Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа по дисциплине «Защита информации»

Тема Алгоритм сжатия LZW

Студент Светличная А. А.

Группа ИУ7-73Б

Преподаватель Чиж И. С.

Москва — 2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном информационном обществе, где обмен и хранение данных становятся все более значимыми, эффективные методы сжатия данных становятся ключевым элементом для оптимизации использования ресурсов. Сжатие данных представляет собой процесс уменьшения объема информации, несущий важное значение.

Одним из важных алгоритмов сжатия данных является алгоритм LZW (Lempel-Ziv-Welch). Разработанный в 1984 году Абрахамом Лемпелем, Йором Зивом и Терри Велчем, LZW представляет собой без потерь метод сжатия, который демонстрирует высокую эффективность при уменьшении размера данных.

Цель: реализовать алгоритм сжатия LZW.

Задачи:

- исследование общих аспектов алгоритма;
- анализ алгоритма LZW;
- программная реализация алгоритма.

1 Аналитическая часть

Основная идея

Алгоритм LZW (Lempel-Ziv-Welch) представляет собой алгоритм сжатия данных, введенный в 1984 году Абрахамом Лемпелем, Йором Зивом и Терри Велчем. Он является одним из ключевых методов без потерь, который демонстрирует высокую эффективность при уменьшении объема данных.

Суть алгоритма заключается в пошаговом построении словаря, который эффективно кодирует повторяющиеся последовательности данных. Исходный словарь инициализируется всеми возможными символами, представляющими входные данные. Процесс обработки данных включает в себя поиск самых длинных подстрок в словаре и их последующее замещение соответствующими кодами. При обнаружении новых подстрок они добавляются в словарь с присвоением уникальных кодов.

Обновление словаря происходит динамически в процессе обработки данных, что позволяет алгоритму адаптироваться к уникальным особенностям входных данных. Результатом работы алгоритма является закодированный поток данных, представляющий из себя сжатую версию исходной информации.

Алгоритм LZW применяется в различных областях, включая формат GIF для сжатия изображений и формат ZIP для архивации файлов. Его успех объясняется высокой степенью компрессии при относительно невысокой вычислительной сложности, что сделало его популярным выбором в индустрии сжатия данных.

2 Конструкторская часть

Алгоритм LZW

Алгоритм LZW-сжатия и распаковки представлены ниже на псевдокоде.

Листинг 2.1 – Процедура LZW-сжатия

```
string s;
char ch;

s = empty string;
while (there is still data to be read)
{
    ch = read a character;
    if (dictionary contains s+ch)
    {
          s = s+ch;
    }
    else
    {
          encode s to output file;
          add s+ch to dictionary;
          s = ch;
    }
}
```

Входная строка : /WED/WE/WEE/WEB/WET

Таблица 2.1 – Пример таблицы для LZW-сжатия

Вход(символы)	**Выход(коды)**	**Новые коды**	**Строки**
/W	/	256	/W
E	W	257	WE
D	E	258	ED
/	D	259	D/
WE	256	260	/WE
/	E	261	E/
WEE	260	262	/WEE
/W	261	263	E/W
EB	257	264	WEB
/	B	265	B/
WET	260	266	/WET
<EOF>	T		

Листинг 2.2 – Процедура LZW-распаковки

```
1  string entry;
2  char ch;
3  int prevcode, currcode;
4
5  prevcode = read in a code;
6  decode/output prevcode;
7  while (there is still data to read)
8  {
9      currcode = read in a code;
10     entry = translation of currcode from dictionary;
11     output entry;
12     ch = first char of entry;
13     add ((translation of prevcode)+ch) to dictionary;
14     prevcode = currcode;
15 }
```

Входные коды : / W E D 256 E 260 261 257 B 260 T

Таблица 2.2 – Пример таблицы для LZW-Распаковки

Вход	**Старый код**	**Строка**	**Символ**	**Новый вход таблицы**
/	/	/		
W	/	W	W	256 = /W
E	W	E	E	257 = WE
D	E	D	D	258 = ED
256	D	/W	/	259 = D/
E	256	E	E	260 = /WE
260	E	/WE	/	261 = E/
261	260	E/	E	262 = /WEE
257	261	WE	W	263 = E/W
B	257	B	B	264 = WEB
260	B	/WE	/	265 = B/
T	260	T	T	266 = /WET

3 Технологическая часть

3.1 Тестирование

1. Большое количество повторяющихся комбинаций

```
1    Input file: sdsdsdsdsdsdsdsdsdsdsdsdsdsdsdsdsdsd
2    Size of input file: 37 B
3    Size of output file: 19 B
4    Decoding: true
```

2. Отсутствие повторяющихся комбинаций

```
1    Input file: qwertyuiopasdfghjklzxcvbnmqazwsxedcrnujmik,
       ↪ ol.p;awzsexdrcftvgybhunjimko,lp
2    Size of input file: 74 B
3    Size of output file: 87 B
4    Decoding: true
```

ЗАКЛЮЧЕНИЕ

В ходе лабораторной работы была достигнута поставленная **цель**: разработана программная реализации алгоритма LZW.

Все **задачи** лабораторной работы выполнены:

- исследованы общие аспекты алгоритма;
- проведен анализ алгоритма LZW;
- программно реализован данный алгоритм.