

Повышение качества дедупликации с помощью Frequency Based Chunking

Дмитриевцев А.С., СПбГУ, Санкт-Петербург a.dmitriyevtsev@spbu.ru,
Гориховский В.И., СПбГУ, Санкт-Петербург gorihovskyvyacheslav@gmail.com

Аннотация

В данной работе предлагается параллельная версия алгоритма улучшения качества дедупликации Frequency Based Chunking. Использование вычисления частот в нескольких потоках позволяет намного улучшить скорость записи в файловой системе, поддерживающей такой подход. Также в работе рассматривается влияние коллизий, возникающих при одновременном анализе чанков.

Введение

Дедупликация данных – подход к сжатию данных, направленный на устранение повторяющихся копий на одном или группе носителей с целью снижения накладных расходов на хранение и передачу информации, широко используется в системах хранения резервных копий.

Content-Defined Chunking (CDC) – метод разделения на блоки для первичной дедупликации, в котором данные используются для определения границ будущих чанков. Для улучшения дедупликации существует подход Frequency Based Chunking[1], который основывается на анализе частот подчанков в данном CDC-разбиении и производит его измельчение. Такой подход позволяет значительно увеличить коэффициент дедупликации, однако на порядок замедляет скорость записи.

В представленной работе предлагается метод оптимизации FBC, основанный на параллельном вычислении частот. Этот метод интегрирован в ChunkFS[2] – модельную файловую систему, предназначенную для комплексного анализа методов дедупликации. Основной целью работы является исследование возможности применения FBC в системах резервного копирования, в особенности TATLIN.BACKUP от компании YADRO.

Алгоритм FBC

Этапы работы Frequency Based Chunking представлены на 1. Скользящее окно проходит по чанкам, полученным после работы CDC-алгоритма. При

нахождении частого подчанка, он добавляется в словарь. На основе частотного анализа проводится дальнейшее дробление: некоторые исходные CDC-чанки разделяются на несколько новых чанков меньшего размера, которые были найдены на предыдущем этапе. Таким образом, вследствие измельчения чанков, мы находим большее число дубликатов, что приводит к повышению качества дедупликации.

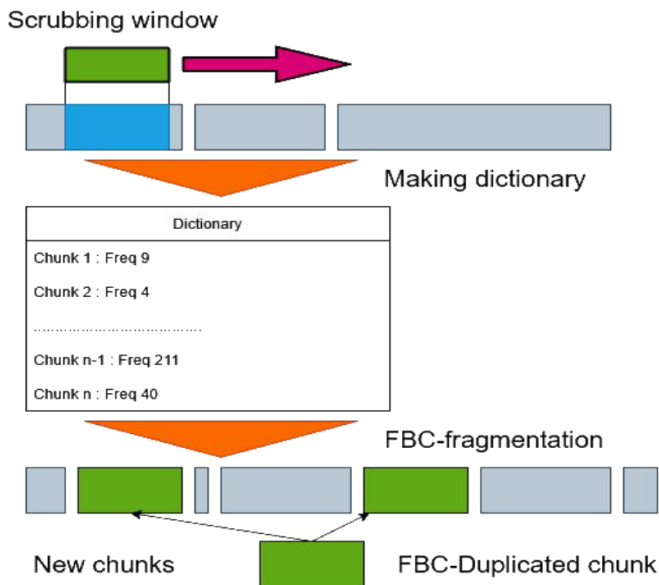


Рис. 1: Этапы алгоритма Frequency Based Chunking

Специфика интеграции

Ранее была реализована и интегрирована в ChunkFS базовая[3] версия алгоритма. Проведенный комплексный анализ эффективности FBC, выявил ряд особенностей алгоритма:

- измельчение чанков приводит к вплоть до стократному снижению скорости записи
- системы хранения резервных копий предполагают редкие, но крупные периоды записи данных

- запись может производиться независимо от чтения уже размещенных данных
- FBC – скраббирующий алгоритм, но при этом требуется существенное улучшение производительности записи
- чтение может легко выполняться параллельно, так как не является деструктивной операцией

Одной из наиболее трудоемких операций в алгоритме FBC является вычисление частот, поэтому наиболее естественным улучшением производительности является их параллельное вычисление, не предполагавшееся в оригинальной работе.

Параллельный подход к FBC

Алгоритм проходит чанки скользящим окном, периодически добавляя в словарь новые записи о наиболее частых подчанках из данного разбиения. Данные действия можно выполнять параллельно, обеспечив конкурентный доступ к словарю частых чанков, однако подобный подход к вычислению частот предвещает недостаток – при параллельной записи могут возникать коллизии. Источником коллизий является одновременное возникновение нового подчанка в разных потоках, затрудняющее корректное определение частоты этого подчанка. Из-за этого может не произойти разбиение самих чанков. Однако выдвигается предположение, что коллизии несущественно влияют на качество дедупликации. Если подчанк действительно частый, он встретится в разных потоках неоднократно и будет добавлен в словарь. Чанки, уже находящиеся в FBC словаре, повторно в него не добавляются. Поэтому не будут участвовать в подразбиении только относительно редкие подчанки.

Для решения задачи необходимо обеспечить эффективный конкурентный доступ к словарю FBC-чанков. Был проведен анализ эффективности различных реализаций: `Mutex<HashMap>`, `RwLock<HashMap>`, `DashMap`

Наибольшую эффективность показал вариант с `DashMap`, поэтому выбор был сделан именно в пользу данной реализации.

Экспериментальное исследование

Было проведено экспериментальное исследование, направленное на оценку производительности решения и влияния коллизий на качество дедупликации.

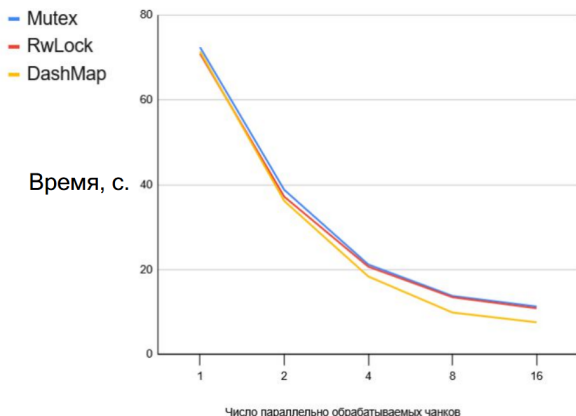


Рис. 2: Скорость вычисления частот для разных вариантов синхронизации, на фрагменте 20% от Enron

Постановка эксперимента

Исследовательский вопрос 1 - Как влияет изменение числа потоков на качество дедупликации?

Исследовательский вопрос 2 - Насколько эффективно параллельное вычисление частот чанков?

Для исследования были выбраны два датасета: Enron emails (1,4 Гб), две версии ядра Linux в совокупности (3.4.6 и 3.4.7, 900 Мб). Enron emails содержит малое количество дубликатов крупного размера и предназначен для определения улучшения качества дедупликации в сравнении с базовыми CDC алгоритмами. Ядра Linux в свою очередь содержат большое количество дубликатов и лучше выделяют коллизии.

В качестве метрик были выбраны: время записи целого датасета в секундах и коэффициент дедупликации (считается как начальный размер данных, деленный на размер данных после обработки). Исследовались запуски на 1, 2, 4, 8 и 16 потоках, в которых весь датасет загружался в словарь и вычислялось время, затраченное на составление словаря, и объем такого словаря. Такие запуски проводились по 10 раз, из которых выбирались худшие, средние и лучшие результаты.

Исследования проводились в системе со следующими характеристиками: ОС Linux, дистрибутив Manjaro 24.1.1, процессор Intel® Core™ i7-12650H,

объем оперативной памяти 16 Гиб.

Результаты экспериментального исследования

На 4 приведены результаты влияния коллизий на качество дедупликации, для датасета Enron emails слева и ядер Linux справа. С ростом числа потоков наблюдается снижение коэффициента дедупликации за счет коллизий. При этом большее снижение заметно на датасете Enron emails. Это можно объяснить малым количеством дубликатов в этом датасете и, соответственно, меньшей частотой встречи подчанков. В датасете с ядрами Linux более вероятная последующая повторная встреча подчанка может нивелировать коллизию. Отсюда можно сделать вывод, что коллизии не оказывают существенного влияния на качество дедупликации: около 1-5% при записи новых чанков. Что значительно меньше выигрыша от использования FBC: около 30-40%[3].

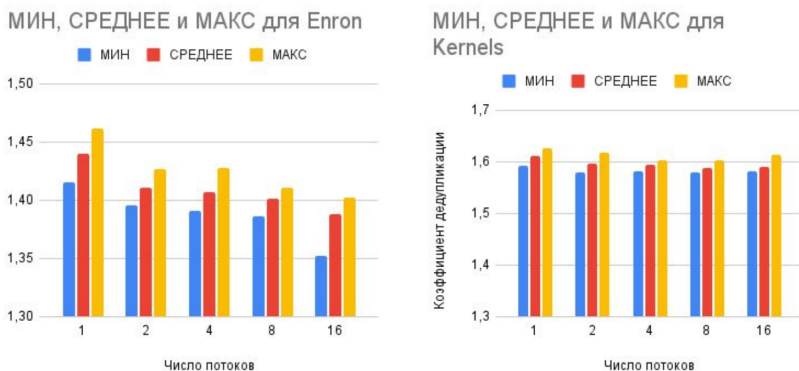


Рис. 3: Коэффициент дедупликации при параллельном исполнении

Результаты оценки времени составления словаря представлены на 4. Разница между лучшими и худшими показателями незначительна (не превышает 2%), поэтому на графике приведены только средние значения. Параллельная реализация показала существенный прирост в скорости составления словаря частых чанков, вплоть до 10 раз на 16 потоках. Однако, возможно, прирост эффективности между использованием 8 и 16 потоков является уже не настолько существенным, чтобы тратить столько ресурсов, которые можно было бы использовать на оптимизацию других частей ChunkFS. Например, на оптимизацию скорости записи чанков в хранилище. Более того, использование 16 потоков приводит к снижению коэффициента дедупликации ввиду

возникновения большого количества коллизий.

Время работы алгоритма

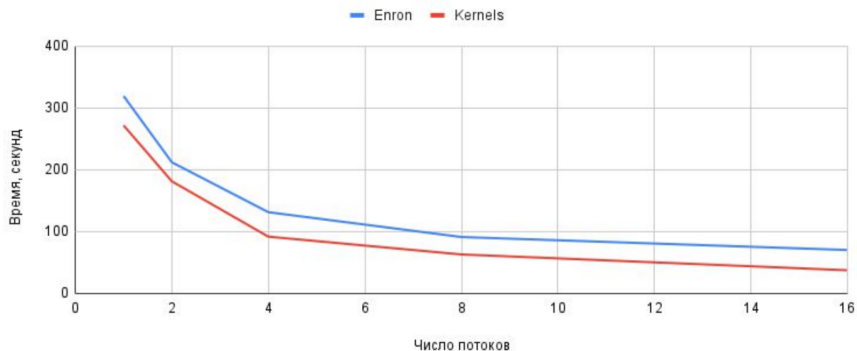


Рис. 4: Время на составление словаря для используемых датасетов

Для систем хранения резервных копий мы можем рекомендовать использовать параллельную реализацию на 8 потоках. Более того, для снижения количества коллизий можно предложить проводить запись первой версии последовательно. Это может уменьшить количество коллизий и снизить ухудшение качества дедупликации от использования параллелизма. Такой подход не ухудшит производительность регулярного использования системы, а только скорость ее инициализации.

Заключение

В работе предложен подход к распараллеливанию алгоритма FBC. Параллельное вычисление частот позволило достичь скорости алгоритма, сравнимой с базовыми CDC алгоритмами, при практически сохранении улучшения качества дедупликации. Также на основе экспериментального исследования предлагается эвристическое решение для использования в системах хранения резервных копий. Оно позволяет снизить количество коллизий при вычислении частот за счет последовательной записи первой версии и параллельной записи последующих.

Список литературы

- [1] Lu, Guanlin & Jin, Yu & Du, David. (2010). Frequency Based Chunking for Data De-Duplication. 287 - 296. 10.1109/MASCOTS.2010.37. <https://ieeexplore.ieee.org/document/5581583>
- [2] Piletskii O., Gorikhovskii V. (2025). ChunkFS: A Tool for Data Deduplication Methods Comparison. 221 - 227. Proceedings of FRUCT'37.
- [3] Дмитриевцев А. С., Гориховский В. И. Повышение качества дедупликации с помощью Frequency Based Chunking. CFP: сборник материалов конференции, Санкт-Петербург, 23–24 апреля 2025 года. Санкт-Петербургский политехнический университет Петра Великого. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2025. – С. 307-309.