ЗВІТ

Команда №4
Єлізавета Пілецька, Діана Лизенко, Анастасія Яблуновська, Гобела Максим
Ментори: Юлія Колодій, Андрій Музичук

Дослідження методів компресії даних та створення власної програми-кодека

ВСТУП

У даному проекті нам пропонується розробити кілька алгоритмів стиснення та проаналізувати їх роботу з текстовими файлами, зображеннями, відео, аудіо тощо. Для цього ми імплементували кілька алгоритмів, а саме: **LZ77, LZW, алгоритм Хаффмана, Deflate (LZ77 + Хаффмана)**. Варто також зазначити, що усі формати спочатку конвертуються у нас в бітові рядки.

ВИМОГИ

Для роботи алгоритму Deflate знадобиться встановити бібліотеки **heapq, pickle, datetime**. Також необхідна бібліотека **argparse**, яка буде запускати основну програму.

ЗАПУСК

Для запуску потрібно ввести у термінал наступну команду:

```
python3 main_argparse.py text.txt lz77
```

Тобто, як видно, це назва файлу + документ, що цікавить + алгоритм, за яким Ви хочете здійснити стиснення.

LZ77

Техніка стиснення LZ77 зменшує розмір даних, визначивши та кодуючи повторювані шаблони в вхідному потоці. Він використовує механізм "розсувного вікна"для виявлення найбільш розширених послідовностей подібності між нинішнім набором даних та тими, які були проаналізовані раніше. Ідентифіковані відповідності замінюються на кортежі (відстань, довжина), що означає розташування та ступінь повторюваної схеми. **Для файлів з широким повторюваним вмістом (наприклад, текст) алгоритм LZ77 працює з високою ефективністю. Тим не менш, при роботі з вже стислими даними (наприклад, відеофайлів MP4), може бути незначний вплив або потенційне збільшення розміру файлів через додаткові метадані**.

LZW

Під час процесу стиснення алгоритм LZW конструює лексикон на основі зіткнених даних, замінюючи вихідну послідовність даних відповідними представленнями коду. **Він виявляється особливо вигідним для файлів документів, однак для аудіо файлів він дає неймовірно мале стиснення. Це так само може бути пов'язаним з форматом даних**.

Алгоритм Хаффмана

Алгоритм Хаффмана використовує частоту символів у даних для побудови оптимального префіксного коду. Символи, які зустрічаються частіше, отримують короткі коди, а рідкісні — довші. **Цей метод добре працює для текстових файлів, де частота символів нерівномірна**. **Однак для аудіо або відео, які часто вже стиснуті (наприклад, MP3 або MP4), алгоритм Хаффмана дає незначний ефект, оскільки дані вже**

майже оптимізовані.

Deflate (LZ77 + Хаффмана)

Deflate поєднує LZ77 для попереднього стиснення даних (шляхом усунення повторень) та алгоритм Хаффмана для подальшого кодування. Це робить його універсальним для різних типів даних. У нашій інтерпретації Deflate це LZ77 + алгоритм Хаффмана, комібнований з хешуванням, який робить процес швидшим та конвертуванням файла у байти і навпаки. **Тестування на зображеннях показали, що така версія алгоритму дуже добре підходить для нестиснутих форматів, (bmp, tiff) і є дуже проблемною для стиснутих, (png, jpg).**При взаємодії з отсанніми форматами алгоритм зробить об'єм пам'яті лише більшим.

ВИСНОВОК

Підсумовуючи, можна сказати, що алгоритми працюють добре, особливо для текстових файлів. Однак для відео аудіо та зображення можуть виникнути проблеми через їхній формат, оскільки стиснення уже стиснутих даних може призводити до збільшення об'єму пам'яті файлу.

РОЗПОДІЛ РОБОТИ:

- **Лизенко Діана**: реалізація LZ77, тестування;
- **Пілецька Єлізавета**: LZW, DEFLATE, тестування аудіо файлів, реалізація файлу README, презентація;
- **Яблуновська Анастасія**: Huffman Coding, тестування текстових файлів, реалізація argprase, презентація;
- **Гобела Максим**: DEFLATE, тестування файлів із зображенням.