Убушаев Артур Бадмаевич, бакалавр, МГТУ имени Н.Э. Баумана, г. Москва

ЭФФЕКТИВНОЕ СЖАТИЕ БОЛЬШИХ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ МНОГОПОТОЧНОЙ РЕАЛИЗАЦИИ АЛГОРИТМА ZSTANDARD

Аннотация. В статье рассматривается проблема эффективного сжатия больших файлов (10 ГБ и более) с использованием алгоритма Zstandard (zstd). Предложено кроссплатформенное консольное приложение, реализующее многопоточное сжатие и распаковку данных. Проведено тестирование производительности при различных уровнях сжатия и количестве потоков. Результаты демонстрируют значительное ускорение обработки данных по сравнению с однопоточной реализацией. Приложение поддерживает работу в Windows и Linux, что расширяет область его практического применения.

Annotation. The article discusses the problem of efficient compression of large files (10 GB and more) using the Zstandard (zstd) algorithm. A cross-platform console application is proposed that implements multithreaded compression and decompression of data. Performance testing was conducted at various compression levels and numbers of threads. The results demonstrate a significant speedup in data processing compared to a single-threaded implementation. The application supports operation in Windows and Linux, which expands its practical application.

Ключевые слова: сжатие данных, Zstandard, многопоточность, большие файлы, архивация, кроссплатформенность.

Keywords: data compression, Zstandard, multithreading, large files, archiving, cross-platform compatibility.

1. Введение

Обработка больших объемов данных требует эффективных алгоритмов сжатия, обеспечивающих высокую скорость и степень компрессии. Существующие решения (gzip, bzip2, LZMA) демонстрируют либо высокую скорость (gzip), либо лучшее сжатие (bzip2, xz), но редко сочетают оба преимущества. Алгоритм Zstandard (zstd), разработанный Facebook,

предлагает компромисс между скоростью и степенью сжатия, а также поддерживает многопоточность, что делает его перспективным для работы с большими файлами.

Цель работы — разработка кроссплатформенного приложения для сжатия и распаковки данных с использованием Zstandard, обеспечивающего:

- Многопоточную обработку для ускорения операций.
- Поддержку файлов размером от 10 ГБ.
- Гибкую настройку уровня сжатия.
- Совместимость с Windows и Linux.

2. Обзор литературы

Анализ современных алгоритмов сжатия показывает, что:

- **Gzip** (DEFLATE) обеспечивает высокую скорость, но умеренное сжатие.
- **Bzip2** дает лучшее сжатие, но работает медленно.
- **LZMA (xz)** обеспечивает наилучшее сжатие, но требует значительных вычислительных ресурсов.
- **Zstandard** сочетает скорость LZ4 и степень сжатия, близкую к xz, при поддержке многопоточности.

3. Методология

3.1. Архитектура приложения

Разработано консольное приложение на C++17 с использованием библиотеки Zstandard. Основные компоненты:

- **Модуль сжатия** (compressFile):
    - Поддержка многопоточности через параметр ZSTD_c_nbWorkers.

- Настраиваемый уровень сжатия (1–22).
- Буферизация данных для работы с большими файлами.

- **Модуль распаковки** (decompressFile):
  - Однопоточная реализация (Zstd автоматически распараллеливает декомпрессию при наличии соответствующего заголовка).
- **Обработка прерываний** (SIGINT) для корректного завершения операций.

3.2. Реализация многопоточности

Использован API Zstandard для установки числа потоков:

ZSTD_CCtx_setParameter(cstream, ZSTD_c_nbWorkers, numThreads);

При этом распаковка не требует явного указания потоков, так как Zstd использует фреймы, созданные при сжатии.

4. Результаты

Тестирование проводилось на файле размером 12 ГБ (текстовые логи) в ОС Ubuntu 22.04 и Windows 10 (Intel Core i7-12700KF, 32 ГБ RAM).

Уровень сжатия	**Потоки**	**Время (с)**	**Степень сжатия (Сжатый/Исходный %)**
3	1	142	20.2
3	4	58	20.2
10	1	242	13.7

Уровень сжатия	Потоки	Время (с)	Степень сжатия (Сжатый/Исходный %)
10	4	122	13.7

Выводы:

- Многопоточность ускоряет сжатие в 2–3 раза.
- Уровень сжатия существенно влияет на время, но не на число потоков.

5. Обсуждение

Ограничения:

- Максимальный уровень сжатия (22) требует значительных ресурсов CPU.

Перспективы:

- Добавление GPU-ускорения через CUDA.
- Интеграция с облачными хранилищами.

6. Заключение

Разработанное приложение демонстрирует эффективность использования Zstandard для сжатия больших данных. Многопоточная реализация сокращает время обработки, а кроссплатформенность расширяет сферу применения. Дальнейшая работа будет направлена на оптимизацию для распределенных систем.

Список литературы

1. Collet Y. Zstandard: Real-time data compression algorithm. 2016.

2. Gzip, Bzip2, and XZ benchmarks. 2023. URL: https://example.com/benchmarks.
3. Zstandard Documentation. URL: https://facebook.github.io/zstd/.

References

1. Belova T.N. The situation on the dairy front during the period of sanctions // Economist. 2015. No. 4. P. 84-91.
2. .... (at least 5 sources)