CLOUD COMPUTING

Project 4: HBase WordCount

Abhishek Naik and Shree Govind Mishra

4th March, 2017

In this project, we have written an HBase WordCount program to count the occurrences of all the unique terms from the clueWeb09 . It maps well to the map-reduce programming model, which implies that we have three main parts in the program:

1. Mapper
2. Reducer
3. The *main()* function

The logic used in the above parts can be described as below:

1. The Mapper:

The static class *WcMapper* extends the *TableMapper* class, while the *map()* function overrides the *map()* function from the ‘*org.apache.hadoop.hbase.mapreduce.TableMapper<Text, LongWritable>*’ class. This provides the input in the form of <key, value> pairs. In this case, it is to be noted that this <key, value> pair is in the form of <rowkey, content>. The *rowkey* is related to a specified URI and the *context* is the corresponding text associated with that URI. The *map()* function outputs <word, frequency> for each word that appears in the context.

In order to do this, we are using a *HashMap hm*. First, we get the hash of all the frequencies. Then, for each entry in the HashMap, we display the corresponding <key, value> pair. This <key, value> pair is further also written to the *context* object. We use the *getWordFreq()* function to get the frequencies of the words in the context.

1. The Reducer:

A reducer assembles the intermediate <key, value> pairs from the various map tasks and builds a single result. For each word, the *reduce()* function calculates the sum of the occurrences of that word and forms <word, occurrence> pairs. Next, it creates a *Put* object to write it down to the *HBase* table via the *context* object.

We calculate the total frequency for the word which is stored in the variable *totalFreq* (initialized with 0). Next we create a *Put* object to add this into the *HBase* table using the *add()* function. The parameters passed to the add function are the column family, name and cell value, respectively. Note that we have referred the *Constants* file here. Finally, the *Put* object is passed as a parameter to the *write()* method of the *context* object.

1. The *main()* function:

The *main()* function is just as it is given. In the *main()* function we are creating a *Configuration* class object *config* along with a *Job* class object *job*. The *main()* function exits when the program has finally completed its execution.

References:

1. <http://lemurproject.org/clueweb09/>