УДК 004.942

Сравнительный анализ современных методов сжатия без потерь данных облачных провайдеров

Юдников Степан Сергеевич[1] — магистрант 2-го курса направления 09.04.02 «Информационные системы и технологии». Научные интересы: информационные системы, искусственный интеллект, методы и алгоритмы сжатия данных. E-mail: s.yudnikovv@gmail.com

Хомоненко Анатолий Дмитриевич[1,2] — д-р техн. наук, профессор, профессор кафедры «Информационные и вычислительные системы»; профессор кафедры математического и программного обеспечения. Научные интересы: информационные системы, обработка больших данных, вероятностное моделирование геоинформационных систем, генетические алгоритмы, информационная безопасность. E-mail: khomon@mail.ru

[1]Петербургский государственный университет путей сообщения Императора Александра I, Россия, 190031, Санкт-Петербург, Московский пр., 9

[2]Военно-космическая академия имени А. Ф. Можайского, Россия, 197198, Санкт-Петербург, ул. Ждановская, 13

Для цитирования: Юдников С. С., Хомоненко А. Д. Сравнительный анализ современных методов сжатия без потерь данных облачных провайдеров // Интеллектуальные технологии на транспорте. 2026. № 1 (45). С. 41–50. DOI: 10.20295/2413-2527-2026-145-41-50

Аннотация. *Экспоненциальный рост объемов данных в облачных средах делает критически важной задачу их эффективного сжатия для оптимизации использования ресурсов хранения, пропускной способности сети и вычислительных мощностей, что напрямую влияет на экономическую и операционную эффективность.* ***Цель:*** *провести сравнительный анализ современных алгоритмов сжатия без потерь для определения оптимальных сценариев их применения в облачных платформах.* ***Результаты:*** *систематизированы принципы работы и классификация современных алгоритмов сжатия без потерь. На основе сравнительного анализа по ключевым параметрам (коэффициент сжатия, скорость операций, ресурсоемкость) определены сценарии оптимального применения алгоритмов Gzip, LZ4, Zstandard и Brotli в облачных сервисах.* ***Практическая значимость:*** *полученные результаты можно использовать для оптимизации затрат на облачную инфраструктуру и повышения производительности распределенных систем обработки данных.* ***Обсуждение:*** *проведенный анализ демонстрирует, что выбор алгоритма зависит от специфики задачи. Использование аппаратного ускорения позволяет значительно повысить производительность компрессии.*

Ключевые слова: *компрессия данных, сжатие без потерь, облачные вычисления, алгоритмы сжатия, Gzip, LZ4, Zstandard, Brotli, производительность облачных систем, оптимизация затрат, сетевой трафик, аппаратное ускорение*

1.2.1 — *искусственный интеллект и машинное обучение (технические науки);* **2.3.1** — *системный анализ, управление и обработка информации (технические науки)*

Введение

Экспоненциальный рост объемов данных в облачных средах, обусловленный цифровой трансформацией, распространением IoT-устройств и развитием аналитических систем, создает серьезные вызовы для эффективного управления информационными ресурсами [1].

Ключевой проблемой современной облачной экосистемы является оптимизация использования трех критически важных ресурсов: емкости хранения, пропускной способности сетей и вычислительной мощности. Традиционные подходы к масштабированию инфраструктуры демонстрируют ограниченную эффективность в условиях нелинейного роста данных, что актуализирует необходимость внедрения интеллектуальных методов оптимизации.

Сжатие данных без потерь представляет собой один из наиболее эффективных инструментов решения указанных проблем. В отличие от компрессии с потерями, применяемой преимущественно для мультимедийного контента, алгоритмы без потерь обеспечивают полное сохранение семантической целостности данных, что особенно критично для бизнес-приложений, транзакционных систем и нормативной отчетности.

Современные облачные платформы предоставляют широкий спектр алгоритмов сжатия — от классического Gzip до высокопроизводительных Zstandard и Brotli, демонстрирующих радикально разные характеристики скорости, степени сжатия и вычислительной сложности [2]. При этом отсутствие системного подхода к выбору алгоритма под конкретную задачу часто ведет к неэффективному использованию ресурсов и росту операционных расходов.

В условиях экспоненциального роста данных в облаках поиск оптимальных методов сжатия без потерь становится ключевым фактором экономии и повышения производительности. В статье представлен сравнительный анализ алгоритмов сжатия с учетом аппаратной акселерации, а также предложены адаптивные критерии их выбора для типичных сценариев использования в облачных средах. Основной результат — практические рекомендации и модель выбора, учитывающая тип рабочей нагрузки и целевые показатели производительности, что и составляет научную новизну работы.

Практическая значимость исследования определяется возможностью непосредственного применения полученных результатов для оптимизации затрат на облачную инфраструктуру и повышения производительности распределенных систем обработки данных.

Характеристика и классификация современных алгоритмов сжатия

Ключевая задача алгоритмов сжатия — снизить избыточность представления данных, минимизируя объем хранимой или передаваемой информации. Их важность особенно возрастает в контексте облачных платформ, где сжатие напрямую влияет на ключевые показатели эффективности: стоимость хранения, пропускную способность сети и скорость обработки транзакций [3].

Существует две парадигмы сжатия данных:

• сжатие с потерями (lossy compression) используется преимущественно для мультимедийных данных (изображения, звук, видео), когда допустимо удаление незначимой информации;

• сжатие без потерь (lossless compression) сохраняет точную копию исходных данных после восстановления. Именно такие алгоритмы применяются при работе с программным кодом, документами, логами, базами данных, конфигурационными файлами и архивами в информационных системах.

В статье основное внимание уделяется алгоритмам сжатия без потерь как наиболее востребованным в сфере облачных технологий, где важно сохранять целостность и точность передаваемых и обрабатываемых данных.

Методы сжатия без потерь можно условно разделить на статистические и словарные.

Статистические методы сжатия основаны на разной вероятности появления символов (или их последовательностей) в данных, присваивая более короткие коды наиболее частым элементам. К числу ключевых методов статистического сжатия относятся:

• кодирование Хаффмана (Huffman coding) — алгоритм, строящий оптимальное префиксное дерево для присвоения переменных кодов символам [4]. Благодаря эффективности и простоте он стал основой ряда распространенных форматов, таких как Gzip, PNG, используется в составе более сложных алгоритмов (например, DEFLATE);

• арифметическое кодирование — работает с вещественными интервалами вероятностей, позволяя достичь более высокой степени сжатия, чем алгоритм Хаффмана, но требует больше ресурсов [5];

• кодирование Шеннона — Фано (Shannon — Fano coding) — предшественник метода Хаффмана, менее эффективный, но демонстрирующий основные принципы статистической компрессии [6].

Словарные методы сжатия основываются на замене повторяющихся фрагментов данных указателями на словарь, содержащий ранее встречающиеся шаблоны. Подход показывает высокую эффективность на текстовых данных. Наиболее известные алгоритмы:

• LZ77 — использует скользящее окно для поиска повторяющихся строк в недавней истории потока данных [7];

• LZ78 — создает динамический словарь из уникальных подстрок и работает эффективнее в случаях, где повторяющиеся шаблоны разнесены по потоку [8];

• LZW — усовершенствованная версия LZ78, получившая широкое распространение, в частности, в формате GIF и протоколе TIFF [9];

• LZ4 и Zstandard (ZSTD) — высокоскоростные алгоритмы нового поколения, представляющие эволюцию семейства LZ. Ключевыми их преимуществами являются экстремальная скорость (LZ4) и настраиваемый компромисс между производительностью и степенью сжатия (ZSTD) [10].

Принципы работы основных алгоритмов сжатия данных

В современных вычислительных системах, в том числе в облачных средах, применяется ряд алгоритмов уменьшения объема информации. Каждый из них имеет свои сильные стороны, ограничения и оптимальные сферы использования.

Алгоритм Gzip и гибридная модель DEFLATE

Данный метод остается одним из наиболее распространенных решений для уменьшения размера данных [11]. В его основе лежит гибридная модель DEFLATE, которая комбинирует два подхода [12]:

• словарное сжатие (LZ77) обнаруживает и заменяет повторяющиеся фрагменты данных ссылками в специальном словаре;

• энтропийное кодирование (Хаффмана) присваивает более короткие коды часто встречающимся элементам, а более длинные — редким.

Главные достоинства Gzip — повсеместная поддержка в программном обеспечении и операционных системах, а также баланс между степенью сжатия и скоростью работы. Этот алгоритм часто используют для уменьшения веб-трафика, ускорения загрузки сайтов, архивирования файлов и хранения журналов событий [13].

Алгоритм LZ4

Данный метод разработан для достижения предельной скорости как упаковки, так и восстановления данных. Эта особенность сделала его популярным в системах, чувствительных к задержкам, — при обработке потоков информации и в приложениях реального времени. В основе LZ4 также лежит принцип LZ77, но реализованный с минимальными вычислениями и упрощенной логикой поиска повторов [14].

Ключевые свойства LZ4:

• исключительно быстрая распаковка (до 1200 МБ/с), что позволяет работать с большими массивами данных почти без задержек;

• умеренная степень сжатия (примерно в 1,5–1,8 раза), что меньше, чем у более сложных алгоритмов, но приемлемо для многих задач;

• низкое потребление вычислительных мощностей, что позволяет использовать его в системах с ограниченными ресурсами, например в IoT-устройствах.

LZ4 часто применяется в облачных платформах для сжатия временных файлов, логов и в системах потоковой передачи данных.

Алгоритм Zstandard: баланс скорости и эффективности

Это современный метод, созданный в 2015 году, который сочетает высокую скорость, характерную для LZ4, и хорошую степень сжатия, как у Gzip. Он предлагает до 22 уровней сжатия, позволяя

гибко выбирать между скоростью и итоговым размером данных [13].

Основные черты ZSTD:

• высокая степень сжатия (может достигать 5:1), что заметно превосходит Gzip;

• эффективная работа в многопоточном режиме и поддержка потоковой обработки, что хорошо подходит для современных серверов;

• широкая интеграция с популярными системами хранения и базами данных.

Эти качества делают ZSTD стандартом дефакто для сжатия в облачных системах.

Алгоритм Brotli

Этот метод, разработанный компанией Google, ориентирован на максимально эффективное сжатие веб-контента [15]. Он использует сложные схемы кодирования, учитывающие контекст данных, а также обширный предустановленный словарь распространенных фрагментов HTML, CSS и JavaScript.

Особенности Brotli:

• наивысшая степень сжатия, особенно для небольших веб-файлов;

• относительно невысокая скорость упаковки, что ограничивает его применение в сценариях, где важна скорость записи;

• достаточная скорость восстановления для клиентских устройств;

• идеален для оптимизации веб-трафика через сети доставки контента (CDN) для ускорения загрузки страниц.

Brotli поддерживается всеми современными браузерами и активно используется в CDN для экономии трафика и повышения скорости отклика сайтов.

Сравнение алгоритмов сжатия

Чтобы выбрать подходящий алгоритм для облачной платформы, необходимо учитывать несколько параметров: итоговый коэффициент уменьшения объема, быстродействие при сжатии и распаковке, а также нагрузку на систему. В табл. 1 приведены сводные характеристики четырех рассмотренных методов:

1. Степень сжатия. Показывает, насколько уменьшается исходный объем данных. Brotli и ZSTD в этом аспекте лидируют, что делает их предпочтительными для экономии места на диске или пропускной способности сети.

2. Скорость операций. Быстрота сжатия критична при работе с большими потоками данных. Здесь лидирует LZ4 благодаря простой реализации. ZSTD предлагает хороший баланс. Brotli — самый медленный.

3. Скорость распаковки. Особенно важна для сервисов, часто читающих данные. LZ4 и здесь является лидером.

4. Требования к ресурсам. Алгоритмы с высокой степенью сжатия (Brotli, ZSTD на высоких уровнях) обычно требуют больше памяти и процессорного времени. LZ4, напротив, очень нетребователен и может работать на маломощном оборудовании.

Выбор конкретного алгоритма должен основываться на специфике решаемой задачи, типе обрабатываемой информации и требованиях к производительности системы.

Таблица 1

Сравнительные характеристики алгоритмов сжатия

Алгоритм	Степень сжатия	Скорость сжатия	Скорость распаковки	Требования к ресурсам
Gzip	Средняя (~2:1)	Средняя	Средняя	Средние
LZ4	Низкая (1,5–1,8:1)	Очень высокая	Очень высокая	Низкие
ZSTD	Высокая (3,5–5:1)	Высокая	Высокая	Средние
Brotli	Очень высокая	Низкая	Средняя	Высокие

Влияние сжатия на эффективность и затраты

Применение сжатия в облачных платформах напрямую влияет на экономическую и операционную эффективность работы с большими данными. Можно выделить несколько ключевых аспектов этого влияния.

Экономия ресурсов хранения

Основное преимущество — значительное сокращение физического объема данных. В облачных средах, где стоимость услуг хранения напрямую зависит от занимаемого места, это приводит к прямому снижению расходов.

В Microsoft Azure Synapse Analytics зафиксированы показатели коэффициента сжатия до 10:1 для специфичных наборов аналитических данных [6]. В Google BigQuery экономия на хранении данных при использовании встроенных алгоритмов сжатия достигает 75 % по сравнению с необработанными данными [8].

Это достигается за счет нескольких факторов:

• использования эффективных алгоритмов сжатия, адаптированных под структуру данных;

• применения колоночных форматов, которые лучше сжимают однородные данные;

• автоматического определения платформой оптимального кодека.

Экономия в объемах хранения не только снижает прямые затраты на дисковое пространство, но и уменьшает нагрузку на системы резервного копирования и восстановления данных, что дополнительно оптимизирует затраты.

Оптимизация сетевого трафика

Передача данных между облачными сервисами, пользователями и периферийными устройствами требует значительных сетевых ресурсов. Сжатие существенно сокращает объем передаваемых данных, что влияет на скорость обмена и стоимость каналов связи.

Так, применение gzip-компрессии позволяет снизить размер логов и телеметрии на 60–80 %, что уменьшает задержки и сетевые расходы при мониторинге инфраструктуры.

Алгоритм Brotli, оптимизированный для веб-трафика, уменьшает объем HTML, CSS и JavaScript-файлов примерно на 20–25 % [10], что положительно сказывается на скорости загрузки веб-страниц и снижении затрат на CDN.

Использование Zstandard в REST API и облачных сервисах позволяет сократить размер запросов и ответов до 50 % [11], значительно повышая пропускную способность и снижая задержки.

Сокращение трафика особенно важно для облаков с гибридной архитектурой и edge-устройств, где каналы связи могут иметь ограниченную пропускную способность и высокую стоимость.

Ускорение аналитики и потоковой обработки

Сжатие данных напрямую ускоряет их обработку в облачных системах. Меньший размер информации быстрее считывается и передается, что сокращает время вычислений.

Сервисы вроде AWS Redshift и Google BigQuery показывают ускорение выполнения запросов в 3–4 раза при работе со сжатыми данными по сравнению с несжатыми [16, 17]. Это происходит потому, что системе нужно прочитать и переместить меньше данных и они эффективнее помещаются в быструю память (кэш).

Алгоритм LZ4, имеющий очень высокую скорость восстановления данных (свыше 1 ГБ/с), отлично подходит для обработки потоков в реальном времени, например при анализе логов и мониторинге.

ZSTD дает хороший баланс между тем, насколько сильно он сжимает, и тем, как быстро данные потом можно использовать. Это делает его надежным решением для задач анализа больших данных.

Эффективное сжатие также помогает системам легче масштабироваться, так как уменьшение объема хранимой и передаваемой информации снижает нагрузку на сеть и серверы.

Использование аппаратных ускорителей сжатия

Для повышения производительности и энергоэффективности облачные платформы активно внедряют специализированные аппаратные блоки, которые разгружают CPU от ресурсоемких операций сжатия:

• Intel QuickAssist (QAT) — выделенный сопроцессор для аппаратного ускорения алгоритмов Gzip и ZSTD, способный в несколько раз увеличить скорость операций и снизить энергопотребление [13];

• процессоры Amazon Graviton на базе архитектуры ARM содержат встроенные аппаратные оптимизации для эффективной работы с алгоритмами LZ4 и ZSTD непосредственно в облачной среде;

• более сложные решения на базе программируемых микросхем (FPGA) и специализированных чипов (ASIC) используются для сжатия в реальном времени, особенно в системах передачи данных и распределенных хранилищах.

Использование такого оборудования позволяет свести задержки к минимуму и увеличить пропускную способность, что крайне важно для приложений, работающих в реальном времени, и аналитических систем.

Примеры и данные из практики

Исследования и реальные случаи использования показывают значительный эффект от сжатия в облачных средах.

В Microsoft Azure Synapse автоматическое сжатие данных в хранилищах колоночного типа может уменьшать занимаемый объем до 10 раз. Это напрямую снижает ежемесячные расходы на хранение.

Согласно данным Google Cloud, встроенное сжатие в BigQuery позволяет экономить до 75% места. При этом аналитические запросы ускоряются почти в 4 раза за счет меньшей нагрузки на систему чтения/записи.

Эксперименты с Intel QAT показывают, что аппаратное ускорение может повысить производительность сжатия и распаковки в 5 раз по сравнению с программными методами, снижая энергопотребление на 30 %.

В задачах потоковой обработки, таких как сбор телеметрии, использование LZ4 обеспечивает скорость восстановления данных до 1200 МБ/с, что критично для работы в реальном времени без задержек.

Практические советы по выбору алгоритма

На основе анализа можно дать следующие рекомендации по выбору:

1. Для долгосрочного хранения больших объемов данных лучше подходят алгоритмы с высокой степенью сжатия, такие как ZSTD и Brotli. Они сильно экономят место при сохранении приемлемой скорости доступа.

2. Для передачи данных и веб-контента оптимальны Brotli и Gzip, так как они хорошо сжимают и поддерживаются повсеместно.

3. Для систем реального времени и потоковой обработки критически важна скорость распаковки. Здесь лидер — LZ4, который минимизирует задержки.

4. Для максимальной производительности и разгрузки процессора стоит рассмотреть использование аппаратного ускорения, например Intel QAT.

Заключение

Сжатие данных — ключевой инструмент для оптимизации облачных платформ, влияющий и на стоимость, и на эффективность работы. В условиях постоянного роста объемов информации из-за развития интернета вещей, аналитики, машинного обучения и других технологий эффективное сжатие становится необходимостью для масштабируемости и устойчивости облачных систем.

Польза сжатия подтверждена расчетами и практикой. Оно может сокращать расходы на хранение в 3–10 раз, ускорять выполнение аналитических запросов в 3–4 раза и значительно снижать сетевой трафик в распределенных системах.

Таким образом, сжатие данных — обязательный элемент при построении облачных систем. Особенно перспективными выглядят следующие направления:

• адаптивное сжатие, которое может автоматически подбирать лучший алгоритм в зависимости от типа обрабатываемых данных;

• аппаратное ускорение сжатия на базе QAT, FPGA, ARM Graviton и интеллектуальных сетевых интерфейсов;

• развитие edge-компрессии с низкой задержкой для распределенных вычислений и IoT;

• глубокая интеграция компрессии с форматами хранения и виртуализации (например, компрессия в файловых системах уровня Ceph, ZFS, Btrfs).

Эффективность облачных решений в XXI веке все более будет зависеть от качества и гибкости применяемых алгоритмов сжатия.

СПИСОК ИСТОЧНИКОВ

1. Верзилин Д. Н., Максимова Т. Г., Шаныгин С. И. Облачные технологии в развитии институтов цифровой трансформации российской экономики: статистическое исследование // Вестник Санкт-Петербургского университета. Экономика. 2025. Т. 41, вып. 1. С. 146–178. DOI: 10.21638/spbu05.2025.107.

2. Давыденко Е. А., Сабиргалиева А. М. Архивирование и сжатие файлов в Linux: методы, алгоритмы и их эффективность // Научный вестник гуманитарно-социального института. 2025. № 20. 5 с.

3. Чуркин Я. М. Сжатие алгоритмом Хаффмана // Инструменты и механизмы устойчивого инновационного развития: сборник статей по итогам Всероссийской научно-практической конференции (Самара, Россия, 6 февраля 2022 г.). Стерлитамак: Агентство международных исследований, 2022. С. 20–22.

4. Левин И. И., Дудников Е. А. Структурная модификация метода Хаффмана для сжатия плотных потоков данных без потерь на РВС // Известия Южного федерального университета. Технические науки. 2024. № 5 (241). С. 48–58. DOI: 10.18522/2311-3103-2024-5-48-58.

5. Howard P. G., Vitter J. S. Arithmetic Coding for Data Compression // Proceedings of the IEEE. 1994. Vol. 82, iss. 6. Pp. 857–865. DOI: 10.1109/5.286189.

6. Добычин Р. С., Петров А. Р. Сравнительный анализ работы алгоритмов Шеннона — Фано и Хаффмана // Междисциплинарные исследования в области математического моделирования и информатики: материалы 7-й научно-практической internet-конференции (Тольятти, Россия, 30–31 марта 2016 г.). Ульяновск: Зебра, 2016. С. 12–14.

7. Kreft S., Navarro G. LZ77-Like Compression with Fast Random Access // Proceedings of the 2010 Data Compression Conference (Snowbird, UT, USA, 24–26 March 2010). Institute of Electrical and Electronics Engineers, 2010. Pp. 239–248. DOI: 10.1109/DCC.2010.29.

8. LZ78 Compression in Low Main Memory Space / D. Arroyuello [et al.] // String Processing and Information Retrieval (SPIRE 2017): Proceedings of the 24th International Symposium, (Palermo, Italy, 26–29 September 2017). Lecture Notes in Computer Science. Vol. 10508 / G. Fici [et al.] (eds). Cham: Springer, 2017. Pp. 38–50. DOI: 10.1007/978-3-319-67428-5_4.

9. Dheemanth H. N. LZW Data Compression // American Journal of Engineering Research. 2014. Vol. 3, iss. 2. Pp. 22–26.

10. Faykus M. H., Calhoun J., Smith M. Lossy and Lossless Compression for BioFilm Optical Coherence Tomography (OCT) // SC-W 2023: Proceedings of the Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis (Denver, CO, USA, 12–17 November 2023). New York: Association for Computing Machinery, 2023. Pp. 281–288. DOI: 10.1145/3624062.3625125.

11. PymzML v2.0: Introducing a Highly Compressed and Seekable GZip Format / M. Kösters [et al.] // Bioinformatics. 2018. Vol. 34, no. 14. Pp. 2513–2514. DOI: 10.1093/bioinformatics/bty046.

12. Oswal S., Singh A., Kumari K. Deflate Compression Algorithm // International Journal of Engineering Research and General Science. 2016. Vol. 4, iss. 1. Pp. 430–436.

13. Design and Optimization of Zstandard Algorithm Based on Concurrent Streaming of Multiple Hash Tables / L. Zheng [et al.] // Proceedings of the Second International Conference on Laser, Optics and Optoelectronic Technology (LOPET 2022) (Qingdao, China, 20–22 May 2022). Proceedings of SPIE. Vol. 12343. Bellingham (WA): Society of Photo-Optical Instrumentation Engineers, 2022. Pp. 571–576. DOI: 10.1117/12.2649516.

14. Эффективное сжатие лог-файлов: анализ производительности LZ4, Zstandard и гибридного подхода / И. Г. Рзун [и др.] // Вестник Академии знаний. 2025. № 4 (69). С. 435–443.

15. Brotli: A General-Purpose Data Compressor / J. Alakuijala [et al.] // ACM Transactions on Information Systems. 2019. Vol. 37, iss. 1. Art. no. 4. 30 p. DOI: 10.1145/3231935.

16. Гаврикова С. В. Обзор баз данных временных рядов // International Journal of Open Information Technologies. 2023. Vol. 11, no. 11. Pp. 83–102.

17. Google BigQuery // Bisong E. Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners. Berkeley (CA): Apress, 2019. Pp. 485–517. DOI: 10.1007/978-1-4842-4470-8_38.

Дата поступления: 23.11.2025
Решение о публикации: 23.01.2026

Comparative Analysis of Modern Lossless Data Compression Methods Used by Cloud Providers

Stepan S. Yudnikov[1] — 2nd year Master's Degree Student in 09.04.02 Information Systems and Technologies. Research interests: information systems, artificial intelligence, methods and algorithms of data compression. E-mail: s.yudnikovv@gmail.com

Anatoly D. Khomonenko[1,2] — Dr. Sci. in Engineering, Full Professor, Professor of the Information and Computing Systems Department; Professor of the Department of Mathematical and Software Engineering. Research interests: information systems, big data processing, probabilistic modeling of geographic information systems, genetic algorithms, information security. E-mail: khomon@mail.ru

[1]Emperor Alexander I St. Petersburg State Transport University, 9, Moskovsky ave., Saint Petersburg, 190031, Russia

[2]Mozhaisky Military Aerospace Academy, 13, Zhdanovskaya str., Saint Petersburg, 197198, Russia

For citation: Yudnikov S. S., Khomonenko A. D. Comparative Analysis of Modern Lossless Data Compression Methods Used by Cloud Providers. *Intellectual Technologies on Transport*, 2026, no. 1 (45), pp. 41–50. DOI: 10.20295/2413-2527-2026-145-41-50. (In Russian)

Abstract. *The exponential growth of data volumes in cloud environments makes it critically important to efficiently compress this data to optimize the use of storage resources, network bandwidth and computational power, which directly affects economic and operational efficiency.* ***Purpose:*** *to conduct a comparative analysis of modern lossless compression algorithms to identify optimal scenarios for their application in cloud platforms.* ***Results:*** *the principles of operation and classification of contemporary lossless compression algorithms have been systematized. Based on a comparative analysis of key parameters, such as compression ratio, operation speed, and resource intensity, optimal application scenarios for the Gzip, LZ4, Zstandard and Brotli algorithms in cloud services have been determined.* ***Practical significance:*** *the results obtained can be used for optimizing costs associated with cloud infrastructure and for improving the performance of distributed data processing systems.* ***Discussion:*** *the conducted analysis demonstrates that the choice of algorithm depends on the specific characteristics of the task. The use of hardware acceleration can significantly improve compression performance.*

Keywords: *data compression, lossless compression, cloud computing, compression algorithms, Gzip, LZ4, Zstandard, Brotli, cloud system performance, cost optimization, network traffic, hardware acceleration*

REFERENCES

1. Verzilin D.N., Maximova T.G., Shanygin S.I. Oblachnye tekhnologii v razvitii institutov tsifrovoy transformatsii rossiyskoy ekonomiki: statisticheskoe issledovanie [Cloud Technologies in the Development of Institutions for Digital Transformation of the Russian Economy: Statistical Research], *Vestnik Sankt-Peterburgskogo universiteta. Ekonomika [St. Petersburg University Journal of Economic Studies]*, 2025, vol. 41, iss. 1, pp. 146–178. DOI: 10.21638/spbu05.2025.107 (In Russian)

2. Davydenko E.A., Sabirgalieva A.M. Arkhivirovanie i szhatie faylov v Linux: metody, algoritmy i ikh effektivnost [Archiving and Compressing Files in Linux: Methods, Algorithms and Their Effectiveness], *Nauchnyy Vestnik Gumanitarno-Sotsialnogo Instituta*, 2025, no. 20, 5 p. (In Russian)

3. Churkin Ya. M. Szhatie algoritmom Khaffmana [Huffman Coding for Data Compression], *Instrumenty i mekhanizmy ustoychivogo innovatsionnogo razvitiya: sbornik statey po itogam Vserossiyskoy nauchno-prakticheskoy konferentsii [Tools and Mechanisms for Sustainable Innovative Development: A Collection of Articles Based on the Proceedings of the All-Russian Scientific and Practical Conference], Samara, Russia, February 06, 2022.* Sterlitamak, Agency of International Research, 2022, pp. 20–22. (In Russian)

4. Levin I.I., Dudnikov E.A. Strukturnaya modifikatsiya metoda Khaffmana dlya szhatiya plotnykh potokov dannykh bez poter na RVS [Structural Modification of The Huffman Method for Compression of Dense Data Streams Without Loss on a RCS], *Izvestiya Yuzhnogo federalnogo universiteta. Tekhnicheskie nauki [Izvestiya Southern Federal University. Engineering Sciences]*, 2024, no. 5 (241), pp. 48–58. DOI: 10.18522/2311-3103-2024-5-48-58. (In Russian)

5. Howard P.G., Vitter J.S. Arithmetic Coding for Data Compression, *Proceedings of the IEEE*, 1994, Vol. 82, iss. 6, pp. 857–865. DOI: 10.1109/5.286189.

6. Dobychin R.S., Petrov A.R. Sravnitelnyy analiz raboty algoritmov Shennona — Fano i Khaffmana [Comparative Analysis of the Shannon — Fano Algorithm and the Huffman Algorithm for Data Coding], *Mezhdistsiplinarnye issledovaniya v oblasti matematicheskogo modelirovaniya i informatiki: materialy 7-y nauchno-prakticheskoy internet-konferentsii [Interdisciplinary Research on Mathematical Modelling and Computer Science: Proceedings of the 7th Scientific and Practical Internet Conference], Togliatti, Russia, March 30–31, 2016.* Ulyanovsk, Zebra Publishing House, 2016, pp. 12–14. (In Russian)

7. Kreft S., Navarro G. LZ77-Like Compression with Fast Random Access, *Proceedings of the 2010 Data Compression Conference, Snowbird, UT, USA, March 24–26, 2010.* Institute of Electrical and Electronics Engineers, 2010, pp. 239–248. DOI: 10.1109/DCC.2010.29.

8. Arroyuelo D., et al. LZ78 Compression in Low Main Memory Space. In: *Fici G., et al. (eds) String Processing and Information Retrieval (SPIRE 2017): Proceedings of the 24th International Symposium, Palermo, Italy, September 26–29, 2017.* Lecture Notes in Computer Science. Vol. 10508. Cham, Springer, 2017, pp. 38–50. DOI: 10.1007/978-3-319-67428-5_4.

9. Dheemanth H.N. LZW Data Compression, *American Journal of Engineering Research*, 2014, vol. 3, iss. 2, pp. 22–26.

10. Faykus M.H., Calhoun J., Smith M. Lossy and Lossless Compression for BioFilm Optical Coherence Tomography (OCT), *SC-W 2023: Proceedings of the Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis, Denver, CO, USA, November 12–17, 2023*. New York, Association for Computing Machinery, 2023, pp. 281–288. DOI: 10.1145/3624062.3625125.

11. Kösters M., et al. PymzML v2.0: Introducing a Highly Compressed and Seekable GZip Format, *Bioinformatics*, 2018, vol. 34, no. 14, pp. 2513–2514. DOI: 10.1093/bioinformatics/bty046.

12. Oswal S., Singh A., Kumari K. Deflate Compression Algorithm, *International Journal of Engineering Research and General Science*, 2016, vol. 4, iss. 1, pp. 430–436.

13. Zheng L., et al. Design and Optimization of Zstandard Algorithm Based on Concurrent Streaming of Multiple Hash Tables, *Proceedings of the Second International Conference on Laser, Optics and Optoelectronic Technology (LOPET 2022), Qingdao, China, May 20–22, 2022*. Proceedings of SPIE. Vol. 12343. Bellingham (WA), Society of Photo-Optical Instrumentation Engineers, 2022, pp. 571–576. DOI: 10.1117/12.2649516.

14. Rzun I. G., et al. Yakhimovich I. S. Effektivnoe szhatie log-faylov: analiz proizvoditelnosti LZ4, Zstandard i gibridnogo podkhoda [Efficient Log File Compression: Performance Analysis of LZ4, Zstandard, and a Hybrid Approach], *Vestnik Akademii znaniy [Bulletin of the Academy of Knowledge]*, 2025, no. 4 (69), pp. 435–443. (In Russian)

15. Alakuijala J., et al. Brotli: A General-Purpose Data Compressor, *ACM Transactions on Information Systems*, 2019, vol. 37, iss. 1, art. no. 4, 30 p. DOI: 10.1145/3231935.

16. Gavrikova S. V. Obzor baz dannykh vremennykh ryadov [Time Series Databases Overview], *International Journal of Open Information Technologies*, 2023, vol. 11, no. 11, pp. 83–102. (In Russian)

17. Bisong E. Google BigQuery. In: *Bisong E. Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners.* Berkeley (CA), Apress, 2019, pp. 485–517. DOI: 10.1007/978-1-4842-4470-8_38.

Received: 23.11.2025
Accepted: 23.01.2026