Survey on Word Count Problem with Map Reduce using Hadoop

Mr. Shahbaj Khan¹ Ms. Rachan Dasondhi²
¹Research associate ²Assistant Professor
¹,²Jawahar Lal Institute of technology, Borawan

Abstract— In cloud computing, data is moved to a remotely located cloud server. Cloud server faithfully stores the data and return back to the owner whenever needed. Data and computation integrity and security are major concerns for users of cloud computing facilities. Today’s clouds typically place centralized, universal trust in all the cloud’s nodes. Hadoop is founded on MapReduce, which is among the most popular programming items for huge knowledge analysis in a parallel computing environment. In this paper, we reward a particular efficiency analysis, characterization, and evaluation of Hadoop MapReduce WordCount utility.

Key words: Performance analysis, cloud computing, Hadoop WordCount

I. INTRODUCTION

Yesteryear decade features seen your rise regarding cloud calculating [1], an arrangement where businesses in addition to individual users utilize hardware, storage space, and software program of 3rd party companies named cloud providers rather than running their very own computing commercial infrastructure. Cloud calculating offers customers the illusion of needing infinite calculating resources, of which they can use all the or less than they have to have, without being forced to concern themselves with exactly how those resources are offered or maintained [2].

The derivation of big knowledge is indistinct and there are a lot of definitions on huge data. For examples, Matt Aslett outlined massive knowledge as “tremendous data is now virtually universally understood to refer to the recognition of larger business intelligence through storing, processing, and examining data that was previously ignored because of problem of normal data management applied sciences” [5]. Recently, the term of giant data has got a brilliant momentum from governments, industry and research communities. In [6], significant information is outlined as a term that encompasses using tactics to capture, approach, analyze and visualize potentially significant datasets in a cheap timeframe now not obtainable to usual IT applied sciences. The figure below would throw more light to your understanding.

II. MAP REDUCE PROBLEM

Word count is typical examples where Hadoop map reduce developers start their hands on. This sample map reduce is intended to count the no of occurrences of each word in the provided input files. Below line show about Map Reduce Problem.

- **Map()**
  - Process a key/value pair to generate intermediate key/value pairs
- **Reduce()**
  - Merge all intermediate values associated with the same key

Users implement interface of two primary methods:
- **Map**: (key1, val1) → (key2, val2)
- **Reduce**: (key2, [val2]) → [val3]

The point to be noted here is that first the mapper class executes completely on the entire data set splitting the words and forming the initial key value pairs. Only after this entire process is completed the reducer starts. Say if we have a total of 10 lines in our input files combined together, first the 10 lines are tokenized and key value pairs are formed in parallel, only after this the aggregation/reducer would start its operation.

III. WORD COUNT PROBLEM WITH MAP REDUCES

The word count operation takes place in two stages a mapper phase and a reducer phase. In mapper phase first the test is tokenized into words then we form a key value pair with these words where the key being the word itself and value ‘1’. For example consider the sentence “tringtring the phone rings”.

In map phase the sentence would be split as words and form the initial key value pair as

- <tring,1>
- <tring,1>
- <the,1>
- <phone,1>
- <rings,1>

In map phase the sentence would be split as words and form the initial key value pair as

In the reduce phase the keys are grouped together and the values for similar keys are added. So here there are only one pair of similar keys ‘tring’ the values for these keys would be added so the output key value pairs would be

- <tring,2>
- <the,1>
- <phone,1>
- <rings,1>

Fig. 1: Flow of Map Reduce
Survey on Word Count Problem with MapReduce using Hadoop

(IJSRD/Vol. 5/Issue 5/2017/050)

All rights reserved by www.ijsrd.com 216

This would give the number of occurrence of each word in the input. Thus reduce forms an aggregation phase for keys.

Algorithm for Word Count using Map-Reduce

```java
Mapper<LongWritable, Text, Text, IntWritable> {
  private static final IntWritable one = new IntWritable(1);
  private Text word = new Text();
  public static void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
    String line = value.toString();
    StringTokenizer tokenizer = new StringTokenizer(line);
    while(tokenizer.hasNext()) {
      word.set(tokenizer.nextToken());
      output.collect(word, one);
    }
  }
}
```

IV. CONCLUSION

Map-Reduce, proposed in this paper provides an online, on-demand and closed-loop solution to managing these faults. The control loop in word count mitigates performance penalties through early detection of anomalous conditions on slave nodes. Anomaly detection is performed through a novel sparse-coding based method that achieves high true positive and true negative rates and can be trained using only normal class (or anomaly-free) data. The local, decentralized nature of the sparse-coding models ensures minimal computational overhead and enables usage in both homogeneous and heterogeneous Map-Reduce environments.

REFERENCES