МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ 8_2

по дисциплине

«Структуры и алгоритмы обработки данных»

Тема. Кодирование и сжатие данных методами без потерь.

Выполнил студент группы ИКБО-60-23 Шеенко В.А

Принял старший преподаватель Скворцова Л.А.

Москва 2024

СОДЕРЖАНИЕ

1 ЦЕЛЬ РАБОТЫ

Получение практических навыков и знаний по выполнению сжатия данных рассматриваемыми методами.

ЗАДАНИЕ №1

2.1 Постановка задачи

Применение алгоритма группового сжатия текста RLE Сжать текст, используя метод RLE (run length ecoding/кодирование длин серий/групповое кодирование).

Требования к выполнению заданию

1) Описать процесс сжатия алгоритмом RLE.

2) Придумать текст, в котором есть длинные (в разумных пределах) серии из повторяющихся символов. Выполнить сжатие текста. Рассчитать коэффициент сжатия.

3) Придумать текст, в котором много неповторяющихся символов и между ними могут быть серии. Выполнить групповое сжатие, показать коэффициент сжатия. Применить алгоритм разделения текста при групповом кодировании, позволяющий повысить эффективность сжатия этого текста. Рассчитать коэффициент сжатия после применения алгоритма.

2.2 Ход решения

2.2.1 Теоретическая часть

RLE (Run-Length Encoding) — это один из простейших алгоритмов сжатия данных. Его суть заключается в замене последовательностей одинаковых символов (так называемых серий) на более компактное представление, указывающее символ и количество его повторений. Например, строка AAAAABBBCC преобразуется в 5A3B2C.

Алгоритм особенно эффективен при обработке данных, содержащих длинные повторяющиеся последовательности, таких как изображения с большими однотонными областями или текстовые данные, где встречаются серии одинаковых символов.

К достоинствам алгоритма относится простота реализации и скорость работы. Однако эффективность сжатия сильно зависит от структуры данных:

для текста с малым количеством серий размер после сжатия может даже увеличиться.

2.2.2 Экспериментальная часть

Для примера возьмем следующую строку:

XXXXXXXXXXYYYYYYYYZZZZZZZWWWWWWWW

В этой строке явно видны длинные серии из символов `X`, `Y`, `Z` и `W`. Применяя RLE, получаем:

10X8Y7Z8W

Здесь каждая серия заменяется на число, указывающее длину серии, и сам символ.

Исходная длина строки: 34 символа.

Длина сжатой строки: 12 символов.

Коэффициент сжатия:

Коэффициент сжатия - отношение длины исходного сообщения к выходному коду.

$$\frac{34}{12} = 2{,}8\bar{3}$$

Кроме того, проверим эффективность данного алгоритма сжатия для символьной последовательности, где не так много повторяющихся символов, идущих подряд:

AXBYCZAAWWRRQQTTZZZMMNLLKJHHHHGGFEEE

Разобьем этот текст на последовательности:

AXBYCZAAWWRRQQTTZZZMMNLLKJHHHHGGFEEE

Снова воспользуемся алгоритмом RLE:

1A1X1B1Y1C1Z2A2W2R2Q2T3Z2M1N2L1K1J4H2G1F3E

Исходная длина строки: 36 символа.

Длина сжатой строки: 42 символов.

Коэффициент сжатия:

$$K = \frac{36}{42} = 0{,}857$$

RLE в данном случае оказался неэффективным, так как одиночные символы увеличивают объем данных при добавлении чисел.

Повышение эффективности: разделение текста.

Идея разделения текста заключается в том, чтобы повысить эффективность сжатия путем обработки только тех участков текста, где применение RLE действительно выгодно. Это особенно важно для данных, где преобладают одиночные символы, которые увеличивают объем после кодирования.

AXBYCZAAWWRRQQTTZZZMMNLLKJHHHHGGFEEE

Преобразуется в:

AXBYCZ2A2W2R2Q2T3Z2MN2LKJ4H2GF3E

Исходная длина строки: 36 символа.

Длина сжатой строки: 32 символов.

Коэффициент сжатия:

$$K = \frac{36}{32} = 1{,}125$$

Для текста с большим количеством одиночных символов RLE часто приводит к увеличению объема. Чтобы сделать сжатие эффективным, стоит выделять серийные участки и обрабатывать их отдельно. Такой подход позволяет сократить избыточные данные без потери информации.

ЗАДАНИЕ №2

3.1 Постановка задачи

Исследование алгоритмов группового сжатия (методы Лемпеля –Зива: LZ77, LZ78) на примерах. Тексты для сжатия по вариантам в табл. 1 в столбце 1 (для LZ77) и в столбце 2 (для LZ78).

Требования к выполнению заданию

1) Выполнить каждую задачу варианта задания, представив алгоритм решения в виде таблицы и указав результат сжатия. Примеры оформления решения в отчете представлены в Приложении1 этого документа.

2) Описать процесс восстановления сжатого текста.

Таблица 1. Тексты для сжатия

Вариант	Сжатие данных по методу Лемпеля–Зива LZ77 Используя двухсимвольный алфавит (0, 1) закодировать следующую фразу:	Закодировать следующую фразу, используя код LZ78	Закодировать фразу методами Шеннона– Фано
28	1000010101001101000	ceaceraceaerceaercea	Плыл по морю чемодан, В чемодане был диван, На диване ехал слон. Кто не верит – выйди вон!

3.2 Ход решения

Методы Лемпеля–Зива (LZ77 и LZ78) являются основой для многих современных алгоритмов сжатия, включая ZIP и PNG. Они относятся к группе алгоритмов словарного сжатия, которые используют ранее встреченные фрагменты текста для кодирования.

3.2.1 LZ77

Таблица 2. Процесс сжатия LZ77

Исходный текст	1 00 001 01 010 011 0100 0
LZ-код	1.00.101.001.1000.1001.1010.0
R	2 3
Вводимые коды	- 10 11 100 101 110 111

Таблица 3. Таблица комбинаций и соответствующих кодов

Комбинация	Код
0	00
1	01
00	10
001	11
01	100
010	101
011	110
0100	111

Для восстановления данных нужно произвести процесс сжатия в обратном порядке.

Сжатая часть	Код	Комбинация	Исходная	Новый код для комбинации
1	1	1	1	-
00	0	0	00	10
101	10	00	001	11
001	00	0	01	100
1000	100	01	010	101
1001	100	01	011	110
1010	101	010	0100	111
000	000	0	0	-

3.2.2 LZ78

ceaceraceaerceaercea

Содержимое словаря		Содержимое считанного	Код
	1	c	<0, c>
c	2	e	<0, e>
c e	3	a	<0, a>
c e a	4	ce	<1, e>
c e a ce	5	r	<0, r>
c e a ce r	6	ac	<3, e>
c e a ce r ac	7	ea	<2, a>
c e a ce r ac ea	8	ae	<3, e>
c e a ce r ac ea ae	9	rc	<5, c>
c e a ce r ac ea ae rc	10	eae	<7, a>
c e a ce r ac ea ae rc eae	11	rce	<9, e>
c e a ce r ac ea ae rc eae rce	12	a	<0, a>

Результат: 0c0e0a1e0r3e2a3e5c7a9e0a

Восстановление данных идет в обратном порядке

Ввод	№	Исходная	Строка
<0, c>	1	c	c
<0, e>	2	e	ce
<0, a>	3	a	cea
<1, e>	4	ce	ceace
<0, r>	5	r	ceacer
<3, e>	6	ae	ceacerae
…	…	…	…

ЗАДАНИЕ №3

4.1 Постановка задачи

Изучить методы сжатия данных, а именно, алгоритмов Шеннона-Фано и Хаффмана, с применением их на практике для сжатия текстовой информации. Для выполнения задания будут использованы два различных метода сжатия: метод Шеннона-Фано для сжатия текста, представленного в таблице 1, и метод Хаффмана для сжатия строки, полученной из фамилии, имени и отчества.

4.2 Ход решения

4.2.1 Метод Шеннона-Фано

Текст:

Плыл по морю чемодан,
В чемодане был диван,
На диване ехал слон.
Кто не верит – выйди вон!

Алгоритм формирования префиксного дерева:

1. Для каждого символа строки вычисляется его частота появления.
2. Алфавит сортируется по убыванию частоты символов.
3. Разбиваем отсортированные символы на две группы, так чтобы сумма частот символов в каждой группе была как можно более равной.
4. Процесс деления повторяется рекурсивно для каждой группы до тех пор, пока не будут получены отдельные символы.

Таблица

Символ	Количество	Код	Бит
пробел	17	00	34
н	8	1110	32
о	7	1011	28

е	7	1100	28
а	6	1010	24
в	6	1001	24
л	5	0110	20
д	5	0111	20
и	4	11011	20
ы	3	10000	15
м	3	11010	15
п	2	01000	10
р	2	111111	12
ч	2	111110	12
,	2	100011	12
т	2	111100	12
ю	1	100010	6
б	1	010111	6
х	1	010110	6
с	1	1111010	7
.	1	1111010	7
к	1	010101	6
-	1	010100	6
й	1	010011	6
!	1	010010	6

После кодирования получим:

0100001101000001100001000101100110101011111111100010001111101100110101101111010111010001110010011111011001101010110111010111101000011100001011110000011000011111011100110101110100011111010100001111101110011010111011000011000101101010011000111101101101011111011110100101011111001011001110110000100111001111111101111110000010100001001100000100110111111011001001101111100100010

Всего символов – 368

В кодировке ASCII – 8 * 90 = 720

Коэффициент сжатия:

$$K = \frac{720}{368} = 1{,}96$$

Алгоритм декодирования:

- Декодирование происходит путем чтения битов сжатого текста и перехода по префиксному дереву.
- Каждый символ восстанавливается, когда мы достигаем листа дерева.

4.2.2 Метод Хаффмана

Задача состоит в сжатии строки, полученной из полного имени ("Шеенко Владимир Александрович"), с использованием метода Хаффмана. Метод Хаффмана строит оптимальное префиксное дерево для минимизации длины кода с учетом вероятностей появления символов.

Алгоритм формирования префиксного дерева:

- Подсчитываем частоту появления каждого символа.
- Сортируем символы по частоте.
- Строим дерево Хаффмана, начиная с двух символов с наименьшими частотами, соединяя их в новый узел.
- Повторяем процесс до тех пор, пока не останется только один узел (корень дерева).

Символ	Количество	Вероятность	Код
и	3	10,34%	010
а	3	10,34%	001
е	3	10,34%	000
пробел	2	6,9%	1111
о	2	6,9%	1110
л	2	6,9%	1101
в	2	6,9%	1100

к	2	6,9%	1001
н	2	6,9%	1000
р	2	6,9%	0111
д	2	6,9%	0110
м	1	3,45%	10111
ш	1	3,45%	10110
ч	1	3,45%	10101
с	1	3,45%	10100

Результат:

101100000001000100111101111110011010010110010101110100111111100111
0100010011010000110000110011111101100010101 01

Всего символов – 111

В кодировке ASCII – 8 * 29 = 232

Коэффициент сжатия:

$$K = \frac{232}{111} = 2{,}09$$

Декодирование строки выполняется с использованием дерева Хаффмана: читаем биты сжатого текста и переходим по дереву, восстанавливая исходные символы.

ВЫВОД

Получили практические навыки и знания по выполнению сжатия данных рассматриваемыми методами

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Рысин, М. Л. Введение в структуры и алгоритмы обработки данных : учебное пособие / М. Л. Рысин, М. В. Сартаков, М. Б. Туманова. — Москва : РТУ МИРЭА, 2022 — Часть 2 : Поиск в тексте. Нелинейные структуры данных. Кодирование информации. Алгоритмические стратегии — 2022. — 111 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/310826 (дата обращения: 10.09.2024).
2. ГОСТ 19.701-90. Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения : межгосударственный стандарт : дата введения 1992-01- 01 / Федеральное агентство по техническому регулированию. – Изд. официальное. – Москва : Стандартинформ, 2010. – 23 с