Tariq Anwar
CS6240-1
Hw4

Describe the steps taken by Spark to execute your source code. In particular, for each method invocation of your Scala Spark program, give a brief high-level description of how Spark processes the data. (10 points).

The key concept in Spark is to write program in terms of RDD and Transformation. RDDs are collection of objects spread across a cluster, stored in RAM or on disk, which can also rebuilt in case of failures.
Transformation are action performed on RDD, it’s when the actual processing starts.

The page rank program in Scala first creates the RDD from the file using SparkContext.readfile().

Each line of the above RDD is parsed using the SAX parser and a new RDD is created with adjacency list representation of graph. A transformation map is performed on each line.

The above RDD is further cleaned to filter out invalid pages and create a structure of graph in adjacency list format which are grouped by keys. In this process multiple transformation is performed such as map and filter. The values for each unique key is also combined using group by transformation on the RDD and unique set of outlinks is created for each key. This RDD is final representation of graph, we cache this RDD as this structure will be used multiple times in our program. This RDD is represented as below
 RDD[(String, Iterable[String])]

We calculate total no of nodes in the graph by using key value of above RDD and calling count().
The count transformation is action and it will retrieve a value from RDD

From the above RDD we calculate the sink nodes RDD
Next we create Rank RDD which initializes each node with initial page rank , since we already have total no of nodes.

After all of the above transformation and RDD manipulation we have two final RDDs as below:

val links = // load RDD of (url, neighbors) pairs

var ranks = // load RDD of (url, rank) pairs
Later in the pagerank iteration we join both these RDD using join() and calculate the contribution for each outgoing nodes, we also calculate the sink node mass in each iteration. We use these values to calculate the individual page rank by using the Page Rank formula.

After 10 iteration the RDD is sorted and new RDD is created.

The sorted RDD is saved into an external file.
The key thing to notice is that Spark process all the data using RDDs and series of transformation on those RDDs.
The RDD are lazy evaluated i.e. they are not evaluated until an action occurs in thecode.

Tariq Anwar
CS6240-1
Hw4

Compare the Hadoop MapReduce and Spark implementations of PageRank. (10 points)

• For each line of your Scala Spark program, describe where and how the respective functionality is implemented in your Hadoop jobs.

1.To build the graph structure and to find the initial total number of nodes , a preprocessing job is created in Hadoop implementation whose task was to call the SAX parser and build the adjacency list, the reducer also counts the total no of nodes using global variable.

In Spark, we read the file and apply a map transformation and pass the SAX program as parameter, this will give the graph in adjacency list format although some more transformation is also performed such as filter to remove the invalid links.

2.Calcualtion of Sink Nodes.
In Hadoop, this task was performed by Mapper in PageRankMapper class, while emitting the graph structure to the reducer we check if the node is sink node and add the mass of sink node using global variable counter. We retrieve the total sink mass in Page Rank reducer using the same global counter variable. The mass retrieved is calculated in previous iteration. However in spark implementation we can get access to sink node mass in each iteration.

3.Calculation of rank for each node
In Hadoop implementation each node has associated adjacency list , the first element of the list is the page rank of that particular node. This rank value is updated in Page Rank Reducer program after calculation of pagerank using the formula.

In spark, the rank of each page is stored in the rank RDD, this RDD is joined with links RDD in each iteration to find the page rank of each node and page rank is calculated and rank RDD is updated in each iteration.

4.Sorting of data after 10th iteration.
Once the final iteration of map reduce job is calculated, the output is written to HDFS, a new job Top-N is created to find out top 100 result of the page rank calculation.
In spark, this is done through creation of new RDD, which uses the sort by function on values of the rank RDD. Using the take(100) function we can retrieve the top 100 result of the page rank.

Discuss the advantages and shortcomings of the different approaches. This could include, but is not limited to, expressiveness and flexibility of API, applicability to PageRank, available optimizations, memory and disk data footprint, and source code verbosity.

Advantages of SPARK:

1.Since Spark process all of its data in main memory , the biggest advantages it gives is the response time.
2.Code verbosity: Code size is very small as compared to Hadoop because of flexible Rich and powerful APIs provided by Scala.
3. Since PageRank is iterative algorithm , Hadoop writes and reads data from HDFS in each iteration , this is major performance bottleneck for Hadoop . Spark tries to

accommodate as much of the input data in main memory , hence in algorithm such as page rank which are iterative in nature , it does not need to read and write data from file system hence performance is much better as compared to Hadoop.

Flexiblity of API: Spark has much powerful APIs to perform different tasks which in Hadoop might take very large line of codes to perform the same task.

Memory and Disk Data Footprint: Spark tries to utilize the main memory as much as possible because by design RDDs are stored in main memory for execution, if it does not fit than the RDD might spill to disk. Memory footprint is higher but it comes with the advantage that the result output is quick.

Advantage/Disadvantage of Hadoop

In case of iterative task such as page rank, Hadoop performs poorly because in each iteration it reads and writes to disk.

The only advantage of Hadoop over Spark is in non-iterative process with very huge size of input data, its performance could be better than spark.

The disk footprint is very high in Hadoop as compared to Spark.

Hadoop does not offer great set of APIs to perform different task on data because of which map reduce program are highly verbose as compared to Spark Programs.

Performance Comparison

- 6 m4.large machines (1 master and 5 workers)

Spark Running Time: 88.96 Minutes

Hadoop Running Time: 51 Minutes

- 11 m4.large machines (1 master and 10 workers)

Spark Running Time: 40 Minute

Hadoop Running Time: 30.5 Minutes

Discuss which system is faster and briefly explain what could be the main reason for this performance difference. (4 points)

According to the above statistics, the Hadoop seems to be performing better than the Spark, but this is due to the fact that Spark loads all data into main memory, if it cannot load all data into memory it will store it on the disk but when the cluster size is increased , from 5 to 10 , more memory is available to Spark , hence we can see that it's running time is less than half of 5 cluster set up. Although increasing the cluster size also improved the Hadoop version but that improvement is not significant as compared to Spark version.

Are the results the same for to 100 pages? If not, try to find possible explanations.

The top 100 results from both version are almost same , except that the order of some of pages has changed. The reason behind the change in order of some of pages is that, total sink node calculation in the Hadoop version of my program was less than the one calculated in Spark. I verified it with some small graph and found that some sink nodes in the adjacency list values were not taken into account in Hadoop version which I corrected in Spark and that might be the reason behind change in order of some of nodes pagerank.