CS60021: Sample Questions: Mapreduce

Sourangshu Bhattacharya

1. Answer the following questions in 1 – 2 sentences:

   (a) What is the difference between closure and lambda, in functional programming ?

   (b) What information does NameNode store ?

   (c) Does persist function of RDD store information in HDFS ? If not where ?

   (d) Can spark run in distributed mode without an HDFS installation ? Give reasons.

   (e) Which operations define boundaries of stages in spark execution ?

   (f) Which component of spark is responsible for "computing" the lineage graph ?

   (g) What is the utility of combiners, in hadoop map-reduce ?

[7×1]

2. Answer the following questions:

   (a) What is a partitioner in spark ? Explain the utility of partitioner in the context of repeated transformations. What type of transformations benefit from partitioners ?

   (b) Write three main differences between Hadoop Distributed File System and Network File System.

   (c) Explain the concept of pipelined write in HDFS. What is it expected to achieve, over a non-pipelined write.

   (d) Write the purpose of shuffle, in a map-reduce framework. Clearly mention the tasks performed by the sub-phases of shuffle: "copy phase" and "sort phase".

   (e) What are the differences between "actions" and "transformations" in spark ?

[3 + 2.5 + 2.5 + 3 + 2]

3. Assume that you are given N integers, in the range 0 to $MAXNUM$, in an unsorted manner in RDD `"data"`; and a specified number of partitions, *numparts*.

   (a) Write spark code snippets (in Scala) for finding their median in distributed manner such that no single node has to store or process the entire data, using the following algorithm. You can use methods for sorting a local array.

   - Distribute data in *numparts* ordered partitions such that i^{th} partition has $\frac{(i-1)\times MAXNUM}{numparts} - \frac{i\times MAXNUM}{numparts}$.
   - Count the number of integers in each partition and identify the partition containing the median in driver (centrally).
   - Return the median from the relevant partition.

   (b) What is the total amount of data shuffled ?

[12 +3]

4. Consider the following Spark program for calculating Pagerank for a given graph:

```
val links = sc.textFile("links.txt").
            map{a => val (b, c) = a.split(" ").splitAt(1); (b(0), c.toList)}
var ranks = links.mapValues(v => 1.0)

for(i <- 0 until 10) {
  val contributions = links.join(ranks).flatMap {
    case (pageId, (neighbors, rank)) =>
      links.map(dest => (dest, rank / neighbors.size))
  }

  ranks = contributions.reduceByKey((x, y) => x + y).
                            mapValues(v => 0.15 + 0.85*v)
}
ranks.collect()
```

(a) Draw the lineage graph for the above code. Using dotted lines separate the stages. How many stages will it take to execute the entire job ?

(b) Use partitioner and persist to write a more efficient version of the program. How many strings are shuffled in the above code, for each iteration. How many are shuffled in the modifed program ?

(c) Can you use broadcast to do a one time transfer of the `links` RDD, followed by local updates of ranks ? If yes write the code, else justify with reasons.

[7+5+3]