знак $\max_{\mathbf{p}}$ употребляется в том же смысле, что и в 2, а $\mathbf{q}$ есть произвольная гипотеза.

Легко убедиться, что в случае, когда имеется единственная задача (определенное $\mathbf{p}$), оптимальная гипотеза является гипотезой, содержащей только истину. Возникает вопрос: не нарушается ли это соотношение в более сложных случаях? Не имеет ли смысла, при наличии совокупности задач, искать оптимальную гипотезу также и среди гипотез, содержащих не только истину? Ответ на этот вопрос дает следующая теорема:

Теорема 1. *Для любой системы ограничений, наложенных на* $\mathbf{p}$*, существует единственная оптимальная гипотеза* $\mathbf{q}^0$*. Оптимальная гипотеза* $\mathbf{q}^0$ *совпадает с сильнейшей содержащей только истину гипотезой* $\tilde{\mathbf{q}}^0$*. При этом*

$$\max_{\mathbf{p}} N(\mathbf{p}|\mathbf{q}^0) = H(\mathbf{q}^0).$$

Доказательство. Для доказательства нам будет удобно обратиться к геометрической интерпретации понятий: «распределение вероятностей ответов задачи», «гипотеза», «энтропия» и «неопределенность».

Начнем со случая $n = 2$. Благодаря связи $q_1 + q_2 = 1$ многообразие всех возможных гипотез является одномерным. Будем изображать

[8] Для краткости мы не приводим доказательства.

[9] М. М. Бонгард, «Проблема узнавания», глава VII, параграф 3

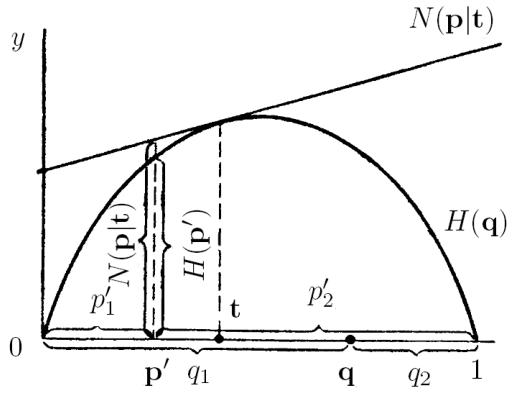

Рис. 1. Геометрическая иллюстрация понятия «неопределенность»

гипотезу точкой $\mathbf{q}$ на отрезке $[0;\ 1]$ оси абсцисс (рис. 1). На этом же отрезке находятся точки $\mathbf{p}$. Для отыскания неопределенности задачи, характеризуемой $\mathbf{p}'$, при гипотезе $\mathbf{q} = \mathbf{t}$ строим кривую $y = H(\mathbf{q})$ и проводим прямую, касательную к ней в точке, соответствующей $\mathbf{t}$. Касательная будет графиком функции $y = N(\mathbf{p}|\mathbf{t})$.

В самом деле, $N(\mathbf{p}|\mathbf{t}) = -p_1 \cdot \log t_1 - p_2 \log t_2$ есть линейная функция от $\mathbf{p}$, $N(\mathbf{t}|\mathbf{t}) = H(\mathbf{t})$ и $N(\mathbf{p}'|\mathbf{t}) > H(\mathbf{p}')$ при $\mathbf{p} \neq \mathbf{t}$. Таким образом, $y = N(\mathbf{p}|\mathbf{t})$ должно быть прямой, имеющей с $y = H(\mathbf{q})$ общую точку при $\mathbf{p} = \mathbf{t}$ и лежащей выше $y = H(\mathbf{q})$ при $\mathbf{p} \neq \mathbf{t}$. Так как, кроме того, $y = H(\mathbf{q})$ выпукла, непрерывна и имеет непрерывную производную всюду, кроме точек $q_1 = 0$ и $q_1 = 1$, то $y = N(\mathbf{p}|t)$ есть касательная к $y = H(\mathbf{q})$.

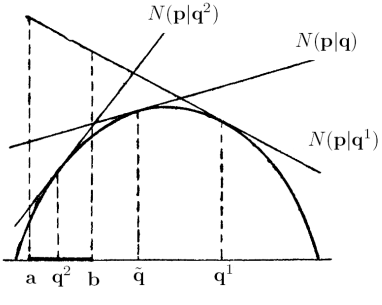

Рис. 2.

Пусть ограничения, наложенные на $\mathbf{p}$, например, таковы, что $\mathbf{p}$ может находиться только на отрезке $[\mathbf{a}, \mathbf{b}]$ (рис. 2). Тогда из изображенных на этом рисунке гипотез гипотеза $\tilde{\mathbf{q}}$ содержит только истину относительно этих ограничений, а $\mathbf{q}_1$ и $\mathbf{q}_2$ — гипотезы, содержащие не

только истину, так как $\max_{\mathbf{p}} N(\mathbf{p}|\mathbf{q}^1) > H(\mathbf{q}^1)$ и $\max_{\mathbf{p}} N(\mathbf{p}|\mathbf{q}^2) > H(\mathbf{q}^2)$. Очевидно, что в этом примере 0 совпадает с точкой $\mathbf{b}$.

Если n произвольно, то $\mathbf{p}$ и $\mathbf{q}$ являются точками $(n-1)$-мерного тетраэдра (симплекса). Обозначим $(n-1)$-мерную гиперплоскость, содержащую этот тетраэдр, через K. В n-мерном случае $y = H(\mathbf{q})$ есть $(n-1)$-мерная выпуклая поверхность. Проведем $(n-1)$-мерную гиперплоскость, касательную к поверхности $y = H(\mathbf{q})$ в точке, соответствующей $\mathbf{q} = \mathbf{t}$. Уравнение этой гиперплоскости будет $y = N(\mathbf{p}|\mathbf{t})$. Посмотрим, какими свойствами она обладает. Образуем $(n-2)$-мерное пересечение гиперплоскостей $y = N(\mathbf{p}|\mathbf{t})$ и $y = H(\mathbf{t}) = \text{const}$. Спроектируем полученное пересечение на K; эта проекция F также является $(n-2)$-мерной плоскостью. Если $\mathbf{p} \in F$, то $N(\mathbf{p}|\mathbf{t}) = H(\mathbf{t})$. F делит K на две части. Для точек $\mathbf{p}$, находящихся по одну сторону F, имеет место соотношение $N(\mathbf{p}|\mathbf{t}) < H(\mathbf{t})$, для находящихся по другую сторону — соотношение $N(\mathbf{p}|\mathbf{t}) > H(\mathbf{t})$. Точка $\mathbf{h}$ с координатами $h_i = \frac{1}{n}$ $(i = 1, 2, \ldots, n)$ всегда попадает в область больших неопределенностей. Действительно, $N(\mathbf{h}|\mathbf{t}) \geqslant H(\mathbf{h}) \geqslant H(\mathbf{t})$. При $\mathbf{t} \neq \mathbf{h}$ имеет место строгое неравенство. При $\mathbf{t} = \mathbf{h}$ плоскость $y = N(\mathbf{p}|\mathbf{t})$ совпадает с $y = H(\mathbf{t})$ и F заполняет весь тетраэдр.

Спроектируем теперь на K область, образованную пересечением поверхности $y = H(\mathbf{q})$ и плоскости $y = H(\mathbf{t})$. Обозначим эту проекцию через G. Очевидно, $\mathbf{t} \in F$, $\mathbf{t} \in G$ и F касается G в точке t.

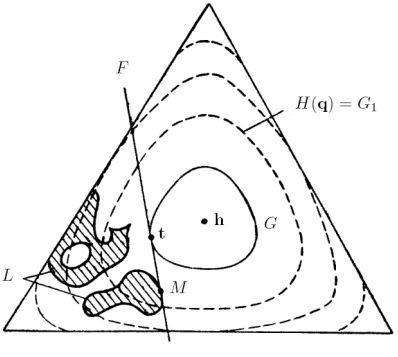

Рис. 3.

Обозначим через $\overline{L}$ множество всех точек тетраэдра, удовлетворяющих ограничениям 1, а через L замкнутое множество, содержащее $\overline{L}$ и его предельные точки. Пересечение L и F обозначим через M.

На рис. 3 изображена плоскость K для случая $n = 3$.

Для того чтобы узнать, является ли $\mathbf{t}$ гипотезой, содержащей только истину относительно ограничений 1 (или, что то же самое, относительно области L), нужно провести через $\mathbf{t}$ поверхность постоянной энтропии G и построить касательную к ней в точке $\mathbf{t}$ гиперплоскость F. Если F

отделяет[10] L от $\mathbf{h}$, то $\mathbf{t}$ есть гипотеза, содержащая только истину (при этом, очевидно, F отделяет L от G).

Отсюда следствие: если область L такова, что не существует $(n-2)$-мерной гиперплоскости, отделяющей ее от $\mathbf{h}$, то единственной содержащей только истину гипотезой является $\tilde{\mathbf{q}} = \mathbf{h}$. Пример такого случая показан на рис. 4.

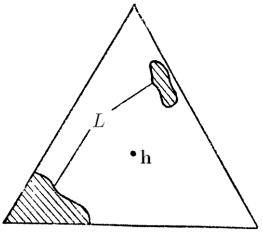

Рис. 4.

Теперь мы можем сформулировать следующую лемму.

Лемма 1. *Если* $\tilde{\mathbf{q}}^0$ *— сильнейшая содержащая только истину гипотеза, то найдутся такие точки* $\mathbf{m}^1; \mathbf{m}^2; \ldots; \mathbf{m}^j; \ldots; \mathbf{m}^s$, $\mathbf{m}^j \in M (s \leqslant n-1)$, *и такие числа* $\alpha^j > 0$, *что* $\sum\limits_{j=1}^{s} \alpha^j = 1$ *и* $\sum\limits_{j=1}^{s} \alpha^j \mathbf{m}^j = \tilde{\mathbf{q}}^0$.

Доказательство. Пусть B — выпуклая оболочка, натянутая на область L. Рассмотрим соотношение между B и поверхностью G, проходящей через $\tilde{\mathbf{q}}^0$. В принципе мыслимы четыре случая: 1) B и G пересекаются, 2) B и G не имеют общих точек, 3) B и G касаются, но не в точке $\tilde{\mathbf{q}}^0$, 4) B и G касаются в точке $\tilde{\mathbf{q}}^0$.

Первый случай не может иметь места, так как при этом не существует гиперплоскости, отделяющей B (а значит, и L) от G.

Реализация второго случая противоречила бы предположению, что $\tilde{\mathbf{q}}^0$ есть *сильнейшая* содержащая только истину гипотеза. Действительно, мы могли бы построить другую поверхность G', охватывающую G и не пересекающуюся с B. На G' всегда найдется точка $\mathbf{t}$ такая, что гиперплоскость, касательная в этой точке к G', отделяет G' от B. Значит, $\mathbf{t}$ было бы гипотезой, содержащей только истину. Но $H(\mathbf{t}) < H(\tilde{\mathbf{q}}^0)$, что и доказывает противоречивость предположений.

Третий случай, как и первый, несовместим с предположением, что $\tilde{\mathbf{q}}^0$ содержит только истину. В самом деле, G есть выпуклая поверхность, не имеющая плоских частей. Поэтому гиперплоскости, касательные к ней в двух разных точках, не могут совпадать. Так как B и G касаются, то единственная разделяющая их гиперплоскость касается G в этой же точке, а значит, касательная к G в $\tilde{\mathbf{q}}^0$ гиперплоскость не отделяет B от G.

[10] L может иметь точки на F

Итак, остается только четвертая возможность. Это значит, что $\tilde{\mathbf{q}}^0$ обязательно принадлежит B. Так как B есть выпуклая оболочка области L, то $\tilde{\mathbf{q}}^0$ является центром тяжести нескольких точек $\mathbf{m}^1, \mathbf{m}^2, \ldots, \mathbf{m}^j, \ldots, \mathbf{m}^s$ области L, которым приписаны положительные веса.

Последнее утверждение и означает, что найдутся такие числа α^j и точки $\mathbf{m}^j \in L$, что

$$\sum_{j-1}^{s} \alpha^j = 1 \tag{5}$$

и

$$\sum_{j-1}^{s} \alpha^j \mathbf{m}^j = \tilde{\mathbf{q}}^0, \tag{6}$$

причем

$$\alpha^j > 0 \qquad (j = 1, 2, \ldots, s). \tag{7}$$

Так как $\tilde{\mathbf{q}}^0$ принадлежит гиперплоскости F, а L расположена по одну сторону от F, то $\mathbf{m}^j \in F$ и, следовательно, $\mathbf{m}^j \in M$. Лемма доказана. □

Следствие 1. *Если* $\mathbf{t} = \tilde{\mathbf{q}}^0$, *то* M *не пусто.*

Отсюда

$$\max_{\mathbf{p}} N(\mathbf{p}|\tilde{\mathbf{q}}^0) = H(\tilde{\mathbf{q}}^0). \tag{8}$$

Переходим к доказательству теоремы. Из формулы 6 следует

$$N(\tilde{\mathbf{q}}^0|\mathbf{q}) = \sum_{j=1}^{s} \alpha^j N(\mathbf{m}^j|\mathbf{q}); \tag{9}$$

учитывая 5 и 7, получаем $N(\tilde{\mathbf{q}}^0|\mathbf{q}) \leqslant \max_j N(\mathbf{m}^j|\mathbf{q})$, где $\max_j$ есть максимум, взятый по s точкам $\mathbf{m}^j$. Так как $\mathbf{m}^j \in L$, то

$$N(\tilde{\mathbf{q}}^0|\mathbf{q}) \leqslant \max_{\mathbf{p}} N(\mathbf{p}|\mathbf{q}). \tag{10}$$

В случае $\mathbf{q} \neq \tilde{\mathbf{q}}^0$

$$H(\tilde{\mathbf{q}}^0) < N(\tilde{\mathbf{q}}^0|\mathbf{q}).(38) \tag{11}$$

Сопоставив 8 и 11 с 10, получаем

$$\max_{\mathbf{p}} N(\mathbf{p}|\tilde{\mathbf{q}}^0) < \max_{\mathbf{p}} N(\mathbf{p}|\mathbf{q}). \tag{12}$$

Это и означает, что $\tilde{\mathbf{q}}^0$ есть оптимальная гипотеза, и притом единственная. □

Следствие 2. *а) Из соотношения 8 вытекает*

$$\max_p N(\mathbf{p}|\tilde{\mathbf{q}}^0) = H(\mathbf{q}^0). \tag{13}$$

б) Из единственности оптимальной гипотезы следует единственность сильнейшей содержащей только истину гипотезы.

в) Если область L такова, что не существует $(n-2)$-мерной гиперплоскости, отделяющей ее от точки $\mathbf{h}$, то оптимальной гипотезой является $\mathbf{q}^0 = \mathbf{h}$. Например, если на $\mathbf{p}$ не наложено никаких ограничений (кроме $\sum p_i = 1$), то оптимальной будет гипотеза $q_i^0 = 1/n$.

Пункт в) поясняет, в каком смысле целесообразно пользоваться распространенным приемом: «поскольку мы не имеем сведений о вероятности некоторых событий, будем считать их равновероятными».

Рассмотрим случай, когда система ограничений 1 имеет вид $p_{j_1} + p_{j_2} + \ldots + p_{j_k} = 0$. Это означает, что ответ может принимать не n, а только $n-k$ значений. Поскольку про остальные координаты вектора $\mathbf{p}$ ничего не сказано, очевидно, оптимальной будет гипотеза $p_i = 0$ при $i = j_s$ и $p_i = \frac{1}{n-k}$ при $i \neq j_s$. Заметим, что, оценивая в первом примере, с которого начался этот параграф, истинность гипотезы о том, что ссылка находится в первой половине книги, мы подсознательно считали, что эта гипотеза говорит о *равновероятности* всех страниц первой половины. Другими словами, мы подсознательно оптимизировали гипотезу в пределах, допускаемых не сделанными явно оговорками.

В результате оптимизации получилась гипотеза, содержащая только истину. При уменьшении свободы выбора гипотеза о том, что ссылка находится в первой половине книги, может оказаться содержащей не только истину. Например, гипотеза: «ссылка с вероятностью 0.99 находится между 1 и 10 стр. и с вероятностью 0.01 — между 11 и 50 стр.» — содержит не только истину, хотя и утверждает, что ссылка — в первой половине книги.

Итак, оценивая истинность гипотезы, мы «додумываем» ее так, чтобы оптимизировать. Если после этого получается гипотеза, содержащая только истину, то мы склонны и исходное высказывание (ограничивающее некоторую область гипотез) считать истинным. Можно

сказать, что имеет место «презумпция истинности». Пока не доказано, что некто лжет, мы полагаем, что он говорит правду.

В заключение рассмотрим еще одно обстоятельство, поясняющее, почему гипотеза, удовлетворяющая соотношению 2, названа *содержащей только истину.* Пусть неравенства 1 определяют некоторую область L (рис. 5), внутри которой *в действительности* может находиться $\mathbf{p}$. Допустим, нам сообщили *не все* неравенства. Другими словами, нам сообщили «только истину, но не всю истину». Сообщенные нам ограничения определяют область L'. Очевидно, $L' \supset L$. Если наша задача состоит в том, чтобы найти оптимальную гипотезу, то при имеющихся в нашем распоряжении сведениях мы выберем $\mathbf{q}^0$-гипотезу, оптимальную относительно области L'. Относительно области L она не обязательно будет оптимальной, но всегда будет *содержащей только истину.*

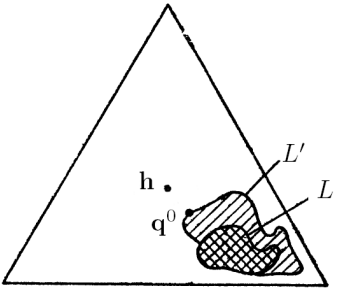

Рис. 5.

Приложение 2. Вопрос об оптимальных решающих алгоритмах при функциях цены трудности, отличных от логарифмической

Некоторые пояснения и определения[11]

Будем рассматривать модельную систему, которая при решении задачи ведет экспериментальную работу (метод проб и ошибок) и таким путем извлекает некоторые сведения, которых она вначале не имела. В качестве «меры трудности» задачи для этой системы принимается некоторая функция числа проб, необходимых для нахождения решения. Кроме того, система может получать сведения о задаче через канал связи. Следствием этого является изменение последовательности проб. В результате этого изменится число требуемых проб, а значит, и трудность задачи.

В качестве меры трудности используется логарифм среднего числа проб.

Пусть мы имеем конечное множество объектов $M = \{m_l\}$, на котором задано распределение вероятностей $p(m_1)$, $p(m_2)$,..., $p(m_l)$,...,$p(m_r)$. Это множество разбито на n подмножеств A_i таких, что $A_1 \cup A_2 \ldots \cup A_n = M$ и $A_i \cap A_j = \emptyset$ при $i \neq j$. Такую ситуацию мы будем называть *задачей* **A**.

Решение задачи в отношении некоторого объекта m_l заключается в нахождении такого i, что $m_l \in A_i$. В соответствии с этим будем говорить: «нам нужно решить задачу **A**», если с вероятностью $p(m_1)$ нам придется решать задачу в отношении m_1, с вероятностью $p(m_2)$ — в отношении m_2 и т. д.

Пусть мы обладаем алгоритмом проверки утверждения $m_l \in A_i$; назовем его *W-алгоритмом.*

[11] М. М. Бонгард, «Проблема узнавания», глава VII.

В дальнейшем мы рассматриваем некоторый класс алгоритмов, которые будут называться *решающими*.

Описание решающего алгоритма. С помощью жребия с некоторым распределением вероятностей выбирается одно из подмножеств A_i. Затем подставляется в W-алгоритм объект m_l, в отношении которого решается задача, и выбранное с помощью жребия A_i. Если ответ — «истина», то решение закончено, если — «ложь», то снова бросаем жребий, и т. д. При этом распределение вероятностей быть выбранными для разных A_i может оставаться постоянным, а может изменяться от шага к шагу.

Различные решающие алгоритмы отличаются друг от друга начальными распределениями вероятностей жребия и законами их изменения.

Если заданы объект m_l и решающий алгоритм, то число применений W-алгоритма (проб), приводящее к отысканию ответа, есть случайная величина $K(m_l)$. Назовем *неопределенностью задачи в отношении объекта m_l для данного решающего алгоритма* логарифм математического ожидания $\overline{K}(m_l)$ числа проб, т. е.

$$N(m_l) = \log \overline{K}(m_l).$$

Неопределенностью задачи $\mathbf{A}$ *для данного решающего алгоритма* будем называть математическое ожидание неопределенности в отношении всех объектов

$$N(\mathbf{A}) = \sum_l N(m_l) = \sum_l p(m_l) \log \overline{K}(m_l).$$

Вопрос об оптимальных решающих алгоритмах при функциях цены трудности, отличных от логарифмической

Почему в качестве характеристики трудности задачи для данного решающего алгоритма мы избрали именно логарифм среднего числа проб? Почему не число проб или не квадрат числа проб? Совершенно очевидно, что в разных конкретных случаях «цена» за применения W-алгоритма должна быть весьма разной. Например, если реализация W-алгоритма — относительно самое длинное место в счетной программе, а ценим мы машинное время, то есть смысл считать трудность просто пропорциональной числу проб. В другом случае, если много времени и

затрат уходит на создание экспериментальной установки, а сами эксперименты (применение W-алгоритма) «стоят дешево» (особенно когда методика хорошо отработана), целесообразная оценка трудности будет выражаться некоторой функцией, очень быстро возрастающей вначале и относительно замедляющей свой рост при увеличении числа проб.

Универсального способа оценки трудности не может быть. Почему же столько места уделено некоторому частному случаю — логарифмической функции цены?

Теория информации не является исключительной областью, где возникают подобные вопросы. Что, например, «лучше» — источник постоянного тока напряжением 12 вольт, дающий ток до 15 ампер, или источник переменного тока 120 вольт, с пределом силы тока 1 ампер? Ответить на этот вопрос невозможно, если не сказано, для какой цели нам нужен источник тока. Допустим, однако, что, кроме источника тока, мы имеем возможность применять различные преобразователи тока. Это значит, что можно ставить в цепь выпрямители, трансформаторы, умформеры и т. п. Если не наложено никаких ограничений на эту дополнительную аппаратуру, то появляется возможность сравнить описанные выше источники. В самом деле, если с помощью понижающего трансформатора и выпрямителя превратить 120 вольт переменного тока в 12 вольт постоянного, то максимальная сила постоянного тока не может быть больше, чем 10 ампер. Это ограничение возникает вследствие *наличия закона сохранения энергии*. Если, наоборот, превратить с помощью какого-либо преобразователя постоянный ток от первого источника в переменный с напряжением 120 вольт, то *в принципе* можно получить ток, больший 1 ампера (теоретический предел 1,5 ампера).

По существу, мы просто сравнили мощности двух источников. У первого она 180 ватт, а у второго 120 ватт. Если мы не имеем преобразователей, то мощность может оказаться вовсе не решающим обстоятельством для предпочтения одного источника другому. Например, для питания лампочки 120 вольт 40 ватт второй источник годится, а первый — нет. При наличии же преобразователей мощность становится *всеобщим эквивалентом* потому, что именно для нее (а не для напряжения, частоты, формы импульсов и т. п.) существует закон сохранения[12] (точнее, закон невозрастания). Таким образом, именно законы

[12]Предполагается, что мы лишены возможности запасать энергию (например, с

сохранения (при возможности соответствующих преобразований) выделяют величины, которые есть смысл возводить в ранг *всеобщих эквивалентов*.

Вспомним теперь, зачем нам понадобилось понятие *неопределенность задачи*. Для решения некоторой задачи система ведет экспериментальную работу. Система изначально «знает», что ответом является одно из A_i, но в нее не заложены сведения, какое именно A_i есть ответ данной задачи. Недостаток знаний решающий алгоритм восполняет с помощью опытов. Существует и другой способ получения сведений об ответе задачи. Это — использование сообщений, приходящих по каналу связи. Мы хотели найти «коэффициент эквивалентности» между этими двумя способами получения сведений. В каких единицах нужно «измерять» эксперимент, необходимый для решения задачи, чтобы уметь предсказывать изменения, которые могут возникнуть в этой величине после прихода данного сигнала?

Связь между *количеством необходимого эксперимента* и свойствами канала связи устанавливается, если измерять *необходимый эксперимент* неопределенностью задачи (в логарифмических единицах)[13].

Связь эта типа закона сохранения: неопределенность задачи не может уменьшиться более чем на пропускную способность канала связи. Мы, однако, знаем, что одного лишь наличия закона сохранения мало для того, чтобы имело смысл сравнивать с помощью «сохраняющейся» величины различные задачи, каналы или сигналы. Необходимо еще, чтобы существовали преобразователи, позволяющие осуществить сопряжение канала связи и сигнала с решающим алгоритмом. Оказывается, в нашем случае такой преобразователь есть. Один из вариантов построения матрицы декодирующего алгоритма дает возможность решающему алгоритму полностью использовать пропускную способность канала связи[14]. Эти соображения и объясняют, в каком смысле логарифмическая функция цены трудности является преимущественной.

Может возникнуть опасение, что искусственный и ничем (кроме закона сохранения) не мотивированный способ вычисления неопределенности должен приводить к результату, мало соответствующему нашим интуитивным представлениям об успешности работы системы. Однако оказывается, что иногда этот искусственный метод дает результат,

помощью аккумуляторов

[13] М. М. Бонгард, «Проблема узнавания», глава VII, формула (11).

[14] М. М. Бонгард, «Проблема узнавания», глава VII, формулы (12)-(13)

гораздо более соответствующий нашей интуиции, чем, например, естественное вычисление среднего числа проб.

Приведем пример такой ситуации. Пусть имеется задача, для решения которой алгоритм, исходящий из гипотезы о равномерном распределении вероятностей ответов, тратит в среднем 10^9 проб. Пусть, далее, мы имеем две системы, решающие эту задачу. Первая система обладает следующим свойством: в 99% случаев она решает задачу в среднем за 1000 проб, а в 1% случаев — за 10^9 проб. Вторая система в 99% случаев находит решение за 10 проб, а в 1% — за те же 10^9 проб.

На вопрос: какая из систем лучше решает задачу? — всякий человек ответит, что вторая система заметно лучше.

Попробуем сравнить работу систем, опираясь на среднее число проб. Для первой системы среднее число проб равно $0.99 \cdot 1000 + 0.01 \cdot 10^9 \approx 10^7 \cdot 1,0001$. Для второй системы $0.99 \cdot 10 + 0.01 \cdot 10^9 \approx 10^7 \cdot 1.000001$. Относительная разница столь ничтожна, что, пользуясь критерием «среднее число проб», мы признали бы обе системы практически равноценными при решении этой задачи.

Сравним теперь системы, пользуясь неопределенностью задачи для них (будем брать десятичные логарифмы). Для первой системы неопределенность равна

$$0.99 \lg 1000 + 0,01 \lg 10^9 = 0.99 \cdot 3 + 0.01 \cdot 9 = 3.06.$$

Для второй системы

$$0.99 \lg 10 + 0.01 \lg 10^9 = 0.99 \cdot 1 + 0.01 \cdot 9 = 1.08.$$

Разница очень заметна и хорошо соответствует интуитивному ощущению, что вторая система *почти всегда* решает задачу в сто раз быстрее, чем первая, а в 1% случаев обе системы решить задачу просто не могут.

Итак, возможен случай, когда *неопределенность* оказывается ближе к интуитивной оценке, чем *среднее число проб*.

Доводы в пользу логарифмической функции цены числа проб нисколько не умаляют того обстоятельства, что в каждом конкретном случае нас может интересовать своя (вовсе не логарифмическая) функция цены. Возникает вопрос: не являются ли оптимальные решающие и декодирующие алгоритмы[15] оптимальными только для *неопределенности*? Не придется ли для каждой функции цены трудности искать

[15] М. М. Бонгард, «Проблема узнавания», глава VII, параграфы 3, 5, 6

свои решающие и декодирующие алгоритмы? Это было бы, очевидно, очень неудобно.

Если ограничить класс решающих алгоритмов алгоритмами без обратной связи, то, действительно, для разных функций цены трудности наилучшими будут разные распределения вероятностей жребия. Например, для алгоритмов без обратной связи при $n = 2$ наименьшее *среднее число* применений W-алгоритма достигается при

$$q_1 = \frac{p_1 - \sqrt{p_1 - p_1^2}}{2p_1 - 1}, \quad q_2 = \frac{p_2 - \sqrt{p_2 - p_2^2}}{2p_2 - 1},$$

а не при $q_1 = p_1$ и $q_2 = p_2$, обеспечивающих наименьшую *неопределенность*.

Вспомним, однако, что алгоритмы с постоянным распределением вероятности жребия (алгоритмы без обратной связи) не являются абсолютно оптимальными. В случае же, если мы снимем запрещение использовать информацию о предыдущих испытаниях для изменения вероятностей жребия, то оптимальным для весьма широкого класса функций цены оказывается один и тот же решающий алгоритм. К доказательству этого положения мы и переходим.

Пусть $F(x)$ — неубывающая функция, определенная при $x \geqslant 1$. Будем называть F *функцией цены трудности. Трудностью задачи в отношении объекта* m_l для данного решающего алгоритма будем называть $T(m_l) = F(\overline{K}_l)$, где $\overline{K}_l$ — среднее число применений W-алгоритма при решении задачи в отношении объекта m_l.

Трудностью задачи **A** назовем величину

$$T(\mathbf{A}) = \sum_l p(m_l)T(m_l) = \sum_l p(m_l)F(\overline{K}_l).$$

Легко видеть, что *неопределенность* есть *трудность с логарифмической функцией цены*. Назовем $F(x)$ выпуклой, если при $\alpha_1 > 0$, $\alpha_2 > 0$ и $\alpha_1 + \alpha_2 = 1$

$$\alpha_1 F_1(x) + \alpha_2 F_2(x) \leqslant F(\alpha_1 x_1 + \alpha_2 x_2). \tag{14}$$

Теорема 2. *При любой выпуклой функции цены наименьшей будет трудность для решающего алгоритма, проверяющего* A_i *в порядке убывания вероятностей* p_i.

Доказательство. Вначале заметим, что повторные проверки одних и тех же A_i могут лишь увеличить среднее число проб, а значит и трудность. Поэтому мы рассматриваем только алгоритмы с обратной связью. Будем считать, что индексы при A_i расставлены так, что

$$p_1 \geqslant p_2 \geqslant \ldots \geqslant p_n.$$

Обозначим через $\pi_{i,j}$ вероятность того, что если ответ A_i, то он будет найден на j-м такте. Очевидно,

$$\sum_{j=1}^{n} \pi_{i,j} = 1, \tag{15}$$

так как ответ обязательно будет найден не более чем за n тактов. Можно также показать, что

$$\sum_{i=1}^{n} \pi_{i,j} = 1, \tag{16}$$

Соотношение 16 имеет место благодаря тому, что $\pi_{i,j}$ является одновременно условной вероятностью проверки A_i на j-м такте, *если* первые $j-1$ тактов не обнаружили ответа.

Трудность задачи A для решающего алгоритма, характеризуемого матрицей $||\pi_{i,j}||$, равна

$$T(\mathbf{A}) = \sum_i p_i F\left(\sum_j j\pi_{i,j}\right). \tag{17}$$

Обозначим $T(\mathbf{A})$ через $f(\mathbf{p}, ||\pi_{i,j}||)$.

Введем функцию f' следующим образом:

$$f'(\mathbf{p}, ||\pi_{i,j}||) = \sum_i p_i \sum_j \pi_{i,j} F(j). \tag{18}$$

Благодаря выпуклости F и равенству 15

$$f(\mathbf{p}, ||\pi_{i,j}||) \geqslant f'(\mathbf{p}, ||\pi_{i,j}||). \tag{19}$$

Рассмотрим матрицу

$$||\pi'_{i,j}|| = \left\| \begin{array}{ccccccc} \pi_{11} & \dots & \pi_{1,j_1} & \dots & \pi_{1,j_2} & \dots & \pi_{1,n} \\ \dots & & \dots & & \dots & & \dots \\ \pi_{i_1,1} & \dots & \pi_{i_1,j_1}+\alpha & \dots & \pi_{i_1,j_2}-\alpha & \dots & \pi_{i_1,n} \\ \dots & & \dots & & \dots & & \dots \\ \pi_{i_2,1} & \dots & \pi_{i_2,j_1}-\alpha & \dots & \pi_{i_2,j_2}+\alpha & \dots & \pi_{i_2,n} \\ \dots & & \dots & & \dots & & \dots \\ \pi_{n,1} & \dots & \pi_{n,j_1} & \dots & \pi_{n,j_2} & \dots & \pi_{n,n} \end{array} \right\|, \qquad (20)$$

полученную из $||\pi_{i,j}||$ изменением четырех элементов, где $\alpha > 0$, $j_2 > j_1$ и $i_2 > i_1$.

При этом

$$f'(\mathbf{p}, ||\pi_{i,j}||) - f'(\mathbf{p}, ||\pi'_{i,j}||) = \\ = \alpha(p_{i_1} - p_{i_2})[F(j_2) - F(j_1)] \geqslant 0. \qquad (21)$$

Свойства 15 и 16 сохраняются после преобразования 20. Пусть $\pi_{1,t}$ — первое отличное от $\pi_{1,1}$ и не равное нулю число в первой строке матрицы $||\pi_{i,j}||$, а $\pi_{s,1}$ — первое отличное от $\pi_{1,1}$ и не равное нулю число в первом столбце. Произведем преобразование 20, приняв $i_1 = 1$; $i_2 = s$; $j_1 = 1$; $j_2 = t$, а за α взяв меньшее из чисел $\pi_{1,t}$ и $\pi_{s,1}$. Если в исходной матрице $||\pi_{i,j}||$ элемент $\pi_{1,1}$ не был равен единице, то после преобразования в матрице $||\pi'_{i,j}||$ в первой строке или в первом столбце станет на один нуль больше.

Будем повторять эту операцию до тех пор, пока не получим матрицу вида

$$||\pi^2_{i,j}|| = \left\| \begin{array}{ccccc} 1 & 0 & 0 & \dots & 0 \\ 0 & \pi^2_{2,2} & \pi^2_{2,3} & \dots & \pi^2_{2,n} \\ 0 & \pi^2_{3,2} & \pi^2_{3,3} & \dots & \pi^2_{3,n} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \pi^2_{n,2} & \pi^2_{n,3} & \dots & \pi^2_{n,n} \end{array} \right\|.$$

Для этого понадобится не более $2n-3$ операций. Так как при каждом преобразовании выполняется соотношение 21, то

$$f'(\mathbf{p}, ||\pi_{i,j}||) \geqslant f'(\mathbf{p}, ||\pi^2_{i,j}||).$$

Проделаем ту же серию операций со второй строкой и вторым столбцом и т. д. до получения единичной матрицы $||\pi^n_{i,j}||$. Очевидно,

$$f'(\mathbf{p}, ||\pi_{i,j}||) \geqslant f'(\mathbf{p}, ||\pi^n_{i,j}||). \qquad (22)$$

Но $f'(\mathbf{p}, ||\pi^n_{i,j}||)$ совпадает с $f(\mathbf{p}, ||\pi^n_{i,j}||)$ — трудностью задачи $\mathbf{A}$ для алгоритма, пробующего с вероятностью единица на первом шаге A_1, на втором A_2 и т. д. Сопоставив это с 22, 46 и 44, получаем

$$T(\mathbf{A}) \geqslant f(\mathbf{p}, ||\pi^n_{i,j}||) = \sum_i p_i F(i), \tag{23}$$

чем и доказана теорема[16]. □

Так как логарифм является выпуклой функцией, то для него справедлива эта теорема. Отсюда следует оптимальность алгоритма с жестким порядком перебора[17].

Доказанная теорема имеет еще одно важное следствие: для всех выпуклых функций цены трудности существует один и тот же оптимальный декодирующий алгоритм, совпадающий с декодирующим алгоритмом, оптимальным для неопределенности.

В дополнении 1 рассматривалось понятие оптимальной гипотезы для совокупности задач (для алгоритмов без обратной связи). Его можно весьма естественно распространить на случай нелогарифмических функций цены трудности. Несколько сложнее с понятием гипотезы, содержащей только истину. В определение 2, кроме неопределенности, входит еще и энтропия распределения $\mathbf{q}$. Что будет аналогом энтропии при других функциях цены трудности?

Пусть $F(x)$ — функция цены трудности. Обозначим через $H^F(\mathbf{p})$ минимум трудности задачи $\mathbf{A}$ для решающего алгоритма без обратной связи. Легко видеть, что по отношению к рассматриваемой трудности функция H^F играет ту же роль, что и энтропия по отношению к неопределенности. В частности, гиперплоскость, касательная в точке $\mathbf{t}$ к поверхности с уравнением[18] $y = H^F(\mathbf{q})$, выражает трудность задач при гипотезе $\mathbf{t}$ (ср. рис. 1).

Поэтому если мы в определениях 2 и 3 заменим $H(\mathbf{q})$ на $H^F(\mathbf{q})$, то теорема о совпадении оптимальной гипотезы с сильнейшей гипотезой,

[16] Иногда может оказаться целесообразным определять трудность задачи в отношении m_l через $\overline{F(k)}$, а не через $F(\overline{k})$, как это сделано выше. В этом случае $T(\mathbf{A}) = f'(\mathbf{p}, ||\pi_{i,j}||)$. Отсюда следует справедливость доказанной теоремы и для трудности, введенной таким способом. При этом отпадает требование выпуклости функции $F(x)$; необходимо только, чтобы она была неубывающей.

[17] М. М. Бонгард, «Проблема узнавания», глава VII, параграф 6.

[18] $H^F(\mathbf{p})$ выпукла, так как через каждую точку этой поверхности можно провести гиперплоскость, не пересекающую $H^F(\mathbf{p})$.

содержащей только истину, останется справедливой при оценке трудности с помощью функции $F(x)$.

Например, Н. Д. Нюбергу удалось показать, что для линейной функции цены трудности $H^F(\mathbf{p}) = \left(\sum_i \sqrt{p_i}\right)^2$. Пользуясь вместо энтропии этим выражением, мы можем говорить о «гипотезах, содержащих только истину» при оценке не по неопределенности, а по среднему числу проб.

Николай Константинович Верещагин
Евгений Витальевич Щепин

ИНФОРМАЦИЯ, КОДИРОВАНИЕ И ПРЕДСКАЗАНИЕ

Подписано в печать 6.02.2012. Формат $60\times90\,{}^{1}\!/_{16}$.
Бумага офсетная № 1. Печать офсетная. Печ. л. 15. Тираж 1500 экз.

Издательство Московского центра
непрерывного математического образования.
119002, Москва, Большой Власьевский пер., д. 11. Тел. (499) 241-74-83.

Отпечатано с готовых диапозитивов в ООО «Принт Сервис Групп».
Москва, ул. Борисовская, д. 14.

Книги издательства МЦНМО можно приобрести в магазине
«Математическая книга», Большой Власьевский пер., д. 11. Тел. (499) 241-72-85.
E-mail: biblio@mccme.ru