National University of Computer and Emerging Sciences

Laboratory Manual-11

for

Fundamentals of Big Data Lab

Course Instructor: Dr Iqra Safdar
Lab Instructors: Mr. Muhammad Mazarib; Mr Muhammad Aiss Shahid
Section: BDS-4B
Date: 26-Apr-2023
Semester: Spring 2023

Department of Computer Science

FAST-NU, Lahore, Pakistan

Pair RDDs in PySpark

Creating pair RDDs

- Two common ways to create pair RDDs
- From a list of key-value tuple
- From a regular RDD
- Get the data into key/value form for paired RDD

```
my_list = ['Sam 23', 'Mary 34', 'Peter 25']
regularRDD = sc.parallelize(my_list)
pairRDD_RDD = regularRDD.map(lambda s: (s.split(' ')[0], s.split(' ')[1]))
```

Transformations on pair RDDs

- All regular transformations work on pair RDD
- Have to pass functions that operate on key value pairs rather than on individual elements
- Examples of paired RDD Transformations
    - reduceByKey(func): Combine values with the same key
    - groupByKey(): Group values with the same key
    - sortByKey(): Return an RDD sorted by the key
    - join(): Join two pair RDDs based on their key

reduceByKey() transformation:

- reduceByKey() transformation combines values with the same key
- It runs parallel operations for each key in the dataset
- It is a transformation and not action

```
regularRDD = sc.parallelize([("Messi", 23), ("Ronaldo", 34),("Neymar", 22), ("Messi", 24)])
pairRDD_reducebykey = regularRDD.reduceByKey(lambda x,y : x + y)
pairRDD_reducebykey.collect()
```

sortByKey() transformation:

- sortByKey() operation orders pair RDD by key
- It returns an RDD sorted by key in ascending or descending order

```
pairRDD_reducebykey_rev = pairRDD_reducebykey.map(lambda x: (x[1], x[0]))
pairRDD_reducebykey_rev.sortByKey(ascending=False).collect()
```

[(47, 'Messi'), (34, 'Ronaldo'), (22, 'Neymar')]

groupByKey() transformation:

- groupByKey() groups all the values with the same key in the pair RDD
  ```
  airports = [("US", "JFK"),("UK", "LHR"),("FR", "CDG"),("US", "SFO")]
  regularRDD = sc.parallelize(airports)
  pairRDD_group = regularRDD.groupByKey().collect()
  for cont, air in pairRDD_group:
  print(cont, list(air))
  ```

join() transformation:

- join() transformation joins the two pair RDDs based on their key
  ```
  RDD1 = sc.parallelize([("Messi", 34),("Ronaldo", 32),("Neymar", 24)])
  RDD2 = sc.parallelize([("Ronaldo", 80),("Neymar", 120),("Messi", 100)])
  RDD1.join(RDD2).collect()
  ```

Transformations list: https://spark.apache.org/docs/latest/rdd-programming-guide.html#transformations
Actions List: https://spark.apache.org/docs/latest/rdd-programming-guide.html#actions

reduce() action:

- reduce(func) action is used for aggregating the elements of a regular RDD
- The function should be commutative (changing the order of the operands does not change the result) and associative
- An example of reduce() action in PySpark
  ```
  x = [1,3,4,6]
  RDD = sc.parallelize(x)
  RDD.reduce(lambda x, y : x + y)
  ```

saveAsTextFile() action:

- saveAsTextFile() action saves RDD into a text ,le inside a directory with each partition as a separate file
  ```
  RDD.saveAsTextFile("tempFile")
  ```
- coalesce() method can be used to save RDD as a single text file
  ```
  RDD.coalesce(1).saveAsTextFile("tempFile")
  ```

Action Operations on pair RDDs

- RDD actions available for PySpark pair RDDs
- Pair RDD actions leverage the key-value data
- Few examples of pair RDD actions include
  - countByKey()
  - collectAsMap()

countByKey() action:

- countByKey() only available for type (K, V)
- countByKey() action counts the number of elements for each key
- Example of countByKey() on a simple list

```
rdd = sc.parallelize([("a", 1), ("b", 1), ("a", 1)])
for ke, val in rdd.countByKey().items():
print(ke, val)
```

collectAsMap() action:

- collectAsMap() return the key-value pairs in the RDD as a dictionary
- Example of collectAsMap() on a simple tuple

```
sc.parallelize([(1, 2), (3, 4)]).collectAsMap()
```

LAB TASKS

1. Create a pair RDD named Rdd with tuples (1,2),(3,4),(3,6),(4,5). Transform the Rdd with reduceByKey() into a pair RDD Rdd_Reduced by adding the values with the same key. Collect the contents of pair RDD Rdd_Reduced and iterate to print the output. Output should be like this:

```
Key 1 has 2 Counts
Key 3 has 10 Counts
Key 4 has 5 Counts
```

2. Read the input file using PySpark and create an RDD. Count the occurrence of word in the file. [Hint]: use flatMap() and map() functions for this task.

3. You have key-value pair rdd similar to task1; Count the unique keys and assign the result to a variable total. What is the type of total? Iterate over the total and print the keys and their counts. Output should be look like this.

```
key 1 has 1 counts
key 3 has 2 counts
key 4 has 1 counts
```

4. The volume of unstructured data (log lines, images, binary files) in existence is growing dramatically, and PySpark is an excellent framework for analyzing this type of data through RDDs. In this task you

are going to implement your knowledge related to RDDs to find the frequency of word in a text document. The text document is attached with the manual related to “Works of William Shakespeare ”. The following steps should be followed to complete this task.

- Create a base RDD from Complete_Shakespeare.txt file.
- Use RDD transformation to create a long list of words from each element of the base RDD.
- Count the total words in splitRDD
- Remove stop words from your data. Convert the words in splitRDD in lower case and then remove stop words from stop_words curated list.
- Create pair RDD where each element is a pair tuple of ('w', 1)
- Get the count of the number of occurrences of each word (word frequency) in the pair RDD.
- Group the elements of the pair RDD by key (word) and add up their values.
- Swap the keys (word) and values (counts) so that keys is count and value is the word.
- Finally, sort the RDD by descending order and print the 10 most frequent words and their frequencies.