Будем писать просто log вместо $\log_2$.

Сначала определим понятие информационной энтропии.

Энтропия измеряет количество информации, которую мы можем получить из данного случайного эксперимента.

Определение 1. Энтропия функции на событиях ξ --- это сумма по принимаемым значениям $x - P(\xi = x) \log P(\xi = x)$ (напомним, что символом логарифма без указания основания мы будем обозначать двоичный алгоритм).

Энтропию будем обозначать буквой H, например, энтропия случайной величины ξ обозначается $H(\xi)$.

Можно сказать, что мы рассматриваем случайную величину, которая по событию возвращает минус логарифм его вероятности, и берём её математическое ожидание. Это отражает идею, что мы получаем больше информации, узнав о событии с малой вероятностью. Если мы и так приписывали событию вероятность 1, то узнав, что оно произошло, мы только подтвердили свои прежние знания (на которые опиралась оценка вероятности).

Перечислим простейшие свойства энтропии.

Энтропия константы равна 0.

Энтропия пары независимых функций равна сумме их энтропий. Действительно, вероятность принятия ими данной пары значений равна произведению вероятностей принятия ими этих значений по отдельности; при логарифмировании произведение переходит в сумму, а ожидание суммы равно сумме ожиданий.

Теперь свяжем энтропию случайной величины и количество битов, нужных для её передачи. Пусть мы хотим передавать значение случайной величины так, чтобы просто по потоку данных было сразу понятно, закончена ли передача. Например, мы можем хотеть без дополнительных усилий записывать подряд (без разделителей) коды значений разных случайных величин.

Определение 2. Код называется префиксным, если никакой код значения не является продолжением кода другого значения. Таким образом, префиксный код можно представить в виде двоичного дерева со значениями в листьях.

Докажем, что математическое ожидание длины кода не меньше энтропии.

Обозначим длину i-го кодового слова l_i. Тогда $\sum 2^{-l_i} \leqslant 1$. Это можно доказать индукцией: для дерева кодирования из одной вершины это так (есть одно кодовое слово длины 0), а произвольное дерево кодирования состоит из корня и двух кодовых деревьев --- левого и правого. При этом длина каждого кодового слова в левом и правом поддеревьях увеличивается на 1 по сравнению с их самостоятельным рассмотрением, и интересующая на сумма оказывается равна $2 \times \frac{1}{2} = 1$.

Теперь рассмотрим разность между $\sum -p_i \log p_i$ (энтропией) и $\sum -p_i \log 2^{-l_i}$ (ожидаемой длиной кода, так как $-\log 2^{-l_i} = l_i$). Сразу воспользуемся, что разность логарифмов --- это логарифм отношения.

$$\sum -p_i \log p_i - \sum -p_i \log 2^{-l_i} = \sum p_i \log \frac{2^{-l_i}}{p_i} \leqslant \log \sum p_i \frac{2^{-l_i}}{p_i} = \log \sum 2^{-l_i} \leqslant \log 1 = 0$$

Здесь первое неравенство использует то, что логарифм --- выпуклая вверх функция, то есть логарифм выпуклой комбинации (суммы с неотрицательными коэффициентами, в сумме дающими 1) нескольких чисел не меньше такой же выпуклой комбинации их логарифмов.

Утверждение доказано.

Теперь построим код, который имеет математическое ожидание длины кодового слова не хуже энтропии, увеличенной на 1.

Возьмём вероятности всех событий и округлим вниз до ближайших отрицательных степеней двойки. Мы занизим вероятности не более, чем в два раза. Теперь у нас имеются степени двойки с суммой не более 1. Будем выбирать минимально возможные коды для значений, идя в порядке от больших вероятностей к маленьким. Заметим, что каждому коду длины l можно сопоставить отрезок длины 2^{-l} на отрезке $[0; 1]$ с началом, делящимся на 2^{-l}. Выбирая коды, мы будем просто замащивать отрезок $[0; 1]$. Начало каждого из кодовых отрезков делится на его длину, так как все предыдущие отрезки не короче его, а следовательно, их длины делятся на длину текущего отрезка. Обозначая, как и раньше, длину i-го кодового слова за l_i, а вероятность i-го события за p_i, получим математическое ожидание длины кодового слова, равное

$$\sum p_i l_i = \sum -p_i \log 2^{-l_i} \leqslant \sum -p_i \log \frac{p_i}{2} = \sum -p_i(\log p_i - 1) = H + 1$$

На практике часто используют код Хаффмана. Этот код имеет минимальное ожидание длины кодового слова. Построим его, попутно доказывая, что этот код является оптимальным.

Как произойдёт построение: мы строим кодовое дерево. На первом шаге у нас есть отдельные значения и их вероятности, мы их трактуем как деревья из одной вершины. На каждом шаге у нас есть какие-то деревья кодирования и суммарные вероятности их листьев; мы находим две минимальные вероятности и присоединяем соответствующие деревья к общему корню, получая одно дерево.

Докажем следующее утверждение: пусть есть набор деревьев кодирования для непересекающихся наборов значений случайной величины, в сумме покрывающих все принимаемые значения. Тогда среди всех деревьев кодирования, дающих минимальное математическое ожидание длины кодового слова для выпадающего события и включающих все данные , найдётся такое, что два дерева с минимальными суммами вероятностей листьев из имеющихся входят в него и их корни являются дочерними вершинами для одной и той же вершины в кодом дереве для всех значений сразу.

Доказательство 1. Рассмотрим оптимальное среди включающих данные поддеревья дерево кодирования. Рассмотрим математическое ожидание длины кодового слова. Заметим, что мы можем влиять только на длины кодовых слов для корней данных поддеревьев; части кодов, заданные поддеревьями, дают всегда один и тот же вклад. Поэтому достаточно рассмотреть случай, когда у нас есть отдельные значения.

Рассмотрим два самых длинных кодовых слова. Можно считать, что они различаются в одном последнем символе. Ожидание длины кода не увеличится, если сопоставить их двум самым редким событиям (мы сопоставим менее вероятному значению более длинное слово) Это доказывает наш шаг.

Так как ожидание длины оптимального двоичного кода для события оказалось близко к энтропии, энтропию можно считать средним количеством битов информации, получаемой от эксперимента.

Количество информации можно использовать и для ещё одного определения невероятного.

Пусть мы кидаем симметричную монетку миллион раз, и все разы выпал орёл. Есть ли в этом что-то плохое? Конечно, после такого события невероятно, чтобы монетка была и правда симметричной --- неравенство Чебышева говорит, что такое большое отклонение количества орлов от среднего маловероятно. Но пусть монетка падает орлом только во все чётные разы, а во все нечётные --- решкой. Доля орлов такая, как должна быть, но всё равно хочется сказать, что что-то тут не так.

Применим количество информации, чтобы попытаться сформулировать, что именно здесь невероятно.

Можно определить не только количество информации в случайном эксперименте, но и попытаться определить количество информации в отдельно взятом объекте. Разумеется, количество информации, необходимой для описания объекта зависит от языка описания. Но с этим можно бороться следующим способом.

Мы будем рассматривать описания объектов, которые сами уже заданы последовательностями нулей и единиц (как в компьютере).

Определение 3. Префиксно-корректным способом описания двоичных слов называется отображение (не всюду определённое) из двоичных слов в двоичные слова, такое что для любого слова X, на котором отображение определено (и даёт результат Y), отображение определено на всех продолжениях слова X и даёт тот же результат Y на них.

Например, если взять код Хаффмена для произвольного набора двоичных слов и определить, что любое описание следует читать до границы первого символа, то мы получим префиксно корректный метод описания слов.

Разумеется, такой метод позволяет описывать не все слова. Кроме того, разные методы описания могут позволять краткую запись разных наборов слов. Например, можно рассмотреть следующий метод кодирования: рассмотрим произвольный способ префиксного кодирования десятичных чисел и пробела. Код, начинающийся с единицы должен содержать после него некоторые коды цифр, потом код пробела; цифры описывают сколько (как минимум) двоичных символов должно быть после кода пробела --- эти двоичные символы и считаются словом, заданным описанием. Код, начинающийся с нуля должен после этого нуля содержать некоторые коды цифр, а потом код пробела. Цифры описывают, сколько двоичных знаков числа π следует взять. Этот код отличается тем, что описать миллион знаков числа π много проще, чем миллион знаков числа 2π.

Другой пример странного кода говорит, что каждое возможное двоичное слово (включая пустое) описывает доказательство гипотезы Римана (в ту сторону, в которую она верна). Описание данного кода не сообщает нам, как именно его декодировать.

Поэтому мы будем рассматривать способы описания, для которых можно восстановить по описанию код каким-то чётко определённым способом. Раньше было принято говорить про машины Тьюринга, но нам не надо слишком много деталей, поэтому можно считать, что мы рассматриваем программы на языке Паскаль с неограниченно большими целыми числами и бесконечными массивами.

Теорема 1. Среди вычислимо декодируемых префиксно корректных способов описания есть оптимальный.

Точнее говоря, есть способ декодирования такой, что для любого другого вычислимого способа префиксно корректного декодирования оптимальный способ позволяет описание любого слова с не более чем константной разностью длин.

Разумеется, константа может зависить от способа описания.

Доказательство существования оптимального декодера описаний тривиально: запишем каждую программу префиксным кодом для используемого алфавита с дополнительным символом конца программы, после чего разрешим её обращаться к тому, что останется, как к данным. Скажем, что при попытке обратиться дальше конца имеющихся данных (как и при невозможности считать программу) результат не определён. Для любого способа описаний можно

просто написать его программу в начале описания (это константная длина) и получать не слишком сильно худший результат.

Сложностью префиксно корректного описания некоторого слова X будем называть длину минимального описания, которое наш оптимальный декодер соглашается декодировать и выдаёт на нём X.

Объясним теперь, почему нам кажется невероятным получение последовательности исходов с очень маленькой сложностью описаний. Для последовательности с маленькой сложностью можно придумать относительно простой механизм, обеспечивающий её получение вместо случайной последовательности. Если мы считаем, что вероятность того, что нас исходно обманули про случайность процесса очень мала, скажем $\frac{1}{1000}$, и условная вероятность конкретного простого вида обмана тоже мала, например $\frac{1}{10000}$, то мы получаем событие с вероятностью 10^{-7}, при наступлении которого монетка падает поочерёдно то орлом, то решкой из-за некоторой неслучайности.

Но по формуле Байеса уже для ста бросаний нам надо сравнивать произведения условных вероятностей получения нашей последовательности (при предположении обмана и при предположении случайности) и вероятностей собственно обмана и случайности. Разумеется, $10^{-7} \times 1 > 1 \times 2^{-100}$. Поэтому нам кажется, что условная вероятность правды при условии, что монетка падает поочерёдно орлом и решкой, мала.