Санкт-Петербургский государственный университет

Системное программирование

Сравнение методов дедупликации данных

Пилецкий Олег Антонович, группа 21.Б10-мм

Научный руководитель: доцент кафедры системного программирования, к. ф.-м. н., Гориховский В. И.
Рецензент: ведущий разработчик по разработке ПО, ООО «Ядро Центр Программных Разработок», Ли С. В.

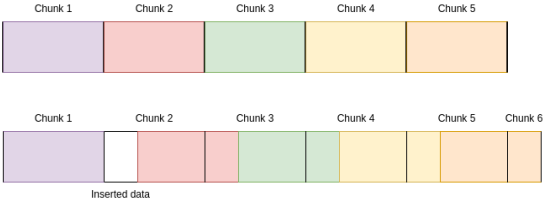
Санкт-Петербург
2025

Введение

- Резервное копирование данных — важный инструмент для защиты информации
- Системы резервного копирования требуют больших затрат на хранения
- Значительная часть данных — дубликаты
- Классическое решение проблемы — дедупликация
  - Чанкирование
  - Хэширование
  - Удаление дубликатов
  - Индексирование
- TATLIN.BACKUP
- Многообразие алгоритмов/мало инструментов для сравнения
- Требуется интеграционное исследование алгоритмов

Существующие методы чанкирования

- Fixed Size Chunking
  - Очень быстрый
  - Мало дедуплицирует, проблема сдвига
- Content Defined Chunking
  - Семейство алгоритмов
  - Делит массив данных на чанки в зависимости от содержимого
  - Работают медленнее, но дедуплицируют многократно лучше



Постановка задачи

Цель работы: разработка системы интеграционного бенчмаркинга методов дедупликации: ChunkFS

Задачи:

- Провести обзор существующих алгоритмов CDC и систем сравнения
- Разработать систему бенчмаркинга алгоритмов дедупликации
- Реализовать базу современных алгоритмов CDC
- Провести интеграционный анализ и сравнение качества алгоритмов CDC

Обзор существующих инструментов

- DedupBench
  - Набор скриптов на C++, Bash и Python
  - Есть реализации современных алгоритмов, сбор метрик
  - Нет поддержки разных хранилищ, повторной записи
  - Ограниченный комплект наборов данных
- CDC Algorithm Tester
  - Набор скриптов на Rust, Bash и R
  - Содержит реализации иных алгоритмов
  - Чанкирование производится только в оперативную память
  - Имеет CLI-инструмент для прогона сценариев

Единственнный сценарий работы инструментов:

- Прогнать набор данных через основную программу
- Получить набор хэшей
- Проанализировать этот набор отдельными скриптами

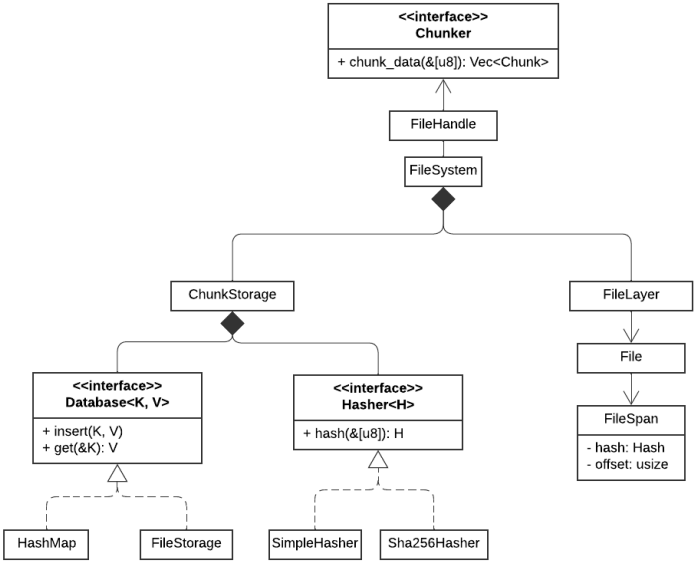
Недостатки существующих инструментов

- Сложность в модификации сценариев использования алгоритмов
- Отсутствие поддержки дисковых хранилищ
- Сложность сбора метрик
- Невозможность проведения интеграционного тестирования алгоритмов
- Оценка лишь скорости самого алгоритма чанкирования

Требования к системе сравнения

- Модельная файловая система на Rust
- Взаимозаменяемость алгоритмов чанкирования и хэширования
- Поддержка различных видов key-value хранилищ
- Бенчмаркинг записи и чтения файлов
- Быстрое конфигурирование экспериментов
- Поддержка методов повышения качества дедупликации

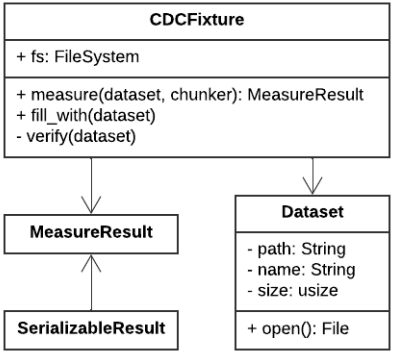
Архитектура ChunkFS



Сбор метрик

Запуск сценариев и сбор метрик — CDCFixture

- Экземпляр файловой системы внутри
- Возможность запустить измерение и предзаполнить файловую систему
- Измерения можно проводить несколько раз
- В каждом прогоне — верификация записанных данных
- Подключение при помощи features = "bench"



Инструмент командной строки

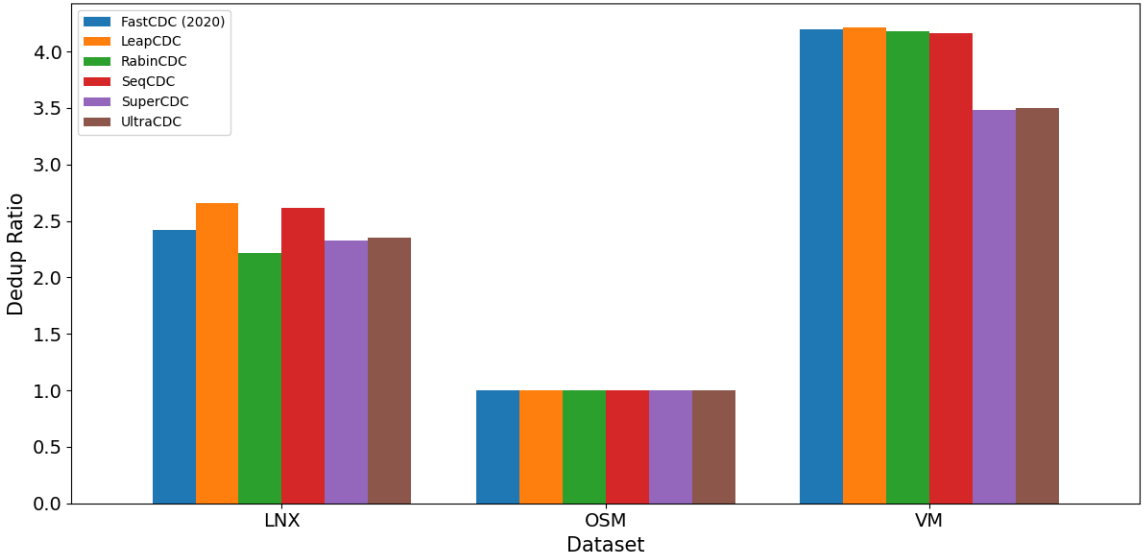
Был создан CLI-инструмент для быстрого запуска сценариев

- Параметры задаются через аргументы
- Возможность использовать конфигурационный файл вместо аргументов
- Возможность предзаполнить файловую систему данными
- Запись отчета на диск в указанную папку

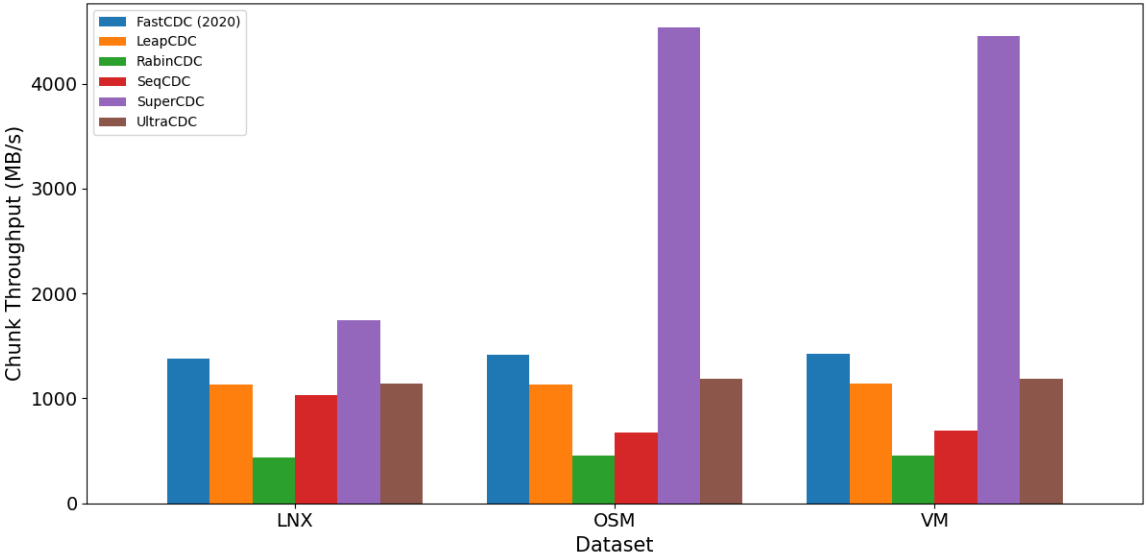
Эксперимент

- Тестовые сценарии
  - Запись разными чанкерами в пустое хранилище
- Метрики
  - Скорость записи и чтения из файловой системы
  - Отдельно: скорость чанкирования
  - Коэффициент дедупликации
- Наборы данных (типичные для бэкапов)
  - Ядра Linux разных версий (LNX) — среднее число дубликатов
  - Бэкапы OpenStreetMaps (OSM) — нет дубликатов
  - Образы виртуальных машин (VM) — много дубликатов
- Алгоритмы: RabinCDC, LeapCDC, FastCDC, UltraCDC, SuperCDC, SeqCDC
  - Реализованы в отдельном пакете cdc-chunkers
- Конфигурация: Intel Core i5-1240P, 40 ГБ RAM

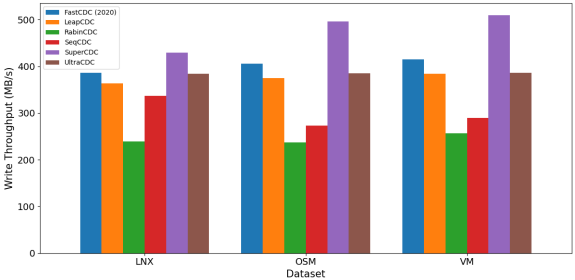
Коэффициент дедупликации

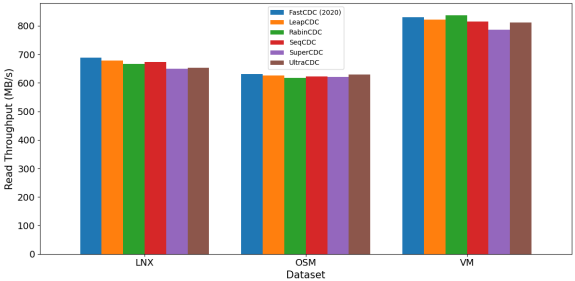


Пропускная способность чанкеров



Пропускная способность записи и чтения в ChunkFS





Заключение

В результате работы:

- Был проведен обзор алгоритмов чанкирования и существующих систем сравнения
- Были разработаны модельная файловая система ChunkFS и реализации алгоритмов. Код реализации: https://github.com/Piletskii-Oleg/chunkfs
- Была разработана система сравнения и бенчмаркинга алгоритмов на ее основе
- Было проведено сравнение алгоритмов при помощи разработанного инструмента

Результаты работы были представлены на конференции FRUCT и опубликованы: https://ieeexplore.ieee.org/document/11007956.

В рамках учебных практик участники проекта занимаются исследованиями в сфере методов оптимизации дедупликации: добавлены FBC и SBC, key-value хранилища на основе LSM и B+ деревьев. ChunkFS была использована на зимней школе YADRO 2025 для бенчмаркинга дедупликации на платформах RISC-V.

Количество чанков/Средний размер

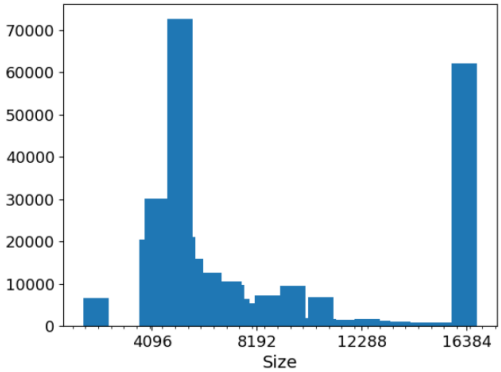
	Rabin	Ultra	Super	Leap	Fast	Seq
Linux	14199	11173	7124	8515	11559	6481
VM	12766	9643	5889	7916	10512	6935
OSM	12732	9766	5865	7881	10483	7143
Nodejs	12271	13148	6384	8436	10998	8567
Wordpress	12250	11759	6550	8576	11221	8497

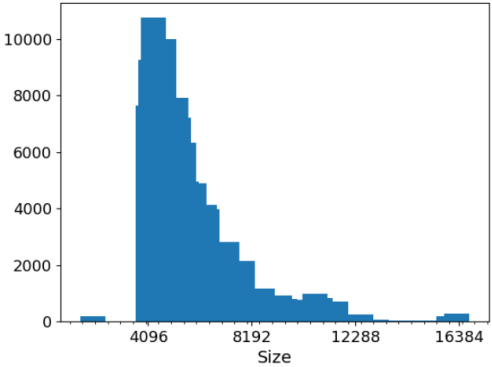
Таблица: Средний размер чанка

	Rabin	Ultra	Super	Leap	Fast	Seq
Linux	392791	470397	698865	545935	442505	710974
VM	75643	119543	196263	120873	91370	139339
OSM	666777	869310	1447308	1077075	809766	1188360
Nodejs	133895	130498	257096	185195	146949	186991
Wordpress	31431	32861	60017	43547	34044	44593

Таблица: Количество уникальных чанков

Графики распределения для SuperCDC





Запись/чтение в rocksdb

	Rabin	Ultra	Super	Leap	Fast	Seq
Linux	**46 ± 2**	39 ± 5	37 ± 2	40 ± 2	39 ± 2	38 ± 2
VM	40 ± 10	**45 ± 1**	**45 ± 1**	44 ± 3	**45 ± 3**	41 ± 1
OSM	21 ± 3	**23 ± 1**	**23 ± 1**	**23 ± 2**	21 ± 1	18 ± 6
Nodejs	**77 ± 39**	61 ± 6	49 ± 17	48 ± 14	60 ± 2	61 ± 4
Wordpress	88 ± 28	136 ± 34	133 ± 10	147 ± 36	**160 ± 20**	160 ± 27

Таблица: Скорость записи в ChunkFS (rocksdb), МБ/с

	Rabin	Ultra	Super	Leap	Fast	Seq
Linux	**269 ± 43**	228 ± 53	233 ± 28	249 ± 30	220 ± 42	190 ± 30
VM	**421 ± 80**	323 ± 31	272 ± 43	313 ± 20	350 ± 38	329 ± 26
OSM	**204 ± 40**	154 ± 14	168 ± 46	180 ± 27	177 ± 40	162 ± 26
Nodejs	**203 ± 27**	169 ± 13	152 ± 20	174 ± 50	192 ± 19	182 ± 10
Wordpress	395 ± 11	**417 ± 39**	342 ± 19	389 ± 17	390 ± 12	363 ± 15

Таблица: Скорость чтения в ChunkFS (rocksdb), МБ/с