Лабораторная работа № 11
СЖАТИЕ/РАСПАКОВКА ДАННЫХ АРИФМЕТИЧЕСКИМ МЕТОДОМ

Цель: приобретение практических навыков использования арифметических методов сжатия/распаковки данных.

Задачи:

1. Закрепить теоретические знания по алгебраическому описанию и использованию арифметических методов сжатия/распаковки (архивации/разархивации) данных.
2. Разработать приложение для реализации арифметических методов.
3. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

11.1. Теоретические сведения

11.1.1. Описание и основные свойства арифметических методов

Как мы установили, вероятностные (префиксные) методы являются достаточно простыми и эффективными: они основаны на использовании кодов переменной длины и для вероятностей появления символов алфавита, кратных степеням числа 2 (1/2, 1/4, 1/8 и т. п.), дают наилучшие результаты. При других значениях вероятностей, как правило, самый короткий код получается большим, чем двоичный логарифм этой вероятности (взятый с отрицательным знаком). Например, при $p(a_i)$ = 0,4 получим $-\log_2 0{,}15 = 1{,}32$. Понятно, что мы не можем закодировать этот символ только 2 битами (либо одним), т. е. решение не всегда является оптимальным. Анализируемого недостатка лишены арифметические методы.

При арифметическом сжатии (кодировании) текст представляется вещественными числами в интервале от 0 до 1. По мере анализа текста отображающий его интервал уменьшается, а количество битов для его представления возрастает. Очередные символы

текста сокращают величину интервала, исходя из значений соответствующих вероятностей.

Основная идея арифметического метода сжатия заключается в том, чтобы присваивать коды не отдельным символам, а их последовательностям.

Таким образом, как и во всех энтропийных алгоритмах, исходной является информация о частоте встречаемости каждого символа алфавита.

Алгоритмы прямого и обратного преобразований базируются на операциях с «*рабочим отрезком*».

Рабочим отрезком называется интервал [*a*; *b*] с расположенными на нем точками. Причем точки расположены таким образом, что длины образованных ими отрезков пропорциональны (или равны) частоте (вероятности) появления соответствующих символов.

11.1.2. Описание примера и особенности методов

Дана последовательность X_k = «*молоко*». Рассчитываем статистику:

$$p(м) = 0{,}1666$$
$$p(л) = 0{,}1666$$
$$p(к) = 0{,}1666$$
$$p(о) = 0{,}5$$

Шаг 0: строим *основной рабочий отрезок*, как показано на рис. 11.1.

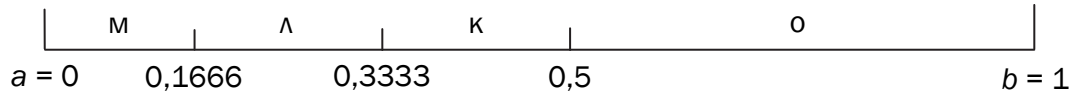

Рис. 11.1. Разметка и точки основного рабочего отрезка

Как показано на рис. 11.1, при инициализации алгоритма $a = 0$, $b = 1$ ($a_0 = 0$, $b_0 = 1$).

Прямое преобразование (сжатие). Один шаг сжатия (кодирования) заключается в простой операции: берется кодируемый символ, для него ищется соответствующий участок на рабочем отрезке. Найденный участок становится новым рабочим отрезком. Его тоже необходимо разбить с помощью точек.

Это и последующие разбиения отрезка (на шаге i) подразумевают определение новых значений верхней (H_i) и нижней (L_i) границ для всего участка и осуществляются по следующим правилам:

$$\left.\begin{aligned} H_i(\alpha_j) &= L_{i-1} + (H_{i-1} - L_{i-1}) \cdot H(\alpha_j)_0; \\ L_i(\alpha_j) &= L_{i-1} + (H_{i-1} - L_{i-1}) \cdot L(\alpha_j)_0, \end{aligned}\right\} \qquad (11.1)$$

где α_j – j-й символ сжимаемой последовательности, L_{i-1} и H_{i-1} – соответственно нижняя и верхняя границы рабочего отрезка на $(i-1)$-м шаге, $L(\alpha_j)_0$ и $H(\alpha_j)_0$ – соответственно исходные нижняя и верхняя границы символа α_j.

Шаг 1: в нашем примере на первом шаге берется первый символ последовательности: «*м*» (α_1 = «*м*») и ищется соответствующий участок на рабочем отрезке. Легко понять по рис. 11.2, что этот участок соответствует интервалу [0; 0,1666]. Он становится новым рабочим отрезком и опять разбивается согласно статистике и соотношениям (11.1):

$$H_1(м) = L_0 + (H_0 - L_0) \cdot H(\alpha_j = м)_0 = \\ = 0 + (0{,}1666 - 0) \cdot 0{,}1666 = 0{,}0277;$$

$$L_1(м) = L_0 + (H_0 - L_0) \cdot L(\alpha_j = м)_0 = \\ = 0 + (0{,}1666 - 0) \cdot 0 = 0;$$

$$H_1(л) = L_0 + (H_0 - L_0) \cdot H(\alpha_j = л)_0 = \\ = 0 + (0{,}1666 - 0) \cdot 0{,}3333 = 0{,}055555;$$

$$L_1(л) = L_0 + (H_0 - L_0) \cdot L(\alpha_j = л)_0 = \\ = 0 + (0{,}1666 - 0) \cdot 0{,}1666 = 0{,}0277.$$

После выполнения всех вычислений на первом шаге получим разметку рабочего отрезка ($a_1 = L_1 = 0$, $b_1 = H_1 = 0{,}1666$) в соответствии с рис. 11.2.

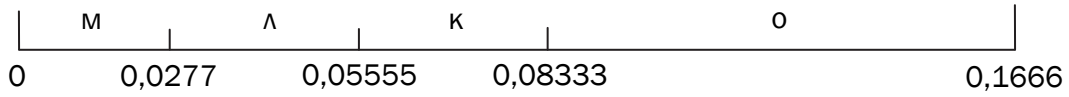

Рис. 11.2. Разметка и точки рабочего отрезка после анализа первого символа (*м*) последовательности «*молоко*»

Шаг 2: анализируем следующий символ входной последовательности (*о*). На новом рабочем отрезке этому символу соответствует интервал [$a_2 = L_2 = 0{,}083333$; $b_2 = H_2 = 0{,}1666$]. Для следующего шага он станет рабочим отрезком (рис. 11.3).

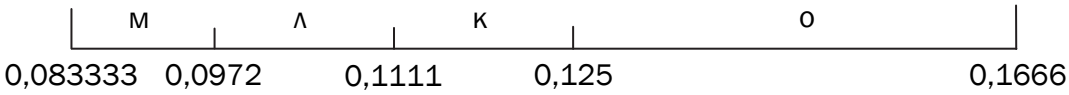

Рис. 11.3. Рабочий отрезок после анализа первых двух символов

Строго говоря, на первом шаге не было необходимости рассчитывать новые границы для каждого символа. Как видим, это только увеличивает время анализа. Достаточно было определить новые верхнюю и нижнюю границы лишь для очередного (на шаге 2) символа (*о*). Мы здесь привели расчеты только для того, чтобы лучше понять процесс вычислений.

Шаг 3: поскольку очередным из анализируемых символов будет буква «*л*», достаточно в соответствии с выражениями (11.1) определить новые границы:

$$H_3(л) = L_2 + (H_2 - L_2) \cdot H(\alpha_j = л)_0 =$$
$$= 0{,}08333 + (0{,}1666 - 0{,}08333) \cdot 0{,}3333 = 0{,}1111;$$
$$L_3(л) = L_2 + (H_2 - L_2) \cdot L(\alpha_j = л)_0 =$$
$$= 0{,}08333 + (0{,}1666 - 0{,}08333) \cdot 0{,}1666 = 0{,}972.$$

Новый рабочий отрезок определим как интервал [$a_3 = L_3 =$ $= 0{,}0972$; $b_3 = H_3 = 0{,}1111$], так как он соответствует символу «*л*». Вид разметки рабочего отрезка представлен на рис. 11.4.

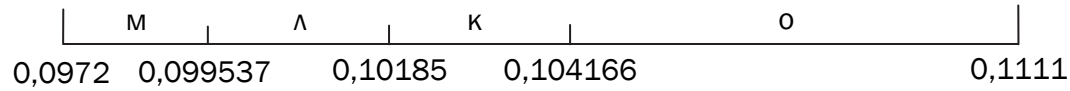

Рис. 11.4. Рабочий отрезок с разметками после анализа последовательности «*мол*»

Продолжаем анализировать символы сообщения. Находим на рабочем отрезке интервал [0,104166; 0,1111], который соответствует очередному символу: «*о*». Производим разметку нового рабочего отрезка, как показано на рис. 11.5.

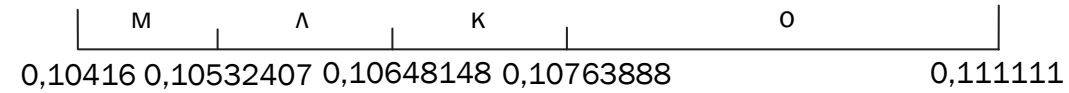

Рис. 11.5. Разметка рабочего отрезка на четвертом шаге

Следующий символ последовательности – «*к*». На рабочем отрезке данному символу принадлежит интервал [0,10648148;

0,10763888]. Новый рабочий отрезок имеет вид, как показано на рис. 11.6.

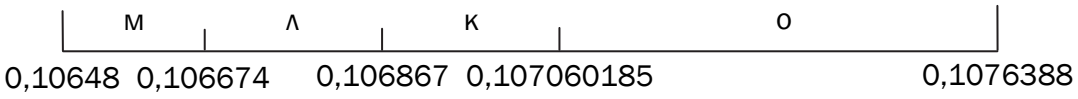

Рис. 11.6. Разметка рабочего отрезка на пятом шаге прямого преобразования

У нас остался последний символ: «*о*». Результатом кодирования цепочки символов является любое число с итогового рабочего отрезка [0,107060185; 0,10763888]. Обычно таким числом является *нижняя граница указанного отрезка*. Поступим и мы в соответствии с данным принципом. Возьмем число с меньшим количеством знаков после запятой.

Таким образом, итогом сжатия входной последовательности «*молоко*» будет число 0,107060185 (для упрощения дальнейших вычислительных операций округлим его до **0,1071**).

Обратное преобразование (декомпрессия). Для восстановления исходного сообщения необходима информация:

- о значении числа, являющегося итогом сжатия сообщения (в нашем случае 0,1071);
- количестве символов в сжатом сообщении;
- вероятностных параметрах всех символов исходного сообщения (таблица вероятностей).

Как и при сжатии, вначале необходимо начальный рабочий отрезок [0; 1) разбить на интервалы, длины которых равны вероятностям появления соответствующих символов, т. е. создать рабочий отрезок, полностью соответствующий рис. 11.1.

Анализ будем проводить с использованием конкретных данных, взятых из последнего примера.

Итак, в качестве исходного участка для обратного преобразования принимается исходный участок для прямого преобразования с одинаковыми точками его разбиения. Ниже он повторяется.

На каждом шаге обратного преобразования выбираем отрезок, в который попадает текущее число (код). Символ, который соответствует данному отрезку, является очередным символом восстановленного (распакованного) сообщения.

В общем случае код символа, восстанавливаемого на шаге i, вычисляется соотношением:

$$\text{код } i = [\text{код } (i-1) - L(\alpha_{i-1})_0] / [H(\alpha_{i-1})_0 - L(\alpha_{i-1})_0], \qquad (11.2)$$

где код $(i-1)$ – число, анализ которого производился на предыдущем шаге – $(i-1)$-м; $H(\alpha_{i-1})_0$ и $L(\alpha_{i-1})_0$ – соответственно верхняя и нижняя исходные границы символа сообщения, восстановленного на предыдущем шаге.

Шаг 1: определяем интервал, в который попадает начальное число: 0,1071 (код 1 = 0,1071). Это интервал [0; 0,166666] и ему соответствует символ «*м*». Данный интервал становится новым рабочим отрезком, а первый символ восстановленного сообщения – «*м*».

Шаг 2: производим вычисление в соответствии с выражением (11.2):

$$\text{код } 2 = [\text{код } 1 - L(\alpha_1 = м)_0] / [H(\alpha_1 = м)_0 - L(\alpha_1 = м)_0] =$$
$$= (0{,}1071 - 0) / (0{,}1666 - 0) = 0{,}643.$$

Вычисленный код соответствует символу «*о*».

Шаг 3: выполняем известное вычисление:

$$\text{код } 3 = [\text{код } 2 - L(\alpha_2 = о)_0] / [H(\alpha_2 = о)_0 - L(\alpha_1 = м)_0] =$$
$$= (0{,}643 - 0{,}5) / (1 - 0{,}5) = 0{,}286.$$

Полученное число соответствует символу «*л*».

Аналогичным образом производятся и последующие вычислительно-аналитические операции.

Другой вариант алгоритма распаковки «сжатого» сообщения основан на *свойстве рекуррентности прямого преобразования*. Производя на каждом шаге вычисление новых границ рабочего участка в соответствии с восстановленным на данном шаге символом (наподобие тех вычислений, которые мы выполняли при сжатии) и анализируя, на какой из отрезков этого участка попадает входное число (в нашем примере это 0,1071), восстанавливаем исходное сообщение.

Таким образом, на первом шаге это число соответствует букве «*м*», которая попадает в интервал [0; 0,1666]. Этот новый диапазон будет рабочим участком на шаге 2.

Так как нам известно, что символов в сообщении было 6, то на последнем шаге необходимо только определить недостающий символ. Таким символом становится символ «*о*». Полученное сообщение «*молоко*» и есть восстановленная («распакованная») последовательность.

Другие примеры и иные графические отображения результатов вычисления рабочих отрезков можно найти в пособии [5].

К числу основных особенностей методов можно отнести следующие:

- проанализированный алгоритм сжатия (кодирования) ничего не передает до полного завершения анализа всего текста;
- обычно результат представляется в формате целочисленной арифметики;
- требуемая для представления интервала $[H_i; L_i]$ точность возрастает вместе с длиной анализируемого текста; постепенное выполнение алгоритма помогает преодолеть эту проблему, при этом необходимо внимательно следить за возможностью переполнения [34]; упомянутые границы текущего интервала могут также представляться в виде целых чисел;
- как мы видим, арифметическое кодирование «работает» при помощи масштабирования или нормализации накопленных вероятностей, поставляемых моделью в интервалах $[H_i; L_i]$ для каждого передаваемого символа; если предположить, что H_i и L_i настолько близки друг к другу, что операция масштабирования «приводит» разные символы сообщения к одному целому числу, входящему в $[H_i; L_i]$, то дальнейшее кодирование продолжать невозможно. Следовательно, программа-кодировщик должна следить за тем, чтобы интервал $[H_i; L_i]$ не «слипался».

Особенности программной реализации метода, его качественные и количественные характеристики достаточно подробно описаны в [34].

11.2. Практическое задание

1. Разработать авторское приложение в соответствии с целью лабораторной работы.

2. С помощью приложения выполнить прямое и обратное преобразования сообщений в соответствии с таблицей.

Каждый студент выполняет задание, состоящее из двух частей. Первая часть предусматривает кодирование/декодирование сообщения, указанного в 2-м столбце, вторая часть – составного сообщения, полученного конкатенацией последовательностей из 2-го столбца, указанных в 3-м столбце. Например, для варианта

№ 1 такой конкатенацией будет последовательность «летоисчислениевремяпрепровождение».

Варианты заданий

Вариант	Сжимаемое сообщение	Строки из столбца 2
1	летоисчисление	1-2
2	времяпрепровождение	2-3
3	мультимиллионер	3-4
4	семенохранилище	4-5
5	песнетворчество	5-6
6	электрифицированный	6-7
7	водоворотоподобный	7-8
8	самообороноспособность	8-9
9	малосимпатичный	9-10
10	достопримечательность	10-11
11	сорокадневный	11-12
12	телегаммааппарат	12-13
13	полуторапроцентный	13-14
14	гидроаэроионизация	14-15
15	экстраординарный	15-1

3. Дать оценку возможности переполнения при выполнении вычислений.

4. Сравнить характеристики арифметического сжатия с вероятностными алгоритмами.

5. Результаты оформить в виде отчета по установленным правилам.

ВОПРОСЫ ДЛЯ КОНТРОЛЯ И САМОКОНТРОЛЯ

1. В чем заключается суть арифметического метода сжатия?

2. Пояснить принцип определения границ отрезков на каждом шаге (на каждом рабочем интервале) прямого и обратного преобразования.

3. В чем заключаются основные отличия метода арифметического сжатия от метода Хаффмана?

4. Какой недостаток, свойственный префиксным методам, отсутствует в арифметическом кодировании?

5. Насколько эффективен арифметический метод для кодирования данных с равномерно распределенными вероятностями появления символов?

6. Какие требования предъявляет арифметический метод к его программной реализации?

7. Какие сложности возникают при реализации алгоритма в программном коде? Как можно их решить?

8. Осуществить сжатие и распаковку сообщений арифметическим методом:

а) «мама мыла раму»;

б) «насос»;

в) «университет»;

г) «шалаш»;

д) «информатика»;

е) «246824328»;

ж) «101101001011010»;

з) собственные фамилия, имя, отчество.