МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский ядерный университет «МИФИ»

**Обнинский институт атомной энергетики –**

филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ»

**(ИАТЭ НИЯУ МИФИ)**

Отделение интеллектуальных кибернетических систем

ЛАБОРАТОРНАЯ РАБОТА №2

«Изучение возможностей Scala»

по дисциплине

«Анализ литературной работы»

Выполнил студент 1 курса

группы ИВТ-М20

Лискунов Р. Г.

Проверил:

кандидат технических наук

Грицюк С. В.

Обнинск, 2020

# Цель работы

Разработать приложение для Apache Spark в автономном режиме. Проанализировать литературное произведение (Федор Михайлович Достоевский – Преступление и наказание) с помощью Spark RDD.

# Краткая теория

Apache Spark — фреймворк с открытым исходным кодом для реализации распределённой обработки неструктурированных и слабоструктурированных данных, входящий в экосистему проектов Hadoop. В отличие от классического обработчика из ядра Hadoop, реализующего двухуровневую концепцию MapReduce с хранением промежуточных данных на накопителях, Spark работает в парадигме резидентных вычислений — обрабатывает данные в оперативной памяти, благодаря чему позволяет получать значительный выигрыш в скорости работы для некоторых классов задач, в частности, возможность многократного доступа к загруженным в память пользовательским данным делает библиотеку привлекательной для алгоритмов машинного обучения.

## Структуры данных RDD

RDD – это распределенная коллекция данных, расположенных по нескольким узлам кластера, набор объектов Java или Scala, представляющих данные. RDD работает со структурированными и с неструктурированные данными. Также, как DataFrame и DataSet, RDD не выводит схему загруженных данных и требует от пользователя ее указания.

RDD-коллекция сериализуется каждый раз, когда Spark требуется распределить данные внутри кластера или записать информацию на диск. Затраты на сериализацию отдельных объектов Java и Scala являются дорогостоящими, т.к. выполняется отправка данных и структур между узлами.

# Ход работы

В начале работы происходит инициализация конфигурации состояния Spark, что позволяет в дальнейшем подключить контекст работы для прочтения файла с литературным произведением и дальнейшим взаимодействием с ним.

Для анализа я подготавливаю исходный текст с помощью устранения пунктуационных знаков: точка, запятая и так далее. В дополнение, слова данного произведения переводятся в нижний регистр для увеличения числа показателей значений.

В процессе обработки текста участвует специальный список стоп-слов, хранящийся в файле stop.txt. При сравнении исходного текста произведения со словами, которые представлены в этом файле, происходит устранение союзов, предлогов, глаголов и других лишних слов.

Затем происходит поиск 50 самых популярных и 50 самых редко используемых слов. Так, исследование показывает, что имя фамилия главного героя – Раскольников встречается 563 раз, что показывает нам близость автора и основного героя романа.

В процессе лабораторной работы был произведён поиск для 50 самых популярных и редко встречающихся слов, приведу пример выполняемого кода:

**Top50 most common words:**

(раскольников,564)

(соня,264)

(разумихин,244)

(петрович,208)

(ивановна,174)

(свидригайлов,141)

(дуня,136)

(порфирий,131)

(раскольникова,127)

(катерина,126)

(петр,120)

(пульхерия,109)

(александровна,109)

(глаза,108)

(деньги,103)

(родя,102)

(брат,91)

(родион,90)

(начал,86)

(сердце,80)

(господи,79)

(романовна,77)

(авдотья,76)

(давеча,74)

(дунечка,73)

(ужасно,73)

(мать,72)

(право,71)

(неужели,70)

(вопрос,68)

(дома,68)

(раскольникову,64)

(господин,63)

(черт,63)

(например,63)

(крайней,62)

(комнате,62)

(вскричал,62)

(какой-то,61)

(настасья,61)

(деле,60)

(комнату,59)

(ко,59)

(стоит,59)

(зосимов,58)

(разумеется,58)

(напротив,57)

(вашей,57)

(убил,57)

(которую,57)

**Top50 least common words:**

(сменялись,1)

(отворяль,1)

(вырезай,1)

(прибудут-с,1)

(публики,1)

(закладом,1)

(входящих,1)

(расскажит,1)

(поставим,1)

(харламова,1)

(немку-то,1)

(скатертью,1)

(энергично,1)

(большей,1)

(приобресть,1)

(унылый,1)

(взрывом,1)

(мертвую,1)

(возраста,1)

(мнительно,1)

(облегчал,1)

(побелевшею,1)

(развитой,1)

(сладостных,1)

(пасмурный,1)

(оригинальных,1)

(обозначаются,1)

(радовались,1)

(надобностей,1)

(прошлялся,1)

(видная,1)

(приблизив,1)

(переделала,1)

(пожалей,1)

(душевная,1)

(офицеру,1)

(разглядывая,1)

(необъяснимые,1)

(припоминался,1)

(неопределенным,1)

(ободрись,1)

(катехизис,1)

(непонятно,1)

(умирающим,1)

(оборвет,1)

(цепь,1)

(бормотали,1)

(наедине-с,1)

(детство,1)

(отгадываний,1)

Для семантического поиска однокоренных слов используется stemmer, который позволяет выделять общей корень у родственных (однокоренных) слов. Его использование характеризовано вызов метода mapPartitions, который позволяет преобразовать каждый раздел исходного RDD в результат из нескольких элементов.

В процессе лабораторной работы был произведён поиск для 50 самых популярных и редко встречающихся слов с использованием стемминга, приведу пример выполняемого кода:

**Top50 most common stems:**

((раскольник,List(раскольников)),564)

((сон,List(сон, соней, соню, сони, соня, соне)),384)

((разумихин,List(разумихина, разумихин, разумихине, разумихину, разумихиным)),349)

((ивановн,List(ивановна, ивановны, ивановну, ивановне, ивановной)),310)

((петрович,List(петровичу, петровичем, петровиче, петровича, петрович)),289)

((вид,List(видов, видится, видом, видит, вида, видимся, вид, видишь, виду, видят, виде, видите, видя, видах, виды)),233)

((глаз,List(глазах, глаза, глазами, глазом, глаз, глазам)),230)

((сам,List(самым, самая, самую, самое, самыми, самою, самый, самых, самые)),224)

((катерин,List(катериною, катерина, катерине, катерину, катериной, катерины)),220)

((сво,List(своя, свой, своем, своею, своим, своему, своими)),215)

((друг,List(другими, другою, другом, друга, друг, другим, другому, другую, другого, другу, друге)),212)

((комнат,List(комнатах, комнату, комнате, комнаты, комнат, комната)),209)

((дом,List(дома, домом, домам, дом, домов, домой, доме, домах, дому)),206)

((дун,List(дуню, дуней, дуни, дуня, дуне)),203)

((порфир,List(порфирий, порфирия, порфирием, порфирии, порфирию)),203)

((говор,List(говорят, говоря, говорившую, говорила, говорили, говорите, говоривших, говори, говорившего, говориться, говорено, говор, говорится, говорим, говоришь)),201)

((раскольников,List(раскольникове, раскольниковым, раскольниковой, раскольникову, раскольникова)),199)

((так,List(таким, таков, така, таком, таки, такими, такою, таких, такую, таку, такому, такого)),196)

((петр,List(петра, петр, петром, петре, петру)),178)

((мо,List(моей, моего, моим, моему, мою, моими, мое, моем, моих, моею)),172)

((час,List(часов, часа, часом, часам, час, часах, часы, часу)),170)

((род,List(рода, родю, родиться, родей, роди, родов, родилась, род, родилось, родя, роде, роду, родах)),167)

((рук,List(руке, руках, рукой, рукою, рук, руками, рука, рукам)),164)

((как,List(каков, каким, како, какими, каком, какою, какому, каку, какое, какие, каких, какого, какую)),163)

((ваш,List(вашего, вашему, вашим, вашими, вашей, вашу, ваших, вашем, вашею)),161)

((сдела,List(сделал, сделано, сделала, сделаешь, сделанная, сделался, сделан, сделает, сделали, сделается, сделай, сделайте, сделав, сделать, сделалось, сделаете, сделаны, сделают, сделаться, сделались, сделаю, сделалась)),161)

((котор,List(которою, котором, которое, которым, которому, которую, которыми)),159)

((стран,List(странного, странное, странен, странным, странную, странная, странными, странной, странны, странных, странном, странные, странною, страннее, странный, странно)),152)

((понима,List(понимаю, понимают, понимающею, понимаете, понимать, понимавший, понимает, понимал, понимала, понимаешь, понимали, понимающим, понимаем, понимавшая)),152)

((одн,List(одних, одному, одну, одном, одною, одними, одни)),151)

((брат,List(брату, брате, братья, братом, браться, брата, брат, брать)),150)

((квартир,List(квартира, квартирам, квартирами, квартир, квартиры, квартиру, квартирой, квартире, квартирах)),146)

((прав,List(правило, правой, праве, правому, правая, праву, правую, правила, правого, правы, правил, правою, правее, правом, правый, права, право, прав)),145)

((цел,List(цели, целых, цель, целые, цел, целует, целию, целый, целом, целью, целей, целая, целое, целого, целыми, целями, целям, целую, целым)),144)

((дел,List(дела, делами, делам, делом, делах, деле, делился, делая, делу)),143)

((последн,List(последние, последнею, последнем, последних, последний, последней, последнего, последняя, последнее, последнюю, последним)),142)

((свидригайл,List(свидригайлов)),141)

((слов,List(словам, слов, слово, словами, словом, словах, слове, слову)),140)

((нача,List(началась, началось, начать, начались, начало, начали, начался, начал, начавший)),140)

((лестниц,List(лестницами, лестница, лестниц, лестницей, лестницы, лестнице, лестницу)),133)

((случа,List(случаются, случаю, случалось, случались, случаях, случаи, случается, случае, случаев, случай, случаем)),133)

((дума,List(думала, думали, думаешь, думает, думать, думающих, думаю, думают, думалось, думаете, думайте)),130)

((мест,List(местов, местам, местами, мест, места, мести, место, месте, месту, местах)),128)

((лиц,List(лицах, лице, лицу, лица, лицом, лицами, лиц)),128)

((пульхер,List(пульхерия, пульхерией, пульхерию, пульхерии)),127)

((сто,List(стоили, стой, стоя, стоят, стоила, стоите, стоило, сто, стоим, стоить, стою, стоит)),127)

((александровн,List(александровну, александровны, александровной, александровне, александровна)),127)

((брос,List(бросив, броситься, бросились, бросить, бросилась, бросилось, бросаясь, бросило, бросили, бросила, бросился, бросившись, бросил)),125)

((замет,List(заметов, заметив, заметь, заметить, заметишь, заметит, заметим, заметили, заметила, заметят)),122)

((сердц,List(сердца, сердцами, сердцах, сердце, сердцу, сердцем)),121)

**Top50 least common stems:**

((поцелу,List(поцелуев)),1)

((чер-р-рт,List(чер-р-рт)),1)

((прибудут-с,List(прибудут-с)),1)

((примыка,List(примыкавшей)),1)

((предвидел,List(предвидел)),1)

((притрогива,List(притрогивалась)),1)

((жертвочк,List(жертвочка)),1)

((захож,List(захожу)),1)

((кровав,List(кровавиться)),1)

((близост,List(близость)),1)

((что-либ,List(что-либо)),1)

((суть-с,List(суть-с)),1)

((изготовля,List(изготовлять)),1)

((отрезвля,List(отрезвляющее)),1)

((помыка,List(помыкают)),1)

((находк,List(находку)),1)

((сгор,List(сгорит)),1)

((разыгрыва,List(разыгрываешь)),1)

((непорядочн,List(непорядочно)),1)

((наряжа,List(наряжаются)),1)

((лужинск,List(лужинская)),1)

((сущность-т,List(сущность-то)),1)

((заглянув,List(заглянув)),1)

((натуру-т,List(натуру-то)),1)

((фатера-т,List(фатера-то)),1)

((замаха,List(замахал)),1)

((забвен,List(забвение)),1)

((созерцан,List(созерцание)),1)

((просвечива,List(просвечивалось)),1)

((пидерит,List(пидерита)),1)

((вышиб,List(вышиб)),1)

((стрем,List(стремиться)),1)

((катехизис,List(катехизис)),1)

((выним,List(вынимая)),1)

((блажен,List(блаженным)),1)

((похва,List(похвала)),1)

((ягод,List(ягоды)),1)

((оборвет,List(оборвет)),1)

((наедине-с,List(наедине-с)),1)

((черноглаз,List(черноглаза)),1)

((искореня,List(искореняет)),1)

((тарака,List(таракана)),1)

((жалост,List(жалости)),1)

((робе,List(робея)),1)

((аллегор,List(аллегории)),1)

((задумался),List(задумался))),1)

((куролес,List(куролесил)),1)

((женат,List(женат)),1)

((квартирой-т,List(квартирой-то)),1)

((отечеств,List(отечества)),1)

# Листинг кода

package LabTwo  
  
import org.apache.log4j.Level.WARN  
import org.apache.log4j.LogManager  
import org.apache.spark.rdd.RDD  
import org.apache.spark.{SparkConf, SparkContext}  
import org.tartarus.snowball.ext.russianStemmer  
  
object LabTwo {  
 val PATH: String = "src/main/data"  
 val NODES: Int = 3  
  
 def main(args: Array[String]): Unit = {  
 val conf: SparkConf = new SparkConf().setAppName("Lab2").setMaster(s"local[$NODES]")  
 val sc: SparkContext = new SparkContext(conf)  
 LogManager.getRootLogger.setLevel(WARN)  
  
 val book: RDD[String] = sc.textFile(s"$PATH/var.txt")  
 val stop: Array[String] = sc.textFile(s"$PATH/stop.txt").collect()  
 val text: RDD[(String, Int)] = parse(book = book, stop = stop)  
  
 println("\n Top50 most common words: ")  
 val most: Array[(String, Int)] = popular(text = text, ascending = false)  
 most.foreach(println)  
  
 println("\n Top50 least common words: ")  
 val least: Array[(String, Int)] = popular(text = text, ascending = true)  
 least.foreach(println)  
  
 val stemmed: RDD[((String, Iterable[String]), Int)] = text  
 .mapPartitions(stemming)  
 .groupBy(\_.\_2)  
 .map(w => (w.\_1, w.\_2.map(\_.\_1.\_1)) -> w.\_2.size)  
  
 println("\n Top50 most common stems: ")  
 val mostStemmed: Array[((String, Iterable[String]), Int)] = stemmed  
 .sortBy(\_.\_2, ascending = false)  
 .take(5)  
 mostStemmed.foreach(println)  
  
 println("\n Top50 least common stems: ")  
 val leastStemmed: Array[((String, Iterable[String]), Int)] = stemmed  
 .sortBy(\_.\_2, ascending = true)  
 .take(5)  
 leastStemmed.foreach(println)  
 }  
  
 private def stemming(iter: Iterator[(String, Int)]): Iterator[((String, Int), String)] = {  
 val stemmer: russianStemmer = new russianStemmer  
 iter.map(w => (w.\_1, 1) -> {  
 stemmer.setCurrent(w.\_1)  
 stemmer.stem  
 stemmer.getCurrent  
 })  
 }  
  
 private def popular(text: RDD[(String, Int)], ascending: Boolean): Array[(String, Int)] = {  
 text.sortBy(\_.\_2, ascending = ascending).take(num = 5)  
 }  
  
 private def parse(book: RDD[String], stop: Array[String]): RDD[(String, Int)] = book  
 .flatMap(\_.toLowerCase.split(" ")  
 .map(\_.replaceAll("[;:,.!?\"«» “–]", ""))  
 .filter(word => word.length > 1 && !stop.contains(word)))  
 .map(word => (word, 1))  
 .reduceByKey(\_ + \_)  
}

# Вывод

В ходе лабораторной работы я настроил сборку проекта с помощью Maven для версии языка Scala 2.11.0. Подключил основную библиотеку для фреймворка Spark и написал исходный код программы для анализа романа Ф. М. Достоевского «Преступление и наказание». В процессе были выявлены 50 самых популярных и 50 самых редко используемых слов. Полученные результаты были дополнены процессом стемминга, что позволило глубже изучить семантический анализ текста. Благодаря расширяемому списку стоп-слов проект может быть расширен в рамках более детального рассмотрения.