**Assignment 2**

MET CS777

Yiduo Feng

09/18/2022

Faculty - Farshid Alizadeh-Shabdiz, PhD, MBA

## **Problem 1 (10 points)**

What are the differences between following RDD operations from functionality’s perspective, and also computation and cost of operation.

Assign a level of computation complexity from level 1 (less costly) to level 3 (most costly) to each one.

1. aggregateByKey() - level 2

Similar to reduceByKey, it can aggregate the values for each key. It can aggregate the value which has different type from that of input.

1. reduceByKey() – level 2

According to the notes from course, data must be in (key, value) pairs, and then shuffle and organize into (k, (v1, v2, …, vn)) pairs. At last, the function “reduces” the list to a single value.

1. groupByKey() – level 3

Similar to reduceByKey, it organize the values into one key pairs, which stores each list as a ResultIterable for future processing (from course lecture2 slide). However, when using groupByKey, since it does not receive functions and without reduce, spark need to separate and move all pairs before processing. The consequence is that the cost between cluster nodes is large, which makes speed slow.

1. combineByKey() – level 1

The computation of combineByKey is same as reduceByKey but API. In reduceByKey, it calls combine function so it should be slower than comByKey. Also, aggregateByKey used comb function.

# **Problem 2 (5 points)**

Name at least four differences between Spark and Hadoop MapReduce?

1. Since Spark is based on memory for computing, when faced with big data, various problems will arise. MapReduce is based on disk processing data.
2. Offline batch programs written on Spark RDDs run faster than MapReduce.
3. Spark is a multi-threaded model, each task runs as a thread and can share resources. MapReduce is a multi-process model
4. MapReduce is not flexible enough which only supports map and reduce.

# **Problem 3 (5 points)**

How does Spark run an application and what are the driver layer functionalities? Explain from the Spark architecture’s perspective.

Build the running environment of Spark Application, SparkContext registers with the resource manager and applies for running Executor resources;

The Driver in Spark creates a SparkContext. The purpose of creating a SparkContext is to prepare the running environment of the Spark application. In Spark, the SparkContext is responsible for communicating with the ClusterManager, applying for resources, assigning and monitoring tasks, etc.; when the Executor part is completed, the Driver is responsible for closing the SparkContext.

The resource manager allocates Executor resources and starts the StandaloneExecutorBackend, and the running status of the Executor will be sent to the resource manager with the heartbeat.

SparkContext builds a DAG graph, decomposes the DAG graph into Stages, and sends the Taskset to the Task Scheduler. Executor applies for Task to SparkContext, Task Scheduler distributes Task to Executor to run, and SparkContext distributes application code to Executor.

Task runs on Executor and releases all resources after running.

# **Problem 4 (5 points)**

What are the differences of running on multi-core computer versus running on multi worker/executor environment?

List pros and cons of each one.

Multi-core:

* Adavantage:

Low recurrence

Finish more work with low cost

Better fault-tolerant

* Disavantage:

Low speed

Not easy to oversee

Multi worker:

* Advantage:

Process fast

Share processor work

High throughput

* Disadvantage:

High memory

deadlock

# **Problem 5 (10 points)**

Why RDD is immutable? Was this a mistake in design of RDDs or has some advantage?

It has some advantage.

1. Easily recreate the RDD
2. Safe sharing of information in the case of high concurrency and multithreading
3. Using caching to improve computation

# **Problem 6 (10 points)**

Spark transformation is divided to narrow transformation and wide transformation. Referring to spark documentation, explain the differences.

Narrow transformation computes in single partition, and data will not move. Wide transformation computes in many partition, and data will move and be shuffled.

# **Problem 7 (15 points)**

List 10 spark transformation operation with one line of example

1. map(func)

rdd.map(lambda x: (x[1],x[2]))

rebuild rdd into a new rdd with two cloumn

1. flatMap(func)

rdd.flatMap (lambda x: (1,x))

(2,3,4) -> (1,1,1,2,2,3,3)

Similar to map(func), but it can map to 0.

1. groupByKey()

rdd = rdd. groupByKey(add)

Similar to reduceByKey, ((a,1)(a,1)) -> (a,2)

1. filter(func)

positive = rdd.filter(lambda x: x > 0)

get the positive element

1. union(rdd)

rdd1 = rdd2.union(rdd3)

similar to join

1. sortByKey()

rdd = rdd.sortByKey()

((2,13),(1,1).(3,4)) -> ((1,1),(2,13),(3,4))

1. distinct()

df = df.distinct()

remove the duplicate row by first column

1. join(rdd)

rdd1 = rdd2.join(rdd3)

combine 2 rdd

1. cartesian(rdd)

rdd.cartesian(rdd)

all possible combinations

1. reduce(func)

list <- 1,23

list.reduce(min) -> 1

List 5 spark action operation with one line example.

1. collect()

print(rdd.collect())

print rdd

1. count()

count = rdd.filter(case).count()

count the number of element after filting

1. first()

df.select(first(column\_name))

get the first element of the column

1. countByKey()

sc.parallelize(((a,1),(a,2),(b,1))).countByKey().items()

-> (a,2) (b,1)

1. countByValue()

sc.parallelize(((a,1),(a,2),(b,1))).countByValue().items()

-> (1,2) (2,1)

# **Problem 8 (20 points)**

We have a data which consists of following columns

* Row number
* First name of student
* Last name of student
* Course number
* Grade

Write an efficient Spark code to calculate

1. Min grade of each student
2. Max grade of each student
3. GPA
4. Number of courses taken

# 

# **Problem 9 (20 points)**

Estimation area of a circle:

Use Spark to estimate area of the unit circle by "throwing darts" at the circle.

Assume you don’t know how to calculate area of a circle in a closed form, but you know how to calculate area of a square. You throw random darts/points in the 2 by 2 square ((-1, -1) to (1,1)) and count how many falls in the unit circle, a circle with radius of one. The fraction can be used to estimate of the area of the unit circle.

