Data Intensive Computing - Review Questions 2

Daniele Montesi, Francesco Staccone

1. Imagine you are working in a big company, and your company is planning to launch the next big Blogging platform. Tomorrow morning you go to your office and see the following mail from your CEO regarding a new work. How do you answer to this email? Hint: use MapReduce to solve it, and explain what you do in Map and Reduce phases.

   The text collected across all the blogs ever written on the blogging system is used as the input data for the MapReduce procedure. As every input to a MapReduce, job is divided into fixed-size pieces called input splits, each of which is consumed by a single mapper. In the Map phase data in each split is passed to a mapping function to produce output values. In our case, the task of the Map phase is to count the number of letters in each word from input splits and prepare a key-value list in the form of ¡n.of letters, 1¿. In the Shuffle phase, then, emitted ¡key, value¿ pairs are then grouped together according to their keys and passed to a single machine, which runs the Reduce script over them. In this phase, output values from the Shuffle phase are aggregated counting the number of words with the same amount of letters, so that we can get frequencies. Thus, we want our Reduce phase to sum the values of the collection of pairs having the same key. In particular, following the request received by the CEO, we will consider only key "1" and key "2", managed by two different reducers. We are not interested in the outputs provided by the other reducer machines, since the total amounts of 1-letter and 2-letters words were asked.

2. Briefly explain the differences between Map-side join and Reduce-side join in Map-Reduce?

   Join is an operation used to combine two or more database tables based on foreign keys. In Map-Reduce, the join is classified into two types depending on where it is actually performed:

   1. Map-side join, when it is performed by the Mapper. The join is performed before data are actually consumed by the map function. To let that happen, the input to each map function must be in the form of a partition and in sorted order. It is also mandatory to have an equal number of partitions, sorted by the join key. In the scenario in which one of the tables is small, a MapReduce local taskis created before the original join MapReduce which reads data of the small table from HDFS and stores it into an in-memory hash table. Then, the in-memory hash table is serialized into a hash table file. Now, when the original MapReduce join task runs, data are moved from the hash table file to the Hadoop distributed cache, that populates the local disk of the mappers, each of which can load this persistent hash table file back into the memory and do the join task as before. This procedure is adequate only when one of the tables is small enough to fit into the memory, so that it can be read just once. In this case, Map-side join minimizes the costs related to sorting and merging in the shuffle and reduce stages, improving the performance of the task.

   2. Reduce-side join, when it is performed by the Reducer. Here there is no need here to have a dataset in a partitioned form: the join operation can be performed on unstructured data as well. The map function emits join key and corresponding tuples of both the tables. As a result of this processing, all the records with same join key reach the same reducer which then joins them. Compared to the Map-side join, it is easier to implement since we can take advantage of the inbuilt sorting and shuffling algorithm in the MapReduce framework that sends the values having identical keys to the same reducer, therefore data are already organized. On the other hand, the Reduce-side join generates a huge network I/O traffic in the sorting and reducer phase, thus there is chance to encounter an OutOfMemory Exception in case of a large number of different datasets with millions of values.

3. Explain briefly why the following code does not work correctly on a cluster of computers. How can we fix it?

4. Assume you are reading the file campus.txt from HDFS with the following format: