Задание 2

И. Герасимов

1 Использование программы

```
python3.8 -f FILE [-r R] [-e EPSILON] [-q Q]
```

Наличие всех параметров R, EPSILON и Q запускает второй режим. Отсутствие хотя бы одного запускает первый режим.

2 Обновление формата для описания источника с памятью

Предлагается следующее (реализуется в задании 2):

В source указывать не строку, а словарь, например:

```
{
  ``switch'': switch_1,
  ``input'': [
    {
     ``code'': [`0'],
     ``next'': 0
    }
  ]
}
```

- По ключю switch указывается имя переключателя;
- input - список возможных значений памяти. Состоит из словарей;
- По ключю code указывается предыдущее требуемое сообщение - чему должна быть равна память.
- По ключю next указывается номер элемента в списке source, который должен обрабатываться, если память равна содержимому по ключю code.

Для источников без памяти input равен пустому списку.

3 Равномерное кодирование стационарных источников без памяти

По теореме о прямом кодировании первое же требование: $H(X) \leq R$. Однако теперь есть ограничение на мощность алфавита кодера. То есть даже если прямая теорема верна и мы получим существующее кодирование, то мощность алфавита кодера может превысить требуемые значения.

Расмотрим вероятность принадлежности последовательности высоковероятному множеству:

$$Pr\left[(x^{(1)}, \ldots, x^{(n)}) \notin T_n(\delta)\right] \leq \epsilon$$

Обозначения используются в соответствии с заданием, в слайдах δ и ϵ имеют противоположные значения.

По закону больших чисел в форме Чебышева:

$$Pr\left[\left|\frac{1}{n}\sum_{i=1}^{n} I\left(x^{(i)}\right) - H(X)\right| \geq \delta\right] \leq \frac{\mathrm{D}I(X)}{n \cdot \delta^2}$$

То есть мы можем ограничить:

$$\frac{\mathrm{D}I(X)}{n \cdot \delta^2} \leq \epsilon$$

Или:

$$\delta \geq \sqrt{\frac{\mathrm{D}I(X)}{n \cdot \epsilon}}$$

Из ограничения на мощность алфавита кодера и теорему о прямом кодировании получаем:

$$(H(X) + \delta) \leq \frac{q}{n} \leq R$$

Чем больше δ тем больше отклонение от энтропии и больше мощность $T_n(\delta)$.

Поэтому зададим наименьшее δ, чтобы минимизировать $|T_n(\delta)|$.

$$\delta = \sqrt{\frac{\mathrm{D}I(X)}{n \cdot \epsilon}}$$

Тогда из неравенств на мощность кода получим неравенства для q:

$$n\left(H(X) + \sqrt{\frac{\mathrm{D}I(X)}{n \cdot \epsilon}}\right) \leq q \leq R \cdot n$$

Этот отрезок не будет пустым, если:

$$n \geq \frac{\mathrm{D}I(X)}{\delta \cdot (R - H(X))^2}$$

Тогда возьмем n начиная с $max\left(\frac{\mathrm{D}I(X)}{\delta\cdot(R-H(X))^2}, \frac{q}{R}\right)$ и будем считать $T_n(\delta)$, и проверять мощность.

Однако здесь вычисления проводится относительно оценки Чебышева, которая может оказаться слишком грубой. Поэтому дополнительно реализована итерация по $n : H(X) \leq \frac{q}{n} \leq R$. Если мы получаем код, мощность которого превышает допустимое значение, то процесс повторяется с уменьшенным δ. Если снова, то идет переход к следующему n. Далее будет пример, когда при описанном выше методе не получен код, но по итерации было построено $T_n(\delta)$, мощность которого удовлетворяет условию, а вычисленная вероятность ошибки меньше требуемой (поскольку источник стационарный и без памяти, вероятность вычислима по модели).

4 Пример 1.1. Стационарный процесс без памяти

Файл источника есть station_zero.json. Запуск программы с параметрами $q = 10, R = 0.7, eps = 0.4$ приводит к следующему выводу:

```
Processing file...
Done.
Executing second mode..
source entropy: 0.32346243587218526
alphabet size: 2
Dispersion: 0.8862655640991078
Found n: 16
coded set length: 137
true epsilon: 0.06460036990180273
```

Результат кодирования можно найти в station_zero.code

5 Пример 1.2 Уменьшение Epsilon

Возьмем тот же источник, но теперь $eps = 0.2$. Получаем следующее:

```
Processing file...
Done.
Executing second mode..
source entropy: 0.32346243587218526
alphabet size: 2
Dispersion: 0.8862655640991078
Cannot find necessary value in base calculation. Perform iteration.
Found n: 15
true epsilon: 0.05481875775398026
coded set length: 121
```

То есть итерационный метод дал положительный результат.

6 Пример 1.3 Очередное уменьшение Epsilon

Для того же источника берем $eps = 0.05$. Получаем следующее:

```
Processing file...
Done.
Executing second mode..
source entropy: 0.32346243587218526
alphabet size: 2
Dispersion: 0.8862655640991078
Cannot find necessary value in base calculation. Perform iteration.
n = 15; unable to achieve condition for code. Selecting smaller data.
n = 16; unable to achieve condition for code. Selecting smaller data.
Found n: 16
true epsilon: 0.012431771780082146
coded set length: 697
```

То есть мы перешли к итерации, на первом n не смогли найти код с нужной мощностью алфавита. Для следующего n попытка уменьшить δ привела к успеху с ошибкой кодирования меньше требуемой.

7 Пример 2. Цепь Маркова, эргодичность, нестационарность

Рассмотрим Марковский процесс. Если у цепи есть финальное распределение вероятностей (определяемое уравнениями Колмогорова и условием нормировки, то цепь является эргодической.

Зададим цепь Маркова из 3 состояний. Тогда матрица перехода имеет следующий вид:

$$\begin{array}{ccc} q_1 & q_2 & 1-q_1-q_2 \\ r_1 & r_2 & 1-r_1-r_2 \\ t_1 & t_2 & 1-t_1-t_2 \end{array}$$

где

$q_1 + q_2 \leq 1,$
$r_1 + r_2 \leq 1,$
$t_1 + t_2 \leq 1,$
$q_i \geq 0, r_i \geq 0, t_i \geq 0.$

Для финальных вероятностей нужно решить систему $\bar{p} = \bar{p}A$:

$$\begin{cases} p_1 = p_1 \cdot q_1 + p_2 \cdot r_1 + p_3 \cdot t_1 \\ p_2 = p_1 \cdot q_2 + p_2 \cdot r_2 + p_3 \cdot t_2 \\ p_3 = p_1 \cdot (1-q_1-q_2) + p_2 \cdot (1-r_1-r_2) + p_3 \cdot (1-t_1-t_2) \\ p_1 + p_2 + p_3 = 1 \end{cases}$$

Зададим $q_1 = 0.05, r_1 = 0.1, t_1 = 0.15, q_2 = 0.85, r_2 = 0.8, t_2 = 0.75$

Тогда:

$$\begin{cases} p_1 = p_1 \cdot 0.05 + p_2 \cdot 0.1 + p_3 \cdot 0.15 \\ p_2 = p_1 \cdot 0.85 + p_2 \cdot 0.8 + p_3 \cdot 0.75 \\ p_3 = p_1 \cdot 0.1 + p_2 \cdot 0.1 + p_3 \cdot 0.1 \\ p_1 + p_2 + p_3 = 1 \end{cases}$$

Решением будет: $p_1 = p_3 = 0.1, p_2 = 0.8$.

Распишем вероятности для символов, выдаваемых источником для следующих моделей:

Монета 1:

$$\begin{cases} P(x=0) = 0.3 \\ P(x=1) = 0.5 \\ P(x=2) = 0.2 \end{cases}$$

Монета 2:

$$\begin{cases} P(x=0) = 0.05 \\ P(x=1) = 0.9 \\ P(x=2) = 0.05 \end{cases}$$

Монета 3:

$$\begin{cases} P(x=0) = 0.1 \\ P(x=1) = 0.85 \\ P(x=2) = 0.05 \end{cases}$$

Получаем:

$$\begin{cases} P(x=0) = 0.3 \cdot 0.1 + 0.05 \cdot 0.8 + 0.1 \cdot 0.1 = 0.08 \\ P(x=1) = 0.5 \cdot 0.1 + 0.9 \cdot 0.8 + 0.85 \cdot 0.1 = 0.855 \\ P(x=2) = 0.2 \cdot 0.1 + 0.05 \cdot 0.8 + 0.05 \cdot 0.1 = 0.065 \end{cases}$$

Для параметров $q = 10, R = 0.9, eps = 0.4$ получаем (можно использовать файл markov.json, можно использовать файл markov_final.json):

```
Processing file...
Done.
Executing second mode..
source entropy: 0.74106370786811
alphabet size = 3
Dispersion = 1.5674956910994595
Cannot find necessary value in base calculation. Perform iteration.
n = 12; unable to achieve condition for code. Selecting smaller delta.
Found n = 12
true epsilon = 0.3997227784071269
coded set length: 288
```

Пример с эргодическим источником (который является марковской цепью) приводится теоретически с практической реализацией равномерного кода.