Федеральное государственное автономное образовательное учреждение высшего образования
«Российский университет дружбы народов имени Патриса Лумумбы»
Инженерная академия
Департамент механики и процессов управления

ОТЧЕТ

По **Курсовой работе**

Направление: **Управление в технических системах**
(код направления / название направления)

Профиль: **Информационные технологии в управлении и кибербезопасности**
(название профиля)

Тема: **Алгоритм сжатия данных: реализация кодирования Хаффмана**
(название лабораторной / курсовой)

Выполнено студентом: **Ачатова Амелия Павловна, Трофимов Тимофей Васильевич**
(ФИО)

Группа: **ИУСбд-01-23**

№ студенческого: 1132233620
1132233600

Москва, 2025

Цель:

Реализовать алгоритм Хаффмана на C++, продемонстрировать его работу и показать практическое применение.

Задачи:

1. Изучить теорию построения оптимальных префиксных кодов
2. Разработать программу для кодирования и декодирования данных
3. Протестировать алгоритм на различных типах данных
4. Привести примеры использования в реальных задачах

Актуальность:

В мире, где объемы данных возрастают экспоненциально, эффективное сжатие информации становится не только технической задачей, но и экономическим императивом. Одним из самых известных и фундаментальных методов без потерь в теории и практике сжатия данных является **кодирование Хаффмана** — алгоритм, предложенный в 1952 году Дэвидом Хаффманом. Несмотря на свою простоту, он по сей день остается краеугольным камнем во многих системах сжатия.

Теория:

Что такое алгоритм Хаффмана?

Кодирование Хаффмана — Это жадный алгоритм, предназначенный для построения оптимального префиксного кода с минимальной средней длиной кодового слова. Он позволяет эффективно кодировать информацию, используя более короткие коды для более частых символов, тем самым уменьшая общий объем передаваемых или сохраняемых данных.

Историческая справка

Дэвид А. Хаффман разработал свой алгоритм в рамках университетского задания в Массачусетском технологическом институте (MIT). Интересен тот факт, что Хаффман отказался идти традиционным путём доказательства оптимальности известных решений и вместо этого разработал собственный алгоритм — решение, которое сразу же получило признание в научном сообществе. Его статья «A Method for the Construction of Minimum-Redundancy

Codes» была опубликована в 1952 году и с тех пор цитируется как основополагающая работа в области кодирования данных.

Классификация

Алгоритм Хаффмана относится к классу:

- **жадных алгоритмов**: на каждом шаге он выбирает наиболее локально выгодное решение (объединяет два наименее вероятных символа),
- **алгоритмов без потерь** (lossless compression): после декодирования возможно полное восстановление исходных данных без искажения,
- **префиксных кодов** (prefix-free codes): ни один код не является префиксом другого, что делает декодирование однозначным и не требующим специальных разделителей.

Основная идея

Ключевая идея алгоритма Хаффмана заключается в следующем: если в некотором наборе символов одни встречаются значительно чаще других, то можно уменьшить общий объем информации, назначив более короткие коды именно этим частым символам. Редкие символы, напротив, получают более длинные коды. При этом структура кода должна оставаться префиксной, чтобы не терялась однозначность при декодировании.

Как работает алгоритм?

1. *Подсчёт частот:*

   Подсчитывается, сколько раз каждый символ встречается в входном тексте.

2. *Построение дерева Хаффмана:*

   - Создаётся бинарное дерево, где листья представляют символы, а вес каждого узла — суммарная частота.
   - Узлы объединяются попарно, начиная с самых редких (минимальные частоты), пока не остаётся один корневой узел (символ с суммарной частотой всех узлов).

3. *Генерация кодов:*

   По построенному дереву каждому символу присваивается уникальный двоичный код: идём от корня к листу, двигаясь влево — добавляем “0”, вправо — “1”.

4. Кодирование:

   Замена каждого символа в исходном тексте на соответствующий ему двоичный код.

5. Декодирование:

   Обратный процесс: используя дерево Хаффмана, по последовательности битов восстанавливается исходный текст.

Пример работы программы

Для строки "ABRACADABRA"

частоты символов:

- A: 5, B: 2, R: 2, C: 1, D: 1

коды по дереву Хаффмана:

- A: 0, B: 111, R: 110, C: 1001, D: 1000

Закодированная строка: 01011001110011110101100

Декодированная строка: ABRACADABRA

Исходный размер: 11 символов * 8 бит = 88 бит

Закодированный размер: 22 бита (сжатие в 4 раза)

Когда Хаффман действительно эффективен

1. **Неравномерное распределения частот символов в сообщении** - чем больше различие между частотами, тем выше степень сжатия.
2. **Стационарность источника** - когда вероятности символов остаются постоянными в пределах сжимаемого блока.
3. **Ограниченный алфавит** - при большом количестве уникальных символов (например, в Unicode-текстах) производительность и выигрыш могут снижаться.

Недостатки

1. **Чувствительность к мелким изменениям**: изменение одной частоты может привести к перестройке всего дерева.

2. **Неоптимальность при дробных вероятностях** - если вероятности символов не являются степенями двойки (например, 0.5, 0.25, 0.125…), то невозможно выдать им идеальные коды длиной log2(1/pi), что и нужно для достижения энтропии. Хаффман вынужден округлять длину кода до ближайшего целого вверх. Это создает избыточность, которая мешает достичь минимальной средней длины.
3. **Низкая эффективность на малых объемах данных** - накладные расходы на таблицу кодов могут перевесить выигрыш от сжатия.
4. **Неспособность уловить зависимости между символами** - каждый символ кодируется независимо от контекста, в отличие от более продвинутых моделей.

Примеры практического применения

1. *Архиваторы:*

   Форматы ZIP, 7z используют Хаффмана в комбинации с другими алгоритмами.

2. *Изображения:*

   Формат PNG применяет Хаффмана для сжатия без потерь.

3. *Сети:*

   Протоколы HTTP/2 и QUIC используют сжатие заголовков методом Хаффмана.

4. *Базы данных:*

   СУБД (например, PostgreSQL) сжимают текстовые данные для экономии места.

При передаче текстовых сообщений применяется сжатие Хаффмана, что снижает трафик на 30-50%

Заключение

Таким образом, алгоритм Хаффмана представляет собой классическое и одновременно практичное решение задачи оптимального сжатия данных. Его простота сочетается с высокой эффективностью, а теоретическая строгость — с реальной применимостью. Он иллюстрирует, как методы теории информации могут быть напрямую внедрены в технологии, оказывающие влияние на повседневную цифровую инфраструктуру.

Использованная литература:

- http://e-maxx.ru/bookz/files/cormen.pdf
- https://habr.com/ru/companies/otus/articles/497566/
- https://habr.com/ru/articles/438512/
- https://www.youtube.com/watch?v=q_N2Y6-O1xw&pp=ygVa0LDQu9Cz0L7RgNC40YLQvCDRgdC20LDRgtC40Y8g0LHQtdC3INC_0L7RgtC10YDRjCDQsNC70LPQvtGA0LjRgtC80Ysg0YXQsNGE0YTQvNCw0L3QsCDRgSsr
- https://github.com/madler/zlib

GitHub: https://github.com/timotroll/coursovaia-4sem-cpp