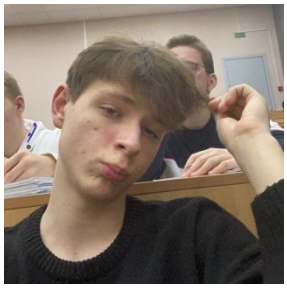
УДК 004.6

ФОРМАТЫ ФАЙЛОВ БОЛЬШИХ ДАННЫХ: ХРАНЕНИЕ ДАННЫХ В ЭКОСИСТЕМЕ HADOOP

И.Д. Стаселько
Студент БГУИР

А.Ю. Сычев
Студент БГУИР



А.П. Протасов
Студент БГУИР

Ю.И. Алексеев
Ассистент кафедры Информатики, магистр технических наук

Т.Д. Позняков
Студент БГУИР

Аннотация. В данной статье мы рассматриваем одну из технологий хранения данных Big Data. Hadoop.Hadoop предлагает один из самых экономичных и эффективных способов хранения данных в огромных объемах . Структурированные, полуструктурированные и неструктурированные типы данных могут храниться и затем обрабатываться с использованием таких инструментов, как Pig ,Ulive и Spark.Также описываются различные форматы хранения данных и происходит сравнение форматов файлов Big Data

Ключевые слова: Hadoop, Big Data, Spark.

Введение. Big data — это различные инструменты, подходы и методы обработки как структурированных, так и неструктурированных данных для того, чтобы их использовать для конкретных задач и целей. В современном мире Big data—социально-экономический феномен, который связан с тем, что появились новые технологические возможности для анализа огромного количества данных. Big Data использует различные технологии хранения данных, в данной статье мы рассмотрим поближе технологию Hadoop.

Hadoop предлагает один из самых экономичных и эффективных способов хранения данных в огромных объемах. Кроме того,структурированные, полуструктурированные и неструктурированные типы данных могут храниться и затем обрабатываться с использованием таких инструментов, как Pig ,Ulive и Spark для получения результатов, необходимых для любого будущего анализа и визуализации. Hadoop позволяет пользователю хранить данные в своем репозитории несколькими способами. Некоторые из общедоступных и понятных рабочих форматов включают XML , CSV и JSON .Хотя JSON, XML и CSV являются удобочитаемыми форматами, но не являются лучшим способом хранения данных в группе Hadoop. Фактически, в некоторых случаях хранение данных в таких необработанных форматах может оказаться крайне неэффективным.Кроме того, параллельное хранение невозможно для данных, хранящихся в таких форматах. Ввиду того, что эффективность

хранения данных и параллелизм являются двумя ведущими преимуществами использования Hadoop, использование необработанных форматов файлов может быть вариантом разрешения поставленной задачи.CERN выбрал четыре технологии-кандидата: ORC , Parquet , Kudu и Avro для этой цели. Однако мы включили к этому другие форматы файлов данных, такие как Arrow и текстовые форматы файлов данных. На рис. 2 показана классификация и иерархия форматов файлов больших данных. Каждый из этих форматов файлов имеют ряд преимуществ и недостатков. В этом разделе рассматриваются эти аспекты файла больших данных и обсуждаются критерии, которые должны использоваться для выбора формата файла.

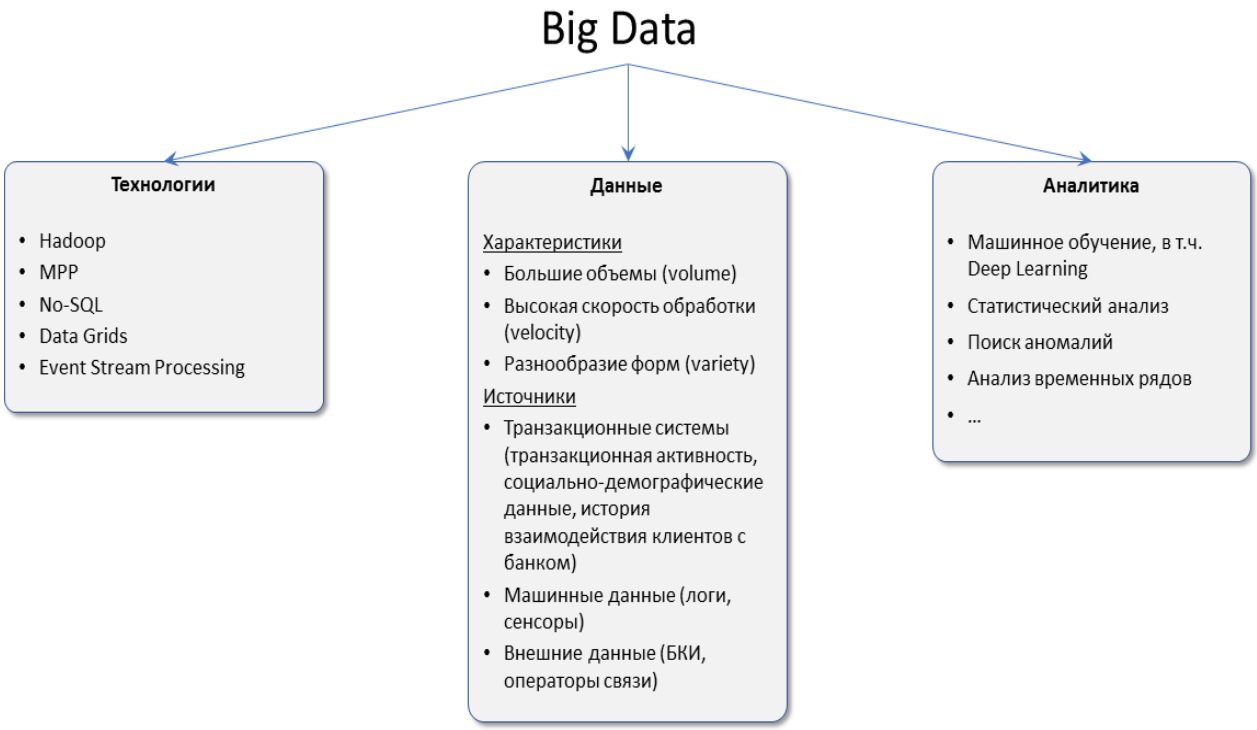


Рисунок *1.* – Распределение Больших Данных

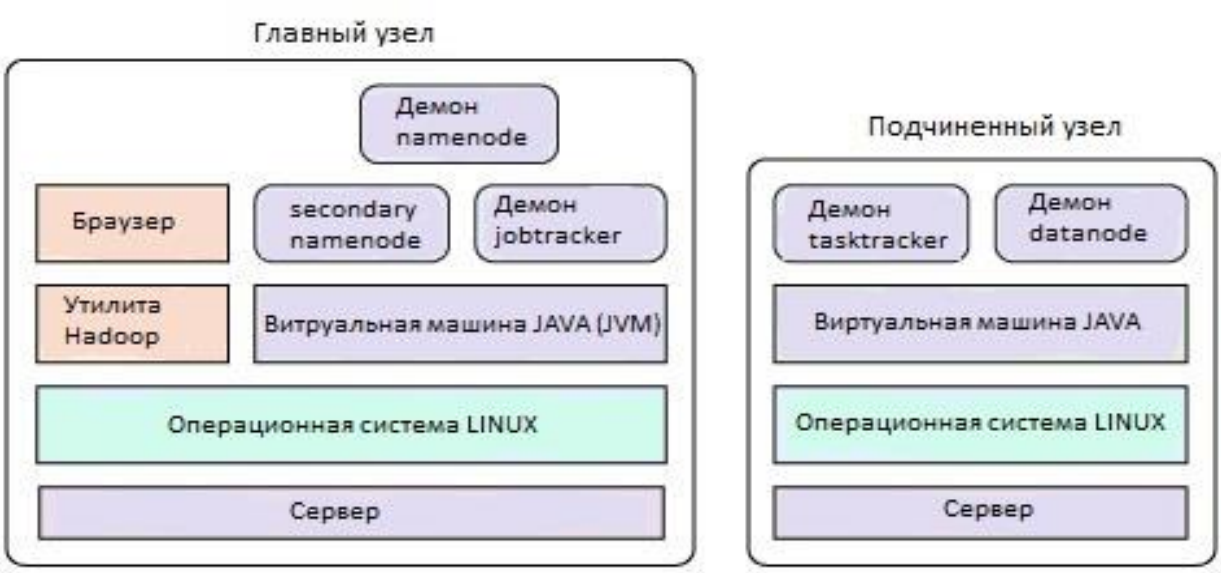


Рисунок *2.* – Классификация и иерархия форматов файлов Больших Данных.

Текстовые форматы.Простейшим примером такого формата файла данных являются записи, хранящиеся построчно, оканчивающиеся символом новой строки или перевод каретки. Для сжатия данных необходимо использовать кодек сжатия на уровне файла, такой как BZIP2, а также необходимо использовать три формата, а именно текст или CSV , JSON и XML доступные в этой категории.

В этой категории доступны форматы, а именно текст или CSV , JSON и XML .

Открытый текст.Данные, хранящиеся в этом формате, в основном форматируются таким образом, что поля имеют фиксированную ширину или разделитель. Отдельные записи в формате CSV разделяются запятыми.

XML.Можно использовать определения внешних схем в XML. Однако производительность сериализации и десериализация, как правило, является плохой.

JSON.Нотация объектов JavaScript (JSON) эффективнее XML, но существует проблема с производительностью сериализации и десериализации.

Основанные на ряде форматы

SequenceFile.Этот формат файла поддерживается, как часть инфраструктуры Hadoop, в составе которой большой файл имеет контейнер двоичных пары "ключ-значение" для хранения нескольких файлов небольшого размера. Поэтому пары "ключ-значение", соответствующие записям, являются закодированными. Формат файла данных SequenceFile легко интегрируется с Hadoop ввиду того, что первый был разработан для второго[2].

Avro. Apache Avro используется для компактного двоичного формата в качестве стандарта сериализации данных. Как правило, используется для хранения постоянных данных, относящихся к протоколам связи и HDFS . Одно из главных преимуществ использования Apache Avro - высокая производительность приема, что связано с быстрой и легкой десериализацией и сериализацией, при том же условии. Важно отметить, что у Avro нет внутреннего индекса. Однако для облегчения произвольного доступа к данным можно использовать метод разбиения по указанию, доступный в HDFS. Алгоритмы сжатия данных, поддерживаемые Apache Avro, включают DEFLATE и Snappy.

Существуют и другие структуры сериализации и десериализации, такие как буферы Furft и Protocol соревнующиеся с Avro. Однако Avro является встроенным компонентом Hadoop, в то время, как эти фреймворки являются внешними. Кроме того, определение схемы в Avro выполняется с использованием JSON, в то время, как буферы Trafft и Protocol зависят от интерфейса языков определения (IDLs) для определения схемы.

Основанные на колонке форматы:

Формат файла столбца записи (RC).Это самый примитивный формат записи в столбце, который был создан в рамках проекта Apache Hive. RCFile двоичный формат, подобный SequenceFile, который гарантирует высокое сжатие для операций, которые включают несколько рядов столбцов , которые сохраняются, как записи в виде столбцов путем создания разделений строк. Вертикальные секции создаются в строке.

Разбиение на столбцы. Метаданные сохраняются для каждой строки, разделенной как ключ, и соответствующие данные сохраняются, как его стоимость.

Формат оптимизированного столбца строк (ORC).Оптимизированный столбец строк похож на Parquet и RCFile в том смысле, что все эти три формата файлов данных существуют в рамках программы Hadoop. Производительность чтения лучше. Однако скорость записи ниже среднего, кроме этого, возможности кодирования и сжатия этого формата файла лучше, чем у его аналогов с поддержкой ORC Zlib и Snappy .

Parquet. Apache Parquet - это стандарт сериализации данных, который ориентирован на столбцы и, как известно, значительно повышает эффективность. Этот формат данных также включает оптимизации, такие, как сжатие ряда значений. Такая же колонка приводит к

улучшенным соотношениям сжатия. Кроме этого, имеются кодировки, такие как bit packing, dictionary и run

Доступно дополнительное кодирование длины. Для сжатия данных поддерживаются алгоритмы Snappy и GZip.

CarbonData.Этот формат данных был разработан компанией Huawei для устранения существующих недостатков в уже имеющихся форматах.

CarbonData – это относительно новый формат файла данных, который позволяет разработчикам использовать преимущества формата ориентированного на столбцы, но при этом иметь возможность обработки запросов произвольного доступа. Данные сгруппированы в блокноты, т.е. хранятся наряду с другой информацией о данных, такой как схема, индексы и смещения[3]. Метаданные хранятся в верхних и нижних колонтитулах, что обеспечивает значительную оптимизацию производительности во время сканирования и обработки последующих запросов.

Форматы,хранящиеся в памяти

Apache Arrow - платформа для разработки приложений с использованием данных в памяти. Более того, она работает на всех языках, что делает ее стандартом для формата столбчатой памяти, обеспечивая поддержку, как иерархических, так и плоских данных. Данные организованы для обеспечения высокопроизводительной аналитики на современном оборудовании. Коммуникация между процессами и потоковая передача работает с нулевой копией или без десериализации и сериализации[5]. Кроме этого, она предоставляет много вычислительных библиотек для решения сложных задач.

Услуги хранения данных

Kudu - это система хранения данных, обеспечивающая масштабируемость и основанная на концепции распределенной системы хранения данных. Данные хранятся внутри таблиц. Кроме того, этот формат данных обеспечивает оптимизированный компромисс между производительностью и скоростью приема, который связан с организацией данных в столбцах и поддержанием индекса. Алгоритмы сжатия данных, поддерживаемые Apache Kudu, включают LZ4, Zlib и Snappy.

Сравнение форматов файлов Big Data

Решение о выборе формата файла приложения зависит от варианта использования или алгоритма, используемого для обработки данных. Следовательно, выбранный формат файла должен соответствовать некоторым основным требованиям, которые будут считаться подходящими для систем Big Data. Формат приложения зависит от варианта использования или алгоритма, используемого для обработки данных. Следовательно, выбранный формат файла должен соответствовать некоторым основным требованиям, которые будут считаться подходящими для систем больших данных.

Во-первых, выбранный формат файла больших данных должен быть выразительным и четко определенным. Во-вторых, он должен поддерживать различные структуры данных. Некоторые из основных структур, которые должны поддерживаться - это структуры, карты, числа, строки, записи и массивы. И наконец, формат файла больших данных должен быть двоичным, простым и обеспечивать поддержку сжатия.

Одним из самых важных требований в приложениях, связанных с HDFS, которые используют такие технологии, как Spark и Hadoop является сокращение времени чтения и записи данных[1]. Проблемы, такие, как ограничения хранилища, эволюция схем и большие данные еще больше усложняют системные требования. Для устранения этих проблем в доменах приложений были созданы несколько форматов файлов больших данных.

Использование соответствующего формата файла может принести следующие преимущества для системы:

1. Время чтения уменьшается.
2. Время записи сокращается.

3. Файлы могут быть разделены, что, другими словами, означает, что больше нет необходимости читать весь файл для получения меньшего его подраздела.

4. Существует поддержка эволюции редактирования схемы, и схема может быть изменена по запросу в зависимости от изменения потребностей системы.

5. Имеются усовершенствованные кодеки сжатия для обеспечения возможности сжатия файлов без потери преимущества базового формата.

Заключение. С учетом вышеупомянутых преимуществ выбор правильного формата файла данных может существенно оптимизировать производительность системы. Однако, в этом отношении имеется множество нюансов. В то время, как некоторые форматы файлов разработаны для общего использования, есть другие форматы, которые предлагают преимущества оптимизации для конкретных приложений или улучшают конкретные особенности. Поэтому для облегчения принятия решения о том, какой файл данных лучше всего подходит для приложения, необходимо провести предварительный анализ и сравнение форматов с учетом конкретных требований.

Список литературы

[1.] Mishra, R. K. (2018). The Era of Big Data, Hadoop, and Other Big Data Processing Frameworks. In PySpark Recipes (pp. 1-14). Apress, Berkeley, CA.

[2.] Bansal, K., Chawla, P., & Kurle, P. (2019). Analyzing Performance of Apache Pig and Apache Hive with Hadoop. In Engineering Vibration, Communication and Information Processing (pp. 41-51). Springer, Singapore.

[3.] Asemota, E., Gallagher, S., Mcrobbie, G., & Cochran, S. (2018). Defining case based reasoning cases with xml

[4.] K. Watanabe, T.Fukamachi, N.Ubayashi and Y.Kamei, Poster: Automated A/B Testing with Declarative Variability Expressions, IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), IEEE 2017”Supercell Case Study”. [Online]. Available: https://aws.amazon.com/solutions/casestudies/supercell/ [Accessed: 29-Nov-2017].

[5.] Gallego, J. J. C., & Fernández-Bermejo, M. D. M. (2018). Analysis of the impact of file formats for open data analytics efficiency: a case study with R. GSTF Journal on Computing (JoC)

BIG DATA FILE FORMATS:STORING DATA IN THE HADOOP ECOSYSTEM

I.S. Staselko
Student of BSUIR

A.U. Sichev
Student of BSUIR

A.P. Protasov
Student of BSUIR

Y.I. Alekseey
Assistant of the Department of Informatics, Master of Technical Sciences.

T.D. Pozhyakov
Student of BSUIR

Abstract. In this article, we consider one of the Big Data storage technologies. Hadoop.Hadoop offers one of the most economical and efficient ways to store huge amounts of data. Structured, semi-structured, and unstructured data types can be stored and then processed using tools such as Pig, Ulive, and Spark. Various storage formats are also described and compared Big Data File Formats

Keywords: Hadoop, Big Data, Spark.