Нумерационное кодирование

5 января 2023 г. 1:24

Имеем:

1. Последовательность на выходе источника **x** длины n: $x \in X^n$
2. Алфавит из M символов (индексы - от 0 до M-1): $X = \{0, 1, \ldots, M-1\}$
3. Вводим такую величину, как композиция - вектор, в котором каждое число говорит о том, сколько раз встретился символ. Например, $\tau_0(\boldsymbol{x})$ говорит о том, сколько раз встретился 0 в последовательности **x**.
   $\tau(\boldsymbol{x}) = (\tau_0(\boldsymbol{x}), \ldots, \tau_{M-1}(\boldsymbol{x}))$

Кодирование:

1. Кодовое слово состоит из двух частей: $\boldsymbol{c} = (\boldsymbol{c}_1, \boldsymbol{c}_2)$
2. $\boldsymbol{c}_1$ описывает $\tau = \tau(\boldsymbol{x})$, т.е. описывает композицию, содержащую информацию о том, сколько раз какой символ встречался в последовательности на выходе источника
3. $\boldsymbol{c}_2$ описывает номер **x** в лексикографически упорядоченном списке всех возможных {**x**}, которые имеют такую же композицию $\tau = \tau(\boldsymbol{x})$
   (например, есть строка из 10 букв c, 3 букв b и 1 буквы a, {**x**} - это все перестановки из них, отсортированные лексикографически, там надо найти **x** и передать его индекс)

Способы передачи композиции (как закодировать $\boldsymbol{c}_1$):

1. Можно передать прямым кодом, но будет затрачено очень много бит - плохой подход
2. Можно представить композицию как двоичную последовательность вида $0^{\tau_0}10^{\tau_1}1, \ldots, 10^{\tau_{M-1}}$, которая имеет длину (n+M-1) и вес (кол-во единиц) (M-1).
   Ищем все варианты строк такой же длины и такого же веса, лексикографически упорядочиваем их и кодируем равномерным кодом номер последовательности
3. Берем Q ненулевых компонент τ и упорядочиваем их по убыванию. Кодируем эту упорядоченную последовательность с равномерными вероятностями арифметическим кодером. Далее при помощи арифметического кодера кодируем буквы, которые соответствуют композиции. Первая буква кодируется с вероятностью 1/M, вторая - с вероятностью 1/(M-1), т.к. первая уже не берется в расчет, и т.д.

То же самое из презентации:

1. Кодируем каждый $\tau_i(\boldsymbol{x})$, кроме $i = M-1$, прямым кодом, используя $\lceil \log(n+1) \rceil$ бит.

2. Представим композицию $\tau_0(\boldsymbol{x}), \ldots, \tau_{M-1}(\boldsymbol{x})$ как двоичную последовательность вида $0^{\tau_0}10^{\tau_1}1, \ldots, 10^{\tau_{M-1}}$, которая имеет длину $(n+M-1)$ и вес (количество единиц) $M-1$.
   - Количество строк такой же длины и веса: $N_\tau(n, M) = \binom{n+M-1}{M-1}$. ← сочетание из n+M-1 по M-1
   - Лексикографически упорядочиваем все строки.
   - Кодируем равномерным кодом номер последовательности, используя $\lceil N_\tau(n, M) \rceil$ бит.

3. Упорядочим Q ненулевых компонент $\boldsymbol{\tau}$ по убыванию, т.е., $\tau_0 \in \{1, .., n\}$, $\tau_1 \in \{1, .., \tau_0\}$, $\tau_2 \in \{1, .., \tau_1\}$ и т.д.
   - $\boldsymbol{\tau}_Q$ кодируем АК с вероятностью $\frac{1}{n}\frac{1}{\tau_0}\frac{1}{\tau_1}\frac{1}{\tau_2}\ldots\frac{1}{\tau_{Q-2}}$.
   - При помощи АК кодируем буквы, которые соответствуют компонентам композиции с вероятностью $\frac{1}{M}\frac{1}{M-1}\frac{1}{M-2}\cdots\frac{1}{M-Q+2}$.

Далее надо закодировать вторую часть кодового слова ($\boldsymbol{c}_2$)

Рассмотрим на примере для M=3, $\tau = (\tau_0, \tau_1, \tau_2)$

В этом случае количество всех возможных **x** для $\tau(\boldsymbol{x})$ будет:

$$N(\boldsymbol{\tau}) = \binom{n}{\tau_0}\binom{n-\tau_0}{\tau_1} =$$

$$\frac{n!}{\tau_0!(n-\tau_0)!}\frac{(n-\tau_0!)}{\tau_1!(n-\tau_0-\tau_1)!} = \frac{n!}{\tau_0!\tau_1!\tau_2!}$$

В общем случае для алфавита объемом M имеем:

$$N(\boldsymbol{\tau}) = \frac{n!}{\tau_0!\tau_1!\ldots\tau_{M-1}!}$$

Эта величина покажет, сколько нужно бит, чтобы передать $\boldsymbol{c}_2$, т.е. номер **x** в лексикографически упорядоченном списке всех возможных {**x**}

Это делается с помощью арифметического кодера

Берем из кодируемого сообщения длиной n по одному символу и кодируем с учетом композиции, т.е. зная, сколько раз каждый символ встречается в сообщении. Вероятность для каждого символа будет:

$$\frac{\tau\ этого\ символа}{n\ -\ кол-во\ уже\ пройденных\ символов}$$

В процессе, как мы понимаем, обновляется n и композиция. С каждым символом n уменьшается на 1, а в композиции соответствующее символу значение уменьшается на 1

IF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CAN

t	x	$\hat{p}(x)$	Композиция $\boldsymbol{\tau}(\boldsymbol{x})$
0	—	—	12,5,5,4,4,4,3,3,2,2,2,1,1,1,1
1	I	1/50	12,5,5,4,4,4,3,3,2,2,2,1,1,1,0
2	F	1/49	12,5,5,4,4,4,3,3,2,2,2,1,1,0
3	_	12/48	11,5,5,4,4,4,3,3,2,2,2,1,1
4	W	5/47	11,4,5,5,4,4,4,3,3,2,2,2,1,1

5	E	4/46	11,4,5,5,3,4,4,3,3,2,2,2,1,1
6	_	10/45	10,4,5,5,3,4,4,3,3,2,2,2,1,1
...	...	...	...

$$G = \frac{12!(5!)^2(4!)^3(3!)^2(2!)^3}{50!}$$

$$L = \lceil -\log G \rceil + 1 = 151 \text{ бит.}$$

При таком подходе, если n устремить к бесконечности, получим, что избыточность будет минимальной достижимой, а скорость кода будет близка к энтропии