Questions from Chapter 3 of textbook: Data-Intensive Text Processing with MapReduce

Textbook can be downloaded from: https://lintool.github.io/MapReduceAlgorithms/

Read Chapter 3 and answer following questions:

Introduction

a. Synchronization is the most tricky aspect of designing MR algorithms? Where do you get a chance to perform this? What techniques does a programmer have to control execution and manage the flow of data in MR?

3.1 Local Aggregation

a. How does local aggregation lead to an increase in algorithmic efficiency?

b. What are two techniques for local aggregation?

c. What is meant by "in-mapper combining"? What are the two advantages of this design pattern? What are the two disadvantages?

d. In order to achieve algorithmic correctness, the following should be true: (Fill in the blanks)

Reducer Input (K, V) = Mapper _____ (K, V) = Combiner _____ (K, V) = Combiner _____ (K, V)

e. In the algorithms 3.4 to 3.7, the textbook is illustrating how the input/output (K, V) of combiner and mapper must be modified to compute the mean of values associated with the same key. Since the mean function is not associative and commutative, you cannot use the reducer as the combiner. Identify which of the following operations are commutative and associative?

- finding max value
- finding min value
- finding product of values
- finding the median of the values
- finding 1st and 3rd quartile of values

(If you don't know what they are, see here: http://web.mnstate.edu/peil/MDEV102/U4/S36/S363.html)

3.2 Pairs and Stripes

a. What are some applications of building a term co-occurrence matrix

b. Read the pairs and stripes approach carefully and understand their (K, V) pairs.
Algorithm 3.9 shows the details of the stripes approach. Notice that the associative array is defined within the map method. What could be the problem if it is defined within the initialize method (like shown in Algorithm 3.3 for in-mapper combining). Hint: Associative array for a term can grow pretty large and it will have to be stored in memory.
What is a possible solution?

3.3 Computing Relative Frequencies

a. What is the drawback of absolute counts? What is meant by marginal of a word count?

b. Between the pairs and stripes approach, which one makes relative frequency computation easier?

c. If you want to compute relative frequency using the pairs approach, what are some modifications that need to be performed?

d. Using the pairs approach, how can you compute marginal of a word before the joint counts?

e. What is the order inversion design pattern? Outline the steps for using order inversion for relative frequency calculation?

3.4 Secondary Sorting

a. We know that one way of sorting by value is to use an in-memory data structure at the reducer. There is a memory bottleneck while doing this. Another approach is to let the MR framework take care of the sorting by a part of the value. This is known as secondary sorting or "value-to-key" design pattern. What modifications are needed to execute this design pattern.