Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа программной инженерии

КУРСОВОЙ ПРОЕКТ

"Исследование подходов к созданию высокопроизводительного, масштабируемого сервиса для дедупликации данных в хранилище"

по дисциплине
«Системы анализа больших данных»

А.Д. Ковалев, И.В. Никифоров

Санкт-Петербург

2020

Содержание

Введение 3
Цель и задачи выполняемой работы 4
Алгоритм работы системы 4
Рекомендации к выполнению 5
Источник данных 5
Генератор данных 5
Хранилище данных 5
Хранилище таблиц hash-значений и ссылок 5
Алгоритм вычисления hash 5
Результаты выполнения работы 5
Дополнительно 6

Введение

Современный тренд в области хранения данных направлен на создание распределенных хранилищ, которые могут быть хорошо масштабируемы, обладают высокой надежностью, безопасностью, производительностью и позволяют хранить и обрабатывать колоссальный объем входных данных. Такие распределенные хранилища размещают в облачной инфраструктуре, которая в свою очередь обладает огромным потенциалом для создания и использования различных сервисов обслуживания данных.

Одним из важных аспектов хранения является минимизация стоимости хранения данных. Для неструктурированных данных или данных, которые представляются в виде последовательности байт на диске, для оптимального хранения можно использовать подходы дедупликации. В данной работе предлагается создать прототип решения по дедупликации данных на локальном хранилище. Под дедупликацией данных понимается физическое хранение на носителе информации (жестком диске) только уникальных блоков данных.

Любой алгоритм дедупликации основан на вычислении hash-функции для блоков входного потока данных. На основе этой hash-функции делается решение, существует ли уже в системе входной блок данных или нет. Если блок не существует, то он записывается в хранилище, а если существует, то он заменяется ссылкой на уже существующий в хранилище.

Для того, чтобы сделать вывод о том, существует ли в системе блок данных с заданным hash-значением, необходимо хранить таблицу hash-значений и ссылку в хранилище на соответствующий блок данных.

Hash-таблица в общем случае может содержать две колонки: "Hash-значение", "Ссылка на блок данных". Но стоит иметь ввиду, что зачастую приходится хранить дополнительную информацию, необходимую для дедупликации и сборщика мусора, например: количество ссылок на блок данных.

Цель и задачи выполняемой работы

Цель работы состоит в реализации системы оптимального хранения данных за счет использования подхода дедупликации данных и проведении тестирования для измерения производительности созданного прототипа.

Задачи, которые требуется решить для достижения цели:

- изучить подходы к оптимизации хранения данных в традиционных базах данных;
- выбрать стек технологий, необходимый для создания прототипа системы дедупликации;
- разработать систему оптимального хранения данных на диске;
- провести нагрузочное тестирование и установить зависимость скорости чтения/записи данных в локальное хранилище в зависимости от размера сегмента;
- установить оптимальный размер сегмента блока данных, при котором наблюдается:
    - максимальная скорость записи
    - максимальная скорость чтения
    - процент ошибки восстановления данных в зависимости от использованного алгоритма hash.

Алгоритм работы системы

В работе можно следовать следующему алгоритму дедупликации неструктурированных данных:

- входной поток данных разделяется на блоки (сегменты) заданного объема;
- для каждого блока вычисляется hash;
- значение hash проверяется с таблицей уже существующих hash для данных, находящихся на носителе;
- если значение hash для данных уже есть, то для входного потока сохраняется лишь ссылка на уже существующие данные;
- если данных нет, то новый блок сохраняется на диске и hash этого блока записывается в таблицу с соответствующим расположением.

Рекомендации к выполнению

Источник данных

Может быть любым. Например, источник данных может представлять из себя локальный файл (текстовый, видео, музыка и т.д.) или же источником данных может выступать сетевой поток данных, получаемый с помощью REST и т.д.

Генератор данных

Считывает данные из источника данных и разделяет поток на сегменты. Размер сегмента необходимо задавать в настройках к системе. Рекомендуется для начала размер сегмента задать равным 4 байта.

Хранилище данных

В качестве хранилища данных в работе предлагается использовать локальные файлы. При этом каждый файл хранилища может содержать несколько сегментов данных. Необходимо продумать разделение сегментов внутри одного файла, например каждый сегмент можно записывать с новой строки или придумать другой разделитель.

Хранилище таблиц **hash-**значений и ссылок

Можно выбрать любую доступную базу данных, например, MongoDB, PostgreSQL, Redis, HBase и т.д. Предполагается, что таблица hash-значений будет содержать не менее 4 столбцов, а именно:

1. hash-значение сегмента данных
2. название файла, где содержится сегмент
3. позиция в файле (номер строки), где содержится сегмент
4. количество повторений данного сегмента

Алгоритм вычисления **hash**

Можно выбрать любой алгоритм, например, MD5, SHA128, SHA256, SHA512, tiger16, tiger32, murmur3_126 и т.д.

Результаты выполнения работы

В конце семестра студент должен:

- продемонстрировать программную реализацию прототипа системы;

- предоставить отчет с полным описанием разработанной системы.

Отчет по курсовому проекту должен содержать:

- выбранный стек технологий;
- подробное описание архитектуры системы;
- программную реализацию;
- результаты тестирования.

Дополнительно

Стоит обратить внимание на возможности компрессии, сжатия hash-таблицы, так как в общем случае она может занимать значительный объем памяти. Также можно обратить внимание на шифрование данных.