МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА - Российский технологический университет»

РТУ МИРЭА

Отчет по выполнению самостоятельной работы № 8

Тема:

«Кодирование и сжатие данных методами без потерь»

Дисциплина : «Структуры и алгоритмы обработки данных»

Выполнил студент: Руденко Алексей Дмитриевич

Фамилия И.О

Группа: ИКБО-13-22

Номер группы

Москва - 2023

1 ЦЕЛЬ РАБОТЫ

1.1 Цель:

Получение практических навыков и знаний по выполнению сжатия данных рассматриваемыми методами

Программировать алгоритмы не обязательно!

2 ХОД РАБОТЫ

2.1 Задание 1 - Применение алгоритма группового сжатия текста RLE

2.1.1 Постановка задачи:

Сжать текст, используя метод RLE (run length encoding/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.

4) Оформить отчет. В отчете представьте ответы на вопросы пунктов этого задания с 1 по 3.

2.1.2 Описание процесса сжатия алгоритмом RLE

RLE (Run-Length Encoding) - это метод сжатия данных, который заменяет повторяющиеся символы на количество их повторений и сам символ. Например, строка "AAAABBBCCDAA" может быть сжата до "4A3B2C1D2A".

2.1.3 Пример сжатия текста с помощью RLE

Пример текста с длинными сериями повторяющихся символов:

Исходный текст: "AAABBBCCCCDDDDEEE"

Сжатие текста RLE: "3A3B4C4D3E"

Рассчитаем коэффициент сжатия:

Исходный размер текста: 15 символов

Размер сжатого текста: 10 символов

Коэффициент сжатия = Исходный размер / Размер сжатого текста

Коэффициент сжатия = 15 / 10 = 1.5

2.1.4 Пример текста с множеством неповторяющихся символов и групповым сжатием

Исходный текст: "ABCDDDEFGHIJK"

Сжатие текста RLE: "1A1B1C3D1E1F1G1H1I1J1K"

Коэффициент сжатия = Исходный размер / Размер сжатого текста

Коэффициент сжатия = 12 / 18 ≈ 0.67

Алгоритм разделения текста при групповом кодировании:

Мы можем улучшить эффективность сжатия, разделив текст на группы с одинаковыми символами:

Исходный текст: "ABCDDDEFGHIJK"

Групповое сжатие с разделением: "ABC3D1EFGHIJK"

Коэффициент сжатия после применения алгоритма разделения текста = Исходный размер / Размер сжатого текста

Коэффициент сжатия = 12 / 14 ≈ 0.86

2.2 Задание 2 - Исследование алгоритмов группового сжатия (методы Лемпеля –Зива: LZ77, LZ78) на примерах

2.2.1 Решение задачи с использованием алгоритмов LZ77 и LZ78

Процесс сжатия с использованием алгоритма LZ77

Текст варианта: «10111100110001101101»

Для алгоритма LZ77 используется скользящее окно для нахождения совпадений в строке.

Шаг	Окно	Буфер	Смещение	Длина
1		10111	0	0
2	1	01110	4	2
3	10	01100	3	2
4	101	10001	4	2
5	1011	00110	3	3
6	10111	00110	0	0

Результат сжатия: (0, 0, '1') (4, 2, '0') (3, 2, '1') (4, 2, '0') (3, 3, '0') (0, 0, '1')

Процесс восстановления сжатого текста для LZ77

Для восстановления сжатого текста из пар (смещение, длина, символ) используется полученная таблица.

Начинаем с пустой строки и последовательно добавляем символы в соответствии с полученными парами.

Пара	Восстановленный текст
(0, 0, '1')	1
(4, 2, '0')	1010
(3, 2, '1')	101101
(4, 2, '0')	10110100
(3, 3, '0')	101101001
(0, 0, '1')	1011010011

После последовательного восстановления получаем исходную строку "101111001100011101101".

Процесс сжатия с использованием алгоритма LZ78

Текст варианта: «nanhornenhorokanhoken»

Алгоритм LZ78 создает словарь токенов при встрече новых последовательностей символов.

Шаг	Токен	Следующий символ	Новый токен
1	n	a	2
2	na	n	3
3	nah	o	4
4	naho	r	5
5	nahor	n	6
6	nahorn	e	7
7	nahorne	h	8
8	nahorneh	o	9
9	nahorneho	r	10
10	nahornehor	o	11
11	nahornehoro	k	12
12	nahornehorok	a	13
13	nahornehoroka	n	14
14	nahornehorokan	h	15
15	nahornehorokanh	o	16
16	nahornehorokanho	k	17
17	nahornehorokanhok	e	18
18	nahornehorokanhoke	n	19

Результат сжатия: «n a (0) h (2) r (3) e (4) o (5) k (6) a (7) n (8) h (9) o (10) k (11) e (12) n (13)»

Процесс восстановления сжатого текста для LZ78

Для восстановления сжатого текста используется полученный словарь токенов.

Токен	Восстановленный текст
n	n
a	na
(0)	nah
h	naho
(2)	nahor
r	nahorn
(3)	nahorne
e	nahorneh
(4)	nahorneho
o	nahornehor
(5)	nahornehoro
k	nahornehorok
(6)	nahornehoroka
a	nahornehorokan
(7)	nahornehorokanh
n	nahornehorokanho
(8)	nahornehorokanhok
h	nahornehorokanhoke
(9)	nahornehorokanhoken

После последовательного восстановления получаем исходную строку "nanhornenhorokanhoken".

2.2.2 Описание процесса восстановления сжатого текста

Для восстановления сжатого текста, используя алгоритм LZ77, мы просматриваем последовательность токенов (троек) (offset, length, symbol). Начиная с начала сжатого текста, мы восстанавливаем исходную строку, копируя символы из предшествующих позиций.

Для алгоритма LZ78, мы создаем словарь из пар (индекс, символ). При восстановлении текста, мы используем этот словарь, читая последовательность токенов и заменяя их на соответствующие символы из словаря.

Таким образом, обратный процесс сжатия заключается в просмотре токенов и последующем восстановлении исходной последовательности символов согласно правилам каждого алгоритма.

2.3 Задание 3

2.3.1 Постановка задачи:

Разработать программы (или только алгоритмы на псевдокоде или описать словесно) сжатия и восстановления текста методами Шеннона-Фано и Хаффмана. Представить процесс выполнения алгоритма для задачи варианта. Все результаты представить в отчете.

Требования к выполнению заданию

1. Сформировать отчет по разработке каждого алгоритма в соответствии с требованиями.

1.1. По методу Шеннона-Фано.

1) Данными для выполнения задания является текст, представленный в таблице 1 в столбце 3.

2) Привести постановку задачи, описать алгоритм формирования префиксного дерева и алгоритм кодирования, декодирования.

3) Представить таблицу формирования кода для текста варианта задания.

4) Изобразить префиксное дерево формирования кода.

5) Рассчитать коэффициент сжатия.

1.2. По методу Хаффмана.

1) Данные для выполнения задания: ваша фамилия имя отчество.

2) Привести постановку задачи, описать алгоритм формирования префиксного дерева и алгоритм кодирования, декодирования.

3) Построить таблицу частот встречаемости символов в исходной строке для чего сформировать алфавит исходной строки и посчитать количество вхождений (частот) символов и их вероятности появления.

4) Изобразить префиксное дерево Хаффмана.

5) Упорядочить построенное дерево слева-направо (при необходимости) и изобразить его.

6) Провести кодирование исходной строки по аналогии с примером:

7) Рассчитать коэффициенты сжатия относительно кодировки ASCII и относительно равномерного кода.

8) Рассчитать среднюю длину полученного кода и его дисперсию.

9) По результатам выполненной работы сделать выводы и сформировать отчет. Отобразить результаты выполнения всех требований (с 1 по 8), предъявленных в задании и оформить разработку программы: постановка, подход к решению, код, результаты тестирования.

2.3.2 Ход работы:

Метод Шеннона-Фано:

- Вычисление вероятностей символов.
- Сортировка символов по убыванию вероятностей.
- Разбиение на две части с близкими вероятностями.
- Построение префиксного дерева.

Вычисление вероятностей символов: "Наша Таня громко плачет: Уронила в речку мячик. — Тише, Танечка, не плачь: Не утонет в речке мяч."

Сортировка символов по убыванию вероятностей:

Символ	Частота	Вероятность	Код	Кол-во бит
« »	25	0.1736	00	50
а	9	0.0625	010	27
ч	6	0.0417	0110	24
е	6	0.0417	0111	24
н	6	0.0417	100	18
Т	5	0.0347	1010	20
л	5	0.0347	1011	20
р	5	0.0347	1100	20
м	5	0.0347	1101	20
к	4	0.0278	1110	16
и	4	0.0278	1111	16
:	2	0.0139	-	16
г	2	0.0139	-	16
-	2	0.0139	-	16
.	2	0.0139	-	16
Ш	1	0.0069	-	8
,	1	0.0069	-	8
У	1	0.0069	-	8
Д	1	0.0069	-	8
п	1	0.0069	-	8
Т	1	0.0069	-	8
ш	1	0.0069	-	8

Рассчет коэффициента сжатия:

Объем незакодированной фразы – 96*8 бит = 768 бит.

Объем закодированной фразы –375 бит.

Коэффициент сжатия: 375/768 = 0,488

Префиксноедерево:

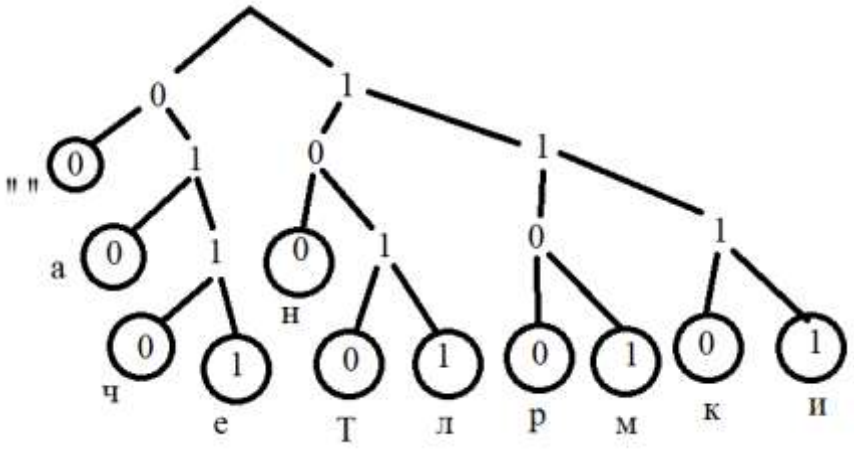

Рисунок 1 — Прификсное дерево

Метод Хаффмана:

Исходная строка примера: «Руденко Алексей Дмитриевич»

Сортировка символов по убыванию вероятностей:

Символ	Частота	Вероятность	Код	Кол-во бит
е	4	0.1736	00	8
и	3	0.0625	011	9
к	2	0.0417	1101	8
« »	2	0.0417	1100	8
Р	1	0.0417	0100	4
у	1	0.0347	01011	5
д	1	0.0347	01010	5
н	1	0.0347	10101	5
о	1	0.0347	10100	5
А	1	0.0278	10111	5
л	1	0.0278	10110	5
с	1	0.0139	10001	5
й	1	0.0139	10000	5
Д	1	0.0139	10011	5

м	1	0.0139	10010	5
т	1	0.0069	11101	5
р	1	0.0069	11100	5
в	1	0.0069	11111	5
ч	1	0.0069	11110	5

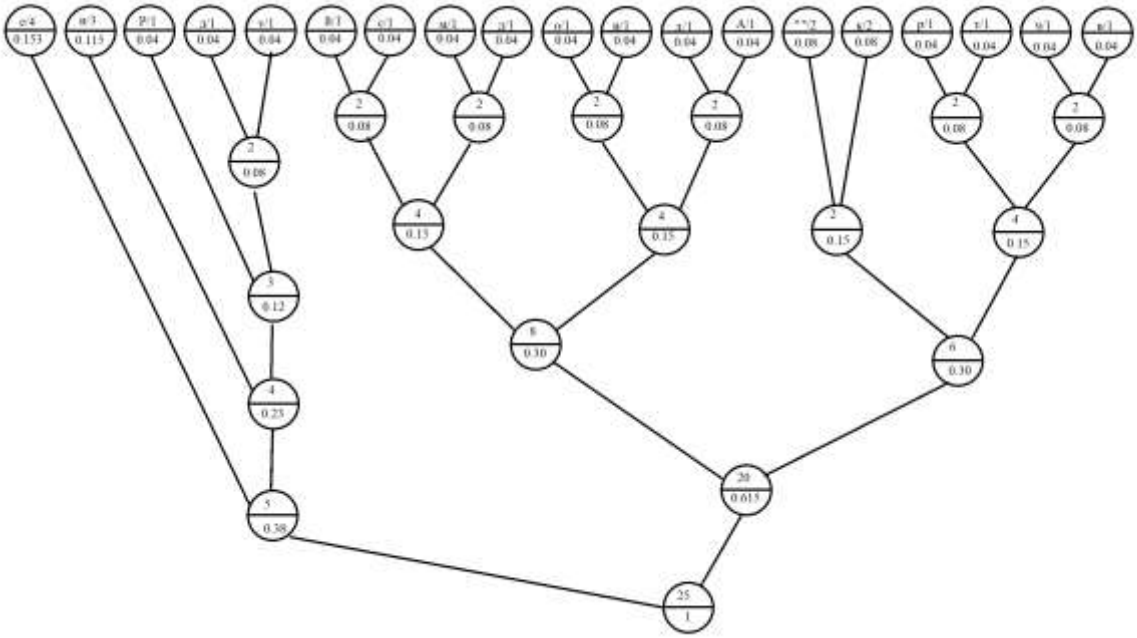

Рисунок 2 — Дерево кодирования Хаффмана

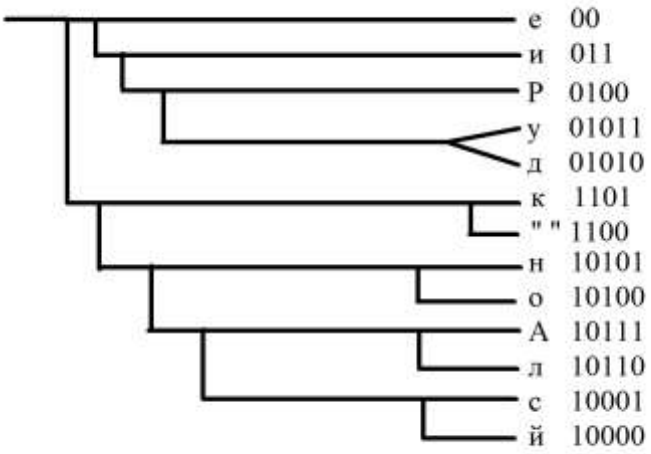

Рисунок 3 — Упорядоченное дерево кодирования Хаффмана

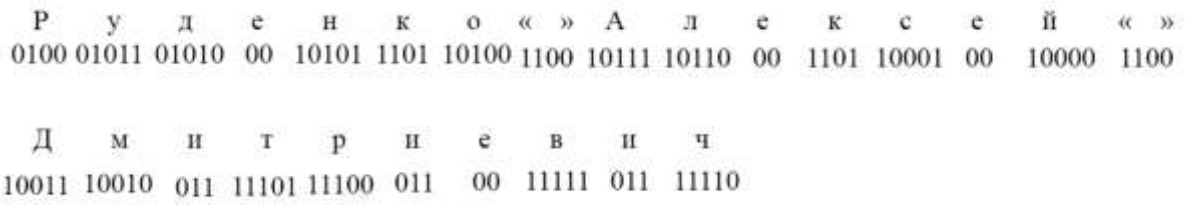

Рисунок 4 — Расшифровка по буквам ФИО

3 ВЫВОД

В рамках практической работы было проведено исследование методов фировки информации. Целью данного исследования было приобрести навыки шифрования и дешифрования, а именно LZ77, LZ78, Хаффмана и Шеннона-Фено.

4 СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. СиАОД Самостоятельная работа 8.pdf
2. Рысин М.Л. и др. Введение в структуры и алгоритмы обработки данных. Ч. 1 - учебное пособие, 2022
3. Рысин М.Л. и др. Основы программирования на языке C++. Учебное пособие, 2022
5. metanit.co
6. ru.cppreference.com/w/
7. habr.com
8. techiedelight.com
9. overcoder.net