# Система обработки больших данных для анализа событий репозитория GitHub

Н. В. Воинов[1], К. Родригес Гарсон, И. В. Никифоров, П. Д. Дробинцев
Санкт-Петербургский политехнический университет Петра Великого
[1]voinov@ics2.ecd.spbstu.ru

***Аннотация*. В статье рассмотрена архитектура системы обработки больших данных, основанной на инструментах Apache Hadoop, Apache Flume и Apache Spark. Продемонстрировано применение разработанной системы для хранения и анализа наборов данных, состоящих из генерируемых событий в репозитории GitHub - крупнейшего в мире веб-сервиса на базе системы контроля версий Git. Выбраны метрики, с помощью которых проведена оценка полученных результатов работы системы.**

***Ключевые слова: большие данные; распределенная обработка данных; Apache Hadoop; MapReduce; репозиторий GitHub***

## I. Введение

На сегодняшний день хранение больших объемов данных более не является привилегией крупных компаний и государственных органов. Любой может хранить информацию в таких объемах, которые сложно было представить еще несколько лет назад, например, последовательности ДНК, история болезней тысяч пациентов, данные о перемещениях автомобиля, транспортная ситуация в крупных городах и т.д. [1-3]

Объемы данных растут с огромной скоростью, что ставит новый вызов – извлекать информацию и знания в результате анализа получаемых данных. В этом могут помочь такие крупные компании как Google, Amazon, Apache Software Foundation, которые создают специальные инструменты для решения поставленной задачи.

В данной статье описана разработанная система, основанная на инструментах Apache Hadoop [4], Apache Flume [5] и Apache Spark [6], анализирующая события репозитория GitHab в качестве тестового набора данных через открытый GitArchive API [7].

## II. Применяемые Технологии и Инструменты

### A. GitHub

Системы версионного контроля – одни из наиболее часто используемых разработчиками программного обеспечения инструментов для сохранения изменений в файле или наборе файлов.

GitHub [8] на сегодня является самой популярной системой версионного контроля, поскольку обладает рядом существенных преимуществ: распределенность, поддержка более 200 языков программирования и форматов данных, функции опубликования и хранения, защита данных с помощью SSL, SSH, https, а также двухфакторная аутентификация при авторизации. GitHub предоставляет доступ к открытому набору данных, который называется GitArchive, содержащему более 20 типов событий, включая commit (“фиксирование”), fork (“ветвление”), создание новых задач, комментарии, добавление участников проекта и другие. Каждый час создается новый архив с событиями. Каждое событие имеет структуру типа json, которая подробно задокументирована, и свой API, позволяющий загрузить его по протоколу HTTP REST.

Для тестирования разработанной системы использовался архив событий репозитория GitHub размером 86.7 Гб.

### B. OLTP и OLAP

Динамичное развитие деловой среды рождает запрос на информационные системы, предоставляющие решения на требования рынка. По своему поведению и способу представления информации существуют два типа систем анализа больших данных.

- Online Transaction Processing Systems (OLTP): системы для обработки большого количества запросов конечных пользователей настолько быстро, насколько возможно, при этом также обеспечивая общую целостность всей системы [9].
- Online Analytical Processing Systems (OLAP): системы для обработки более сложных запросов над огромными объемами данных с целью извлечения знаний и отчетности нежели исполнения запросов конечных пользователей [9].

Основываясь на задачах проекта, для дальнейшей работы была выбрана OLAP-подобная база данных Apache Hadoop, что объясняется следующими причинами:

- Hadoop – это нереляционная база данных, хранящая и обрабатывающая любые типы данных, не только таблицы.
- Hadoop соответствует трем важнейшим характеристикам больших данных: объем, скорость, разнообразие.

- Тестовый набор данных состоит из сохраненных ранее событий, а не событий реального времени.

## III. Архитектура Системы и Реализация

### A. *Общая схема*

Архитектура разработанной системы представлена на рис. 1.

Она состоит из четырех основных компонентов: файлы в формате json, Apache Flume, Apache Hadoop и веб-приложения для отображения результатов анализа.

### B. *Особенности реализации*

Первый этап работы системы – конфигурация базы данных. После этого Flume необходимо соединить с Hadoop. Flume предоставляет три компонента: “источник”, “канал” и “сток”. Источник определяет расположение и формат файлов json, канал хранится в памяти, сток указывает на расположение HFDS. Последний шаг – соединение Spark с Hadoop для выполнения анализа данных. Apache Spark предоставляет библиотеку RySpark для работы с языком Python. Данная библиотека взаимодействует с YARM для исполнения запросов.

Первый компонент на рис. 1 обозначает набор файлов json, каждый из которых содержит события, зарегистрированные в GitHub в определенное время дня. Благодаря использованию Apache Flume возможно последовательное перемещение файлов внутри HDFS. Flume берет каждый файл из источника, загружает его в канал и делает его копию в HDFS через сток.

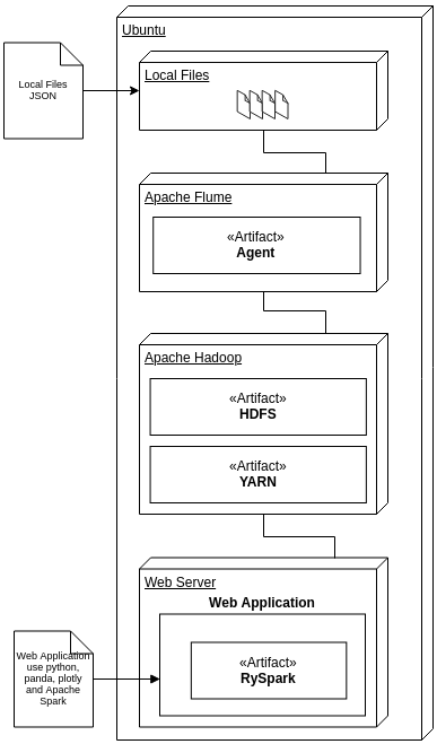


Рис. 1. Архитектура системы

Каждое событие имеет независимую структуру, т.е. одно событие может иметь n параметров, а другое – m параметров. Структура не гомогенная, что представляет основную проблему при анализе данных. Это решается использованием Spark SQL. Spark считывает файлы в HDFS и загружает их в RDD (Resilient Distributed Datasets), затем выполняет анализ возможных типов данных и их структуры. Особенность RDD – поддержка вычислений в памяти крупных распределенных кластеров. После решения конфликтов между типами данных, создается структура данных, которая интерпретируется как классическая таблица в реляционной базе данных. Данная схема является базисом для выполнения запросов SQL-типа.

После того, как все компоненты сконфигурированы и объединены между собой, осуществляется доступ к архивам GitArchive. Учитывая структуру данных и типы событий, хранимых в GitHub, были сформулированы 7 запросов, каждый из которых был направлен на тестовый набор данных.

*1) Классификация и подсчет событий, их группировка по месяцам:* Какое событие всречается чаще всего? Какое событие встречается реже всего?

*2) Поиск всех событий в определенное время суток:* В какое время суток происходит больше всего событий? Меньше всего событий? Какое среднее число событий в день?

*3) Поиск всех событий, которые влияют на увеличение и уменьшение размера репозитория, их группировка по месяцам:* На сколько увеличился или уменьшился размер репозитория за месяц?

*4) Поиск событий “ветвления”, их группировка по репозиториям и месяцам:* Какие 5 самых популярных репозиториев среди разработчиков программного обеспечения?

*5) Поиск событий создания тэгов и исправления ошибок:* Какое количество событий по созданию тэгов и исправлению ошибок?

*6) Сортировка по количеству событий в репозиториях по месяцам:* Самые активные репозитории за месяц? Самые неактивные репозитории за месяц?

*7) Поиск событий добавления кода и документирования кода:* Какое количество событий по добавлению кода и документированию кода?

## IV. Результаты

Для оценки полученных результатов применялись две метрики. Первая оценивала правдивость и согласованность результатов запросов, вторая – время исполнения запросов.

В качестве примера рассмотрим результаты оценки первых двух запросов.

На рис. 2 показаны все 31 925 108 событий, сгруппированных по месяцам. Наиболее частым является событие “Push”, самым редким – “Release”.

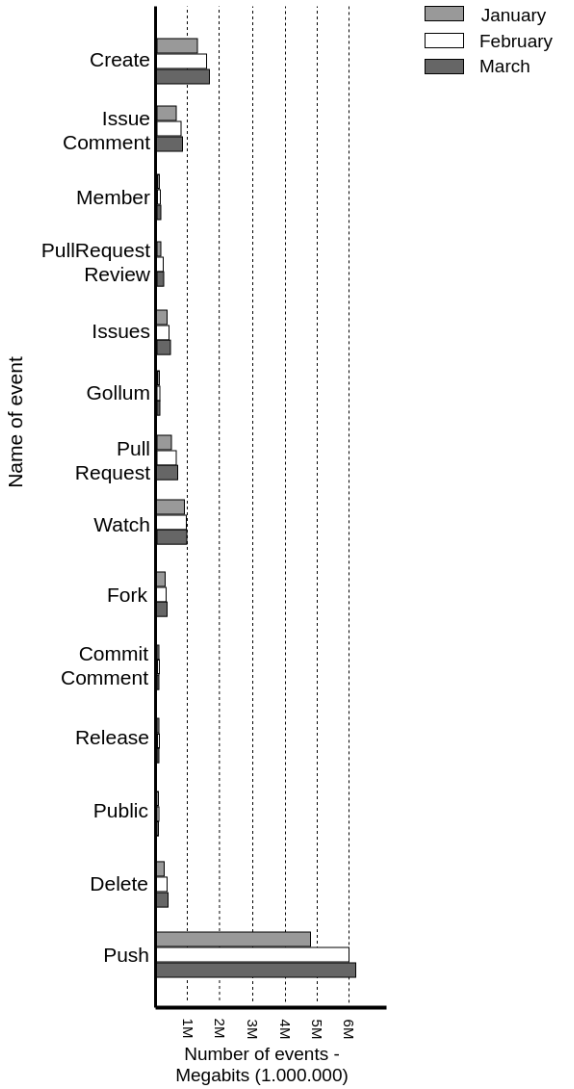


Рис. 2. Результаты запроса 1)

На рис. 3 показано общее количество событий каждый час в течение дня. По оси X расположены часы, по оси Y – количество событий. Больше всего событий фиксируется в 6 часов вечера, меньше всего – в 7 часов утра. Самое напряженное в плане количества событий время суток – ранний вечер. Подобные графики могут стать весьма полезными для людей, ответственных за мониторинг активности сотрудников, и послужить основой для выработки различных решений по распорядку дня и рабочего графика.

В табл. 1 представлено время исполнения каждого запроса. Среднее время исполнения 50.22 минуты. Таким образом, каждую минуту обрабатывается порядка 1.3 Гб данных. Время исполнения запроса 7) значительно меньше по сравнению с другими запросами, поскольку в нем не используется функция GROUP BY, а считываются результаты предыдущих запросов.

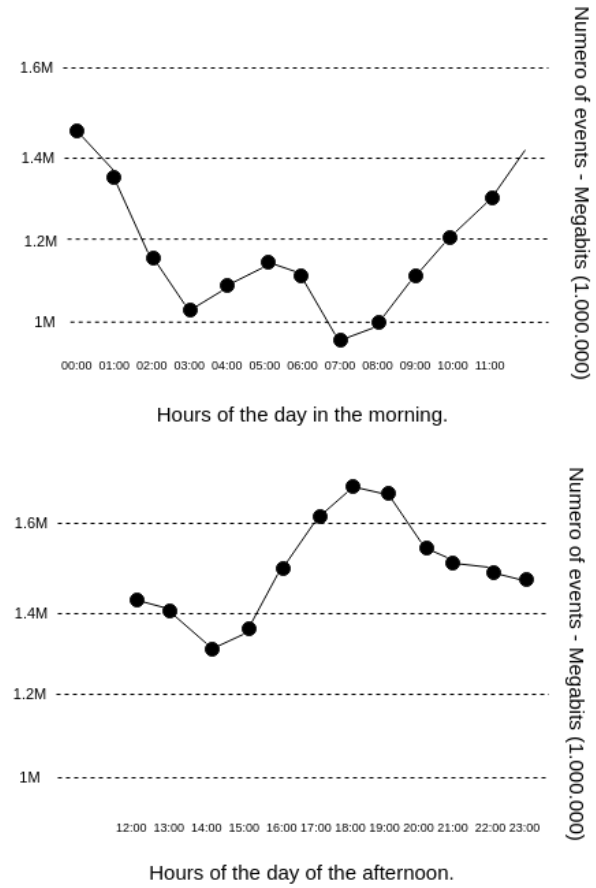


Рис. 3. Результаты запроса 2)

ТАБЛИЦА I ВРЕМЯ ИСПОЛНЕНИЯ КАЖДОГО ЗАПРОСА

| Номер Запроса | Время Исполнения, мин |
|---|---|
| 1 | 51.8 |
| 2 | 56.9 |
| 3 | 55.3 |
| 4 | 55.2 |
| 5 | 52.6 |
| 6 | 55.3 |
| 7 | 24.5 |

Все тесты выполнялись на кластере с одним вычислительным узлом. Характеристики вычислительных ресурсов приведены в табл. 2.

ТАБЛИЦА II ХАРАКТЕРИСТИКИ РЕСУРСОВ

| Ресурс | Описание |
|---|---|
| Операционная система | Ubuntu 18.04.1 LTS |
| Память | 11 Гб |
| Процессор | Intel Core i7 - 7500U до 3.58Гц |
| Количество процессоров | 4 |
| Жесткий диск | 100 Гб HDD |
| Скорость передачи данных | Скачивание 83.15 Mbps - Загрузка 83.23 Mbps |

## V. Выводы

Тестирование разработанной системы показало удовлетворительные результаты. Следующие особенности архитектуры Hadoop доказывают ее преимущества при создании решений по анализу больших объемов данных:

- Hadoop работает как со структурированными, так и полуструктурированными данными, что делает его достаточно гибким инструментом. С точки зрения разработчика, структура исходных данных не важна.
- Хотя архитектура сложна, ее применение и конфигурация достаточно просты, имеется понятная и подробная документация. Важно отметить возможность масштабирования путем добавления новых узлов и распределенных вычислений благодаря MapReduce.
- Впечатляющая скорость вычислений, учитывая достаточно ограниченные ресурсы.

## Список литературы


[1] Laboshin L.U., Lukashin A.A., Zaborovsky V.S. The Big Data approach to collecting and analyzing traffic data in large scale networks. Procedia Computer Science, vol. 103, pp. 536-542, 2017. DOI: 10.1016/j.procs.2017.01.048

[2] Bataev A.V. Evaluation of Using Big Data Technologies in Russian Financial Institutions. Proceedings of the 2018 IEEE International Quality Management, Transport and Information Security, Information Technologies ITQMIS2018, art. no. 8524938, pp. 573-577, 2018. DOI: 10.1109/ITMQIS.2018.8524938

[3] Bataev A.V. Analysis of the Application of Big Data Technologies in the Financial Sphere. Proceedings of the 2018 IEEE International Quality Management, Transport and Information Security, Information Technologies ITQMIS2018, art. no. 8525121, pp. 568-572, 2018. DOI: 10.1109/ITMQIS.2018.8525121

[4] *Apache Hadoop*. Available at: https://hadoop.apache.org (accessed 3 April 2019).

[5] *Apache Flume*. Available at: https://flume.apache.org (accessed 3 April 2019).

[6] *Apache Spark*. Available at: https://spark.apache.org (accessed 3 April 2019).

[7] *GitAtchive*. Available at: https://www.gharchive.org (accessed 3 April 2019).

[8] *GitHub*. Available at: https://github.com (accessed 3 April 2019).

[9] *OLTP vs. OLAP: The Era of Specialization*. Available at: https://www.edx.org/es/course/introductionapache-hadoop-linuxfoundationx-lfs103x (accessed 3 April 2019).