**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра САПР.**

**ОТЧЕТ**

**по лабораторной работе №3**

**по дисциплине «Алгоритмы и структуры данных»**

**Тема: «Алгоритмы сжатия без потерь».**

| Студентка гр. 1301 |  | Ахметзянов Д.А. |
| --- | --- | --- |
| Преподаватель |  | Родионова Е. А. |

Санкт-Петербург

2023

Оглавление

[Формулировка задания. 3](#_gjdgxs)

[Описание методов и оценка временной сложности. 3](#_30j0zll)

[Пример работы программы. 3](#_3znysh7)

[Текст программы. 3](#_tyjcwt)

# Формулировка задания.

Реализовать следующие алгоритмы сжатия символьных данных:

1. Алгоритм Хаффмана (HA)
2. Кодирование длин серий (RKE)
3. Алгоритм Лемпеля-Зива (LZ78)
4. *Преобразование Барроуза-Уиллера (BWT)*
5. *MTF*
6. Арифметическое кодирование (AC)
7. PPM

Проанализировать эффективность и время сжатия на датасете enwik8 для процедур сжатия на основе:

1. Алгоритма Хаффмана
2. Арифметического кодирования
3. LZ78
4. BWT -> MTF -> HA
5. BWT -> MTF -> AC
6. RLE -> BWT -> MTF -> RLE -> HA
7. RLE -> BWT -> MTF -> RLE -> AC
8. PPM

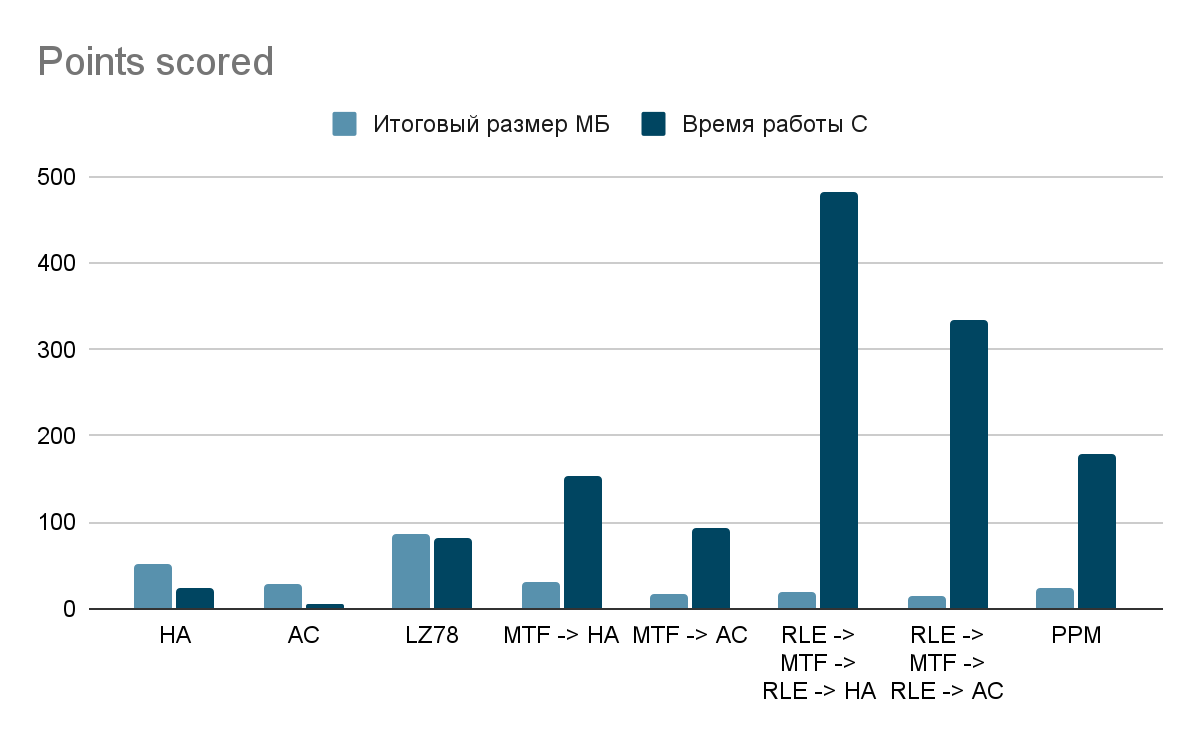
Результаты представить в виде или таблиц. При слишком больших временных затратах на сжатие взять первые 10% enwiki8.

# Описание методов и оценка временной сложности.

1. HA - Суть алгоритма заключается в том, что он строит дерево Хаффмана, в котором каждый лист соответствует символу алфавита, а путь от корня до листа задает код символа. При этом часто встречающиеся символы получают более короткий код, что позволяет сократить длину закодированного сообщения. Сложность - O(N\*log(N))
2. Арифметическая кодировка - это алгоритм сжатия информации без потерь, который позволяет упаковывать символы входного алфавита без потерь при условии, что известно распределение частот символов в тексте. Основная идея заключается в том, чтобы присваивать коды не целым символам, а целым интервалам, которые представляют собой диапазоны вероятностей для каждого символа. При декодировании, если полученное значение попадает в интервал, соответствующий символу, то этот символ восстанавливается. Сложность - O(N)
3. LZ78 - это алгоритм сжатия данных без потерь, он использует словарь для хранения последовательностей символов и кодирует данные в виде пар (индекс, символ), где индекс указывает на последовательность символов в словаре, а символ - на следующий символ, который не входит в эту последовательность. Сложность >= O(N)
4. MTF - это алгоритм кодирования, используемый для предварительной обработки данных, обычно потока байтов, перед сжатием. Он использует простой алгоритм перемещения символов к началу списка, чтобы уменьшить количество повторяющихся символов в данных. При этом каждый символ заменяется на его индекс в списке, а затем перемещается в начало списка. Этот процесс повторяется для каждого символа в потоке данных. Сложность - O(N^2)
5. RLE - это простой алгоритм сжатия данных, который используется для сокращения размера файла путем замены повторяющихся последовательностей символов на один символ и число его повторений. Сложность - О(N)
6. PPM - это алгоритм сжатия данных, который используется для сжатия текстовых файлов. Он основан на предсказании следующего символа в тексте на основе предыдущих символов. Алгоритм PPM использует контекстное моделирование, чтобы определить вероятность следующего символа на основе предыдущих символов. Сложность зависит от размера используемого контекста

Исходный размер файла: 100,000,000 бит

| Название алгоритма | Время работы мкС | Итоговый размер байт |
| --- | --- | --- |
| HA | 23,157,264 | 51195637 |
| AC | 4,557,371 | 29152844 |
| LZ78 | 82,431,160 | 86289003 |
| MTF -> HA | 154,782,848 | 31997273 |
| MTF -> AC | 93,214,587 | 18220527 |
| RLE -> MTF -> RLE -> HA | 481,111,354 | 18961347 |
| RLE -> MTF -> RLE -> AC | 335,063,176 | 15998636 |
| PPM | 179,718,984 | 24372304 |



# Текст программы.

[Ссылка на github](https://github.com/DamirAhm/AADS/tree/master/2_lab3)