Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Отчёт по лабораторной работе №4

Тема: «Работа с пакетом SPARK»

Выполнили:

студенты группы 150504 Горбачевский М. В, Горбачевский К. В.

Проверила:

к.т.н., доцент Герман Ю.О.

Минск 2023

1. Цель работы

Изучить технику обработки текста в SPARK Scala.

1. Краткие теоретические сведения

Команда для чтения текстового файла и ее работа приведены на рисунке 2.1. Путь к файлу прописан полный, строки разделяются знаком «\\».

val lines = spark.read.textFile("C:\\BSUIR\\3course\\fprogr\\laba4\\sport.txt")

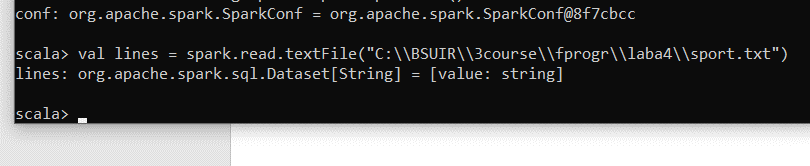


Рисунок 2.1 – Чтение текстового файла

Далее на рисунке 2.2 приведена команда получения массива всех строк и результат ее работы.

val words = lines.map(line => line.split(" "))

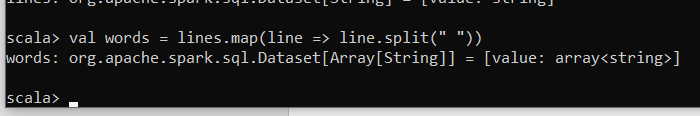


Рисунок 2.2 – Получение массива строк

Ниже приведены строки, которые реализуют вывод первого предложения, содержащего не менее 3 слов. Результат работы приведен на рисунке 2.3.

val wordCounts = words.map(wordArray => wordArray.length)

val maxWordCount = wordCounts.max()

val longestSentence = lines.filter(line => line.split(“ ”).length >= 3).first()

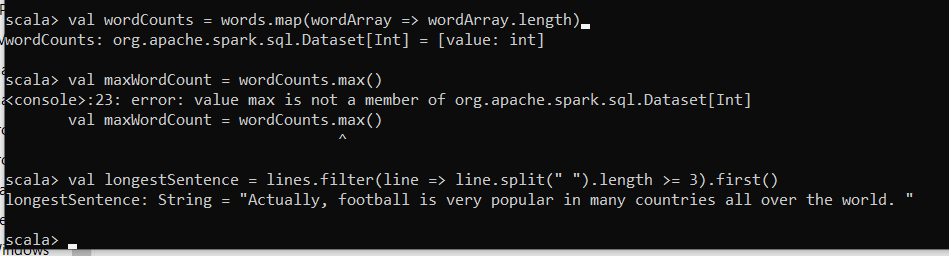


Рисунок 2.3 – Результат работы

Вывод строк из текстового файла(рис. 2.4):

val longestSentence2 = lines.foreach(line => println(line))

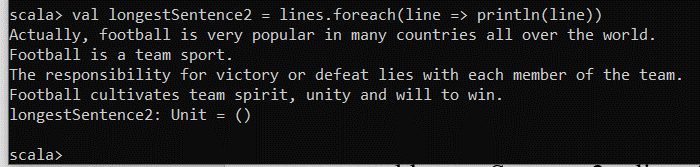


Рисунок 2.4 – Вывод строк из файла

Далее продолжаем работу с этим же текстом. Расщепляем текст на отдельные слова. При этом используется регулярное выражение W+ (означает любую последовательность символов).

val wordsRDD = lines.flatMap(line => line.split(“\\W+”))

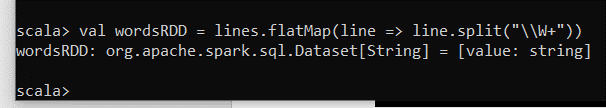


Рисунок 2.5 – Расщепление текста на слова

Приведение всех слов к нижнему регистру (рис. 2.6):

val lowercaseWordsRDD = wordsRDD.map(word => word.toLowerCase())

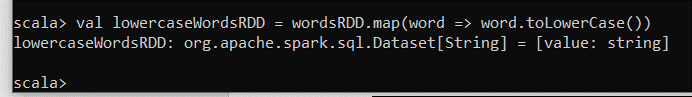


Рисунок 2.6 – Приведение слов к нижнему регистру

Создание множества стоп-слов (рис. 2.7):

val stopWords = Set(“is”, “the”, “of”, “and”)

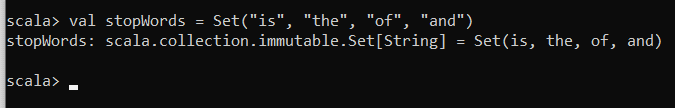


Рисунок 2.7 – Создание множества стоп-слов

Далее удаляем стоп-слова из списка и выводим отфильтрованное множество слов. Результат работы приведен на рисунке 2.8.

val filteredWordsRDD = lowercaseWordsRDD.filter(word => !stopWords.contains(word))

filteredWordsRDD.collect().foreach(println)

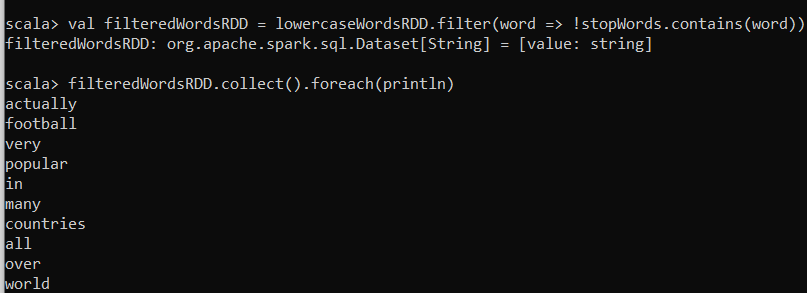


Рисунок 2.8 – Вывод слов после удаления

Далее отфильтруем слова, которые начинаются на “w” и выведем их. Результат работы приведен на рисунке 2.9.

val filteredWordsRDD2 = wordsRDD.filter(word => word.startsWith(“w”))

filteredWordsRDD2.collect().foreach(println)

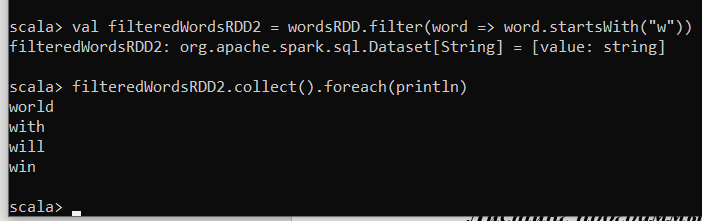


Рисунок 2.9 – Результат работы

Пример работы с паттерном. Первая строчка создает паттерн. Этот паттерн определяет все слова, начинающиеся на w, содержащие где-то посередине r и заканчивающиеся на d. Сформируем отфильтрованное множество по паттерну и выведем результат на экран. Результат работы приведен на рисунке 2.10.

val pattern = “^w.\*r\*d$”

val filteredWordsRDD3 = filteredWordsRDD.filter(word => word.matches(pattern))

filteredWordsRDD3.collect().foreach(println)

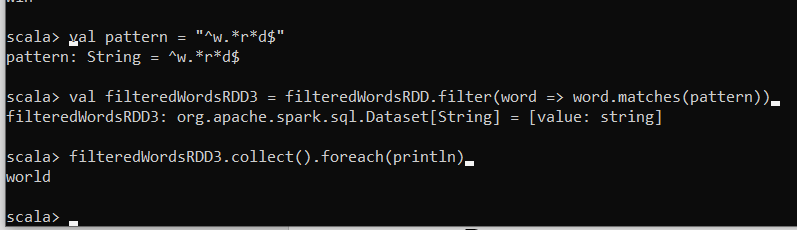


Рисунок 2.10 – Работа с паттерном

1. Ход работы

***Задания к варианту 1:***

Создать собственный текстовый файл на английском или немецком языке – 4-5 предложений.

***Текст для выполнения заданий:***

It is a long established fact that a reader will be distracted by the readable content

of a page when looking at its layout. The point of using Lorem Ipsum is that it has a

more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English.

1. Вывести все слова из текстового файла, исключая stop-слова

val lines = spark.read.textFile("D:\\unik\\sem5\\fprog\\lab4\\var1.txt")

val words = lines.flatMap(line => line.split("\\W+"))

val stopWords = Set("a", "an", "the", "and", "but", "or", "for", "of")

val notStopWords = words.filter(w => !stopWords.contains(w))

notStopWords.collect().foreach(println)

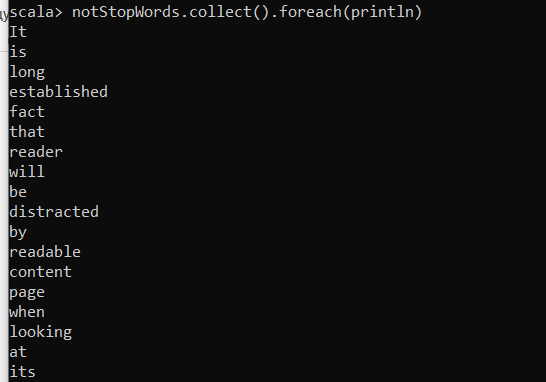


Рисунок 3.1 – Результат работы задания 1

1. Вывести все слова, содержащие букву t

val lines = spark.read.textFile("D:\\unik\\sem5\\fprog\\lab4\\var1.txt")

val words = lines.flatMap(line => line.split("\\W+"))

val pattern = "^.\*t.\*$"

val containsT = words.filter(w => w.matches(pattern))

containsT.collect().foreach(println)

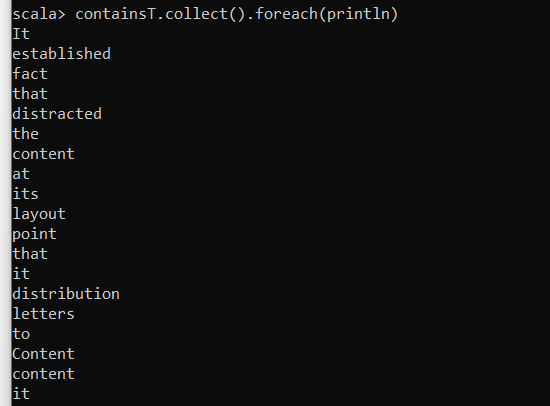


Рисунок 3.2 – Результат работы задания 2

3. Вывести все слова, заканчивающиеся на ing.

val lines = spark.read.textFile("D:\\unik\\sem5\\fprog\\lab4\\var1.txt")

val words = lines.flatMap(line => line.split("\\W+"))

val pattern = "^.\*ing$"

val endsWithIng = words.filter(w => w.matches(pattern))

endsWithIng.collect().foreach(println)

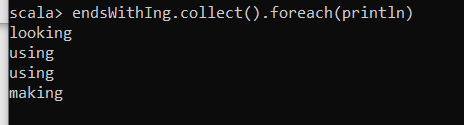


Рисунок 3.3 – Результат работы задания 3

4. Вывести все слова вторая буква которых a.

val lines = spark.read.textFile("D:\\unik\\sem5\\fprog\\lab4\\var1.txt")

val words = lines.flatMap(line => line.split("\\W+"))

val pattern = "^.a.\*$"

val secondIsA = words.filter(w => w.matches(pattern))

secondIsA.collect().foreach(println)

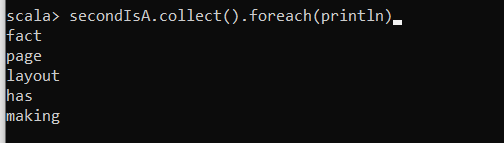


Рисунок 3.4 – Результат работы задания 4

5. Вывести все слова, последняя буква которых s

val lines = spark.read.textFile("D:\\unik\\sem5\\fprog\\lab4\\var1.txt")

val words = lines.flatMap(line => line.split("\\W+"))

val pattern = "^.\*s$"

val lastIsS = words.filter(w => w.matches(pattern))

lastIsS.collect().foreach(println)

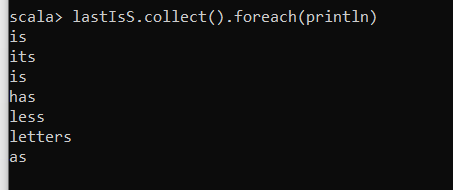


Рисунок 3.5 – Результат работы задания 5

6. Вывести каждое второе слово

val values = wordsRDD.collect()

val secondToLastWord = if (values.length >= 2) values(values.length - 2) else ""

println(s"Second to last word: $secondToLastWord")

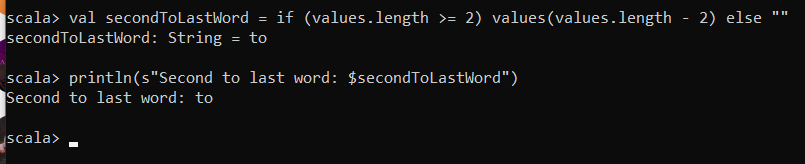


Рисунок 3.6 – Результат работы задания 6

1. **Вывод**

В ходе выполнения данной лабораторной работы нами были проанализированы и изучены техники работы с пакетом SPARK в Scala.