МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА — Российский технологический университет»

РТУ МИРЭА

Институт кибербезопасности и цифровых технологий
(наименование института, филиала)

Кафедра КБ-14 «Цифровые технологии обработки данных»
(наименование кафедры)

КУРСОВОЙ ПРОЕКТ (РАБОТА)

по дисциплине Организация обработки больших данных
(наименование дисциплины)

Тема курсовой работы Организация обработки больших данных средствами Apache Spark

Студент группы БСБО-05-20 В. С. Верхотуров _____
учебная группа, фамилия, имя, отчество студента — подпись студента

Руководитель курсовой работы _____ _____
подпись руководителя

Член комиссии _____ _____
подпись члена комиссии

Работа предоставлена к защите « » 2023 г.

Допущен к защите « » 2023 г.

Москва — 2023 г.

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА — Российский технологический университет»

Институт ИКЦТ направление 09.03.02 «Информационные системы и технологии»

Кафедра КБ-14 «Цифровые технологии обработки данных»

Дисциплина «Организация обработки больших данных»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе (проекту)

Студент В. С. Верхотуров

Группа БСБО-05-20

Работа защищена на оценку _____

Руководитель работы _____

Член комиссии _____

Москва — 2023 г.

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА — Российский технологический университет»

Институт ИКЦТ направление 09.03.02 «Информационные системы и технологии»

Кафедра КБ-14 «Цифровые технологии обработки данных»

Дисциплина «Организация обработки больших данных»

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студент 4 курса группы БСБО-05-20.

1) Тема: Проектирование баз данных
2) Срок представления проекта (работы) к защите:
3) Содержание пояснительной записки:
    - Содержание
    - Краткие теоретические сведения
    - Задание на курсовую работу
    - Анализ социальной акивности средствами Apache Spark
    - Заключение
    - Литература

Руководитель работы А. С. Лебедев

Задание принял к исполнению В. С. Верхотуров

Москва — 2023 г.

СОДЕРЖАНИЕ

СОДЕРЖАНИЕ 4

1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ 5

2 ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ 7

3 АНАЛИЗ СОЦИАЛЬНОЙ АКТИВНОСТИ СРЕДСТВАМИ APACHE SPARK 8

3.1 Анализ данных средствами RDD и SparkSQL 8

3.2 Анализ данных средствами GraphFrames 10

ЗАКЛЮЧЕНИЕ 13

ЛИТЕРАТУРА 14

1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Resilent Distributed DataSet [1, 2, 3] — это неизменяемая распределенная коллекция наборов данных, разделенных по множеству узлов кластера, которую можно восстановить в случае потери раздела, что обеспечивает отказоустойчивость. Он также обеспечивает встроенные вычисления в памяти и ссылается на наборы данных, хранящиеся во внешних системах хранения.

Spark SQL [4, 5, 6] — это модуль Spark для обработки структурированных данных. Он предоставляет абстракцию программирования, называемую DataFrames, а также может действовать как механизм распределенных запросов SQL. Это позволяет немодифицированным запросам Hadoop Hive выполняться до 100 раз быстрее для существующих развертываний и данных.

GraphFrames [7, 8, 9, 10] — это пакет для Apache Spark, который предоставляет графики на основе DataFrame. Он предоставляет API-интерфейсы высокого уровня на Scala, Java и Python. Он призван обеспечить как функциональность GraphX, так и расширенную функциональность с использованием Spark DataFrames. Эта расширенная функциональность включает в себя поиск мотивов, сериализацию на основе DataFrame и высоковыразительные графовые запросы.

2 ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Часть 1. RDD, SparkSQL

Загрузить набор данных из twitter и найти пользователя из РФ, чаще остальных упоминающего фамилии российских политических деятелей (на русском).

Решить задачу необходимо средствами RDD transformations + actions и DataFrame + SparkSQL. Продемонстрировать, что ответ совпадает.

Часть 2. GraphFrames

Загрузить набор данных из twitter и найти наибольшую компоненту связности социального графа (группу пользователей, которые общаются преимущественно друг с другом) для российских пользователей.

Решить задачу необходимо средствами GraphFrames.

twitterid — идентификационный номер твита

userid — идентификационный номер пользователя (анонимизированный для пользователей, у которых на момент приостановки было менее 5000 подписчиков)

user_display_name — имя пользователя (закодированное как userid для анонимных пользователей)

user_screen_name — дескриптор пользователя в Твиттере (закодированный как идентификатор пользователя для анонимных пользователей)

user_reported_location — местоположение пользователя, сообщенное самим пользователем

user_profile_description — описание профиля пользователя

user_profile_url — URL профиля пользователя

follower_count — количество аккаунтов, подписавшихся на пользователя

follow_count — количество аккаунтов, на которые подписан пользователь

account_creation_date — дата создания учетной записи пользователя

account_language — язык аккаунта, выбранный пользователем.

twitter_language — язык твита

twitter_text — текст твита (упоминания анонимизированных аккаунтов

заменены анонимизированными идентификаторами пользователей)

twitter_time — время публикации твита (UTC)

twitter_client_name — имя клиентского приложения, используемого для публикации твита.

in_reply_to_tweetid — идентификатор исходного твита, на который этот твит является ответом (только для ответов)

in_reply_to_userid — идентификатор пользователя исходного твита, на который этот твит является ответом (только для ответов)

quoted_tweet_tweetid — идентификатор исходного твита, который цитируется в этом твите (только для кавычек)

is_retweet — True/False, является ли этот твит ретвитом

retweet_userid — для ретвитов идентификатор пользователя, автора исходного твита.

retweet_tweetid — для ретвитов твитид исходного твита.

latitude - географическая широта, если доступна.

longitude - долгота с географической привязкой, если доступна

quote_count — количество твитов, цитирующих этот твит

reply_count - количество твитов, ответивших на этот твит

like_count — количество лайков, полученных этим твитом

retweet_count — количество ретвитов, полученных этим твитом

hashtags — список хэштегов, использованных в этом твите.

URL-адреса — список URL-адресов, использованных в этом твите.

user_mentions — список идентификаторов пользователей, упомянутых в этом твите (включая анонимные идентификаторы пользователей)

poll_choices — если твит содержал опрос, в этом поле отображаются варианты опроса, разделенные знаком «|».

3 АНАЛИЗ СОЦИАЛЬНОЙ АКТИВНОСТИ СРЕДСТВАМИ APACHE SPARK

3.1 Анализ данных средствами RDD и SparkSQL

Найти наибольшую компоненту связности социального графа (группу пользователей, которые общаются преимущественно друг с другом) для российских пользователей.

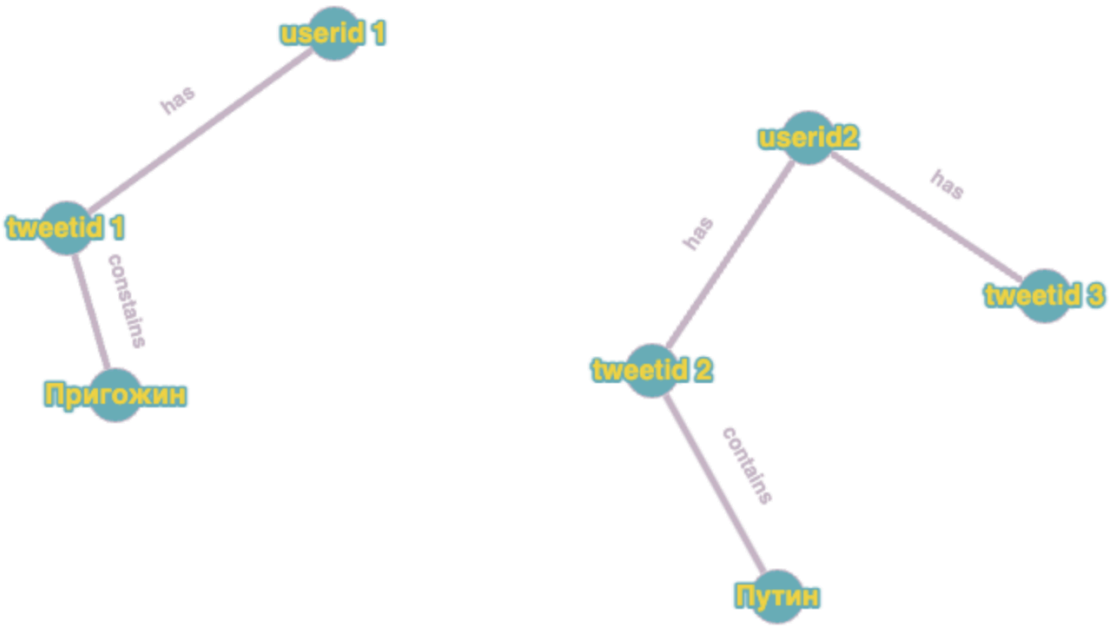

Рисунок 1 – Визуализация датасета

Загрузка датасета в SparkRDD:

Листинг 1 – Нахождение наибольшей компоненты связанности

```
1  def set_tweets(self, path):
2          sc = SparkContext().getOrCreate()
3          self.tweets = sc.textFile(path)\
4                  .map(lambda line: line[1:-1].split('","'))
5          self.tweets = self.tweets.map(
6                  lambda element:
7                          (element[1],
8                           element[10],
9                           element[12])
10         )
```

Выбор сообщений на русском языке:

Листинг 2 – Выбор сообщений

```
@staticmethod
def filter_func(row):
        if row[1] == 'ru':
                for politician in politicians:
                        if politician in row[2]:
                                return True
        return False

def filter_tweets(self):
        self.tweets = self.tweets.filter(
                SparkRDD.filter_func
        )
        self.tweets = self.tweets.map(
                lambda element:
                        (element[0], str(element[2]))
        )
```

Получение userid, который чаще остальных упоминающего фамилии российских политических деятелей (на русском):

Листинг 3 – Получение userid

```
def get_userid(self):
        self.users_tweets = self.tweets.groupByKey()\
                .mapValues(len)
        self.sorted_tweets = self.users_tweets.sortBy(
                lambda element: element[1], ascending=False
        )
        return self.sorted_tweets.max(
                lambda element: element[1]
        )[0]
```

Указание пути и запуск SparkRDD:

Листинг 4 – Получение userid

```
if __name__ == '__main__':
file_path = 'hdfs:///ira_tweets_csv_hashed.csv'
sparkRDD = SparkRDD()
sparkRDD.set_tweets(file_path)
sparkRDD.filter_tweets()
print("USER_ID: " + sparkRDD.get_userid())
```

Загрузка датасета:

Листинг 5 – Получение userid

```
self.tweets = self.spark.read.csv(path, header=True)
self.tweets.createOrReplaceTempView("tweets")
```

Создание вью, которое отдает русские сообщения:

Листинг 6 – Получение вью с русскими сообщениями

```
def tweets_temp(self):
        query = """\
                CREATE OR REPLACE TEMP VIEW tweets_temp AS
                SELECT tweetid,
                        userid,
                        tweet_text,
                        account_language
                FROM tweets
                WHERE account_language == 'ru'
                        and tweet_language == 'ru'
        """
        self.spark.sql(query)
```

Получение userid, который чаще остальных упоминающего фамилии российских политических деятелей (на русском):

Листинг 7 – Получение userid

```
def get_userid(self):
        query = f"SELECT userid FROM tweets_temp " +\
                "WHERE tweet_text " +\
                "LIKE '%{self.politicians[0]}%'"
        for politician in self.politicians[1:]:
                query += f" OR tweet_text '" +\
                f"LIKE '%{politician}%'"
                query += f" GROUP BY userid " +\
                        "ORDER BY COUNT(tweetid) DESC LIMIT 1"
                return self.spark.sql(query).show()
```

Указание пути и запуск SparkSQL:

Листинг 8 – Запуск SparkSQL

```
if __name__ == '__main__':
        file_path = 'hdfs:///ira_tweets_csv_hashed.csv'
        sparkSQL = SparkSQL()
        sparkSQL.set_tweets(file_path)
        sparkSQL.tweets_temp()
        print(sparkSQL.get_userid())
```

3.2 Анализ данных средствами GraphFrames

Найти наибольшую компоненту связности социального графа (группу пользователей, которые общаются преимущественно друг с другом) для рос-

сийских пользователей.

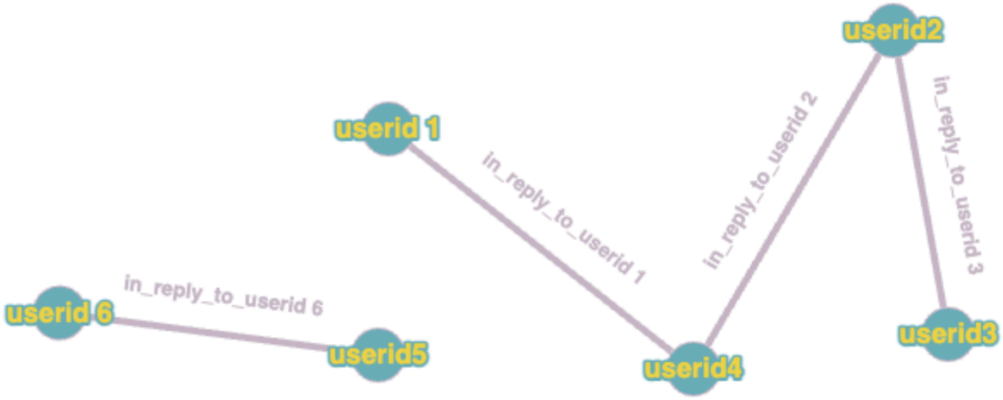

Рисунок 2 – Визуализация создаваемого графа

Создание графа с пользователями в вершине:

Листинг 9 – Создание графа

```
def set_graphframe(self, path):
        self.tweets = self.spark\
                .read.csv(path, header=True)
        self.tweets.createGlobalTempView("users")
        self.filtered_users = self.spark.sql(
                "SELECT * FROM global_temp.users " +\
                "WHERE user_reported_location " +\
                "LIKE '%" +\
                        "Россия" +\
                        "%'"
        )
        vertices = self.filtered_users\
                .select('userid').toDF('id')
        edges = self.filtered_users\
                .select('userid', 'in_reply_to_userid')\
                .toDF('src', 'dst')
        edges = edges.filter(edges.dst != 'null')
        self.gf = GraphFrame(vertices, edges)
```

Получение связанных компонент:

Листинг 10 – Получение связанных компонент

```
def get_components(self):
        components = self.gf.connectedComponents()
        components.createOrReplaceTempView("components")
```

Получение наибольшей компоненты связности:

Листинг 11 – Получение наибольшей компоненты связанности

```
def get_max_component(self):
        self.get_components()
        query = """
                SELECT component, COUNT(*)
                AS count
                FROM components
                GROUP BY component
                ORDER BY count
                DESC
        """
        return self.spark.sql(query).show()
```

ЗАКЛЮЧЕНИЕ

В ходе выполнения первой задачи была допущена погрешность — для поиска политиков использовался заранее заданный список фамилий, в который могли войти не все политики России, упоминающиеся в датасете.

При выполнении второй практической иногда невозможно было определить местоположение пользователя, т.к. не было заполнено поле user_reported_location.

Литература

1. Zaharia, Matei, et al. ”Resilient distributed datasets: A Fault-Tolerant abstraction for In-Memory cluster computing.”9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12). 2012.

2. Zaharia, Matei, et al. ”Resilient distributed datasets.”A fault-tolerant abstraction for in-memory cluster computing in Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. 2014.

3. Deng, Changshou, et al. ”A parallel version of differential evolution based on resilient distributed datasets model.”Bio-Inspired Computing–Theories and Applications: 10th International Conference, BIC-TA 2015 Hefei, China, September 25-28, 2015, Proceedings 10. Springer Berlin Heidelberg, 2015.

4. Armbrust, Michael, et al. ”Spark sql: Relational data processing in spark.”Proceedings of the 2015 ACM SIGMOD international conference on management of data. 2015.

5. Baldacci, Lorenzo, and Matteo Golfarelli. ”A cost model for SPARK SQL.”IEEE Transactions on Knowledge and Data Engineering 31.5 (2018): 819-832.

6. Ivanov, Todor, and Max-Georg Beer. ”Evaluating hive and spark SQL with BigBench.”arXiv preprint arXiv:1512.08417 (2015).

7. Dave, Ankur, et al. ”Graphframes: an integrated api for mixing graph and relational queries.”Proceedings of the fourth international workshop on graph data management experiences and systems. 2016.

8. Mishra, Raju Kumar, et al. ”GraphFrames.”PySpark SQL Recipes: With HiveQL, Dataframe and Graphframes (2019): 297-315.

9. Bahrami, Ramazan Ali, Jayati Gulati, and Muhammad Abulaish. ”Efficient processing of SPARQL queries over graphframes.”Proceedings of the International Conference on Web Intelligence. 2017.

10. Popov, Alexander, and Jennifer Sikos. ”Graph embeddings for frame identification.”Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019). 2019.