15640 project 3 MAP-REDUCE

Pans@andrew.cmu.edu

dengbow@andrew.cmu.edu

1.Design

Our implementation of map-reduces is similar to the hadoop. We have a configuration file to define the master and slavers IP address. Then we run up the master, the clients(slavers) and register to the master. In the master, we wrote a concurrent hashmap, where key is the clientID and the value is the task. By maintain this hashmap, the master can coordinate the task and clients.

Our MapReduce framework is designed with a single Master coordinating process that handles all allocation of work. A new job can be requested on any node but if it is requested on a participant, it is simply bundled up and sent to the master. Once the master has a new job, it splits the task into inputsplits, which cantains a key of offset and value of the inputsplit length, and then the master begins distributing them to the participants.

For the user, when they want to hand a task to the the map-reduced system. We deal with the task similar to the hadoop, and there is no particular fixed format and length request. Firstly, we create an inputsplit object to split the input file by bytes. It will yield pair of offset and length of each inputsplit list. Then the master will handover these inputsplits into In the mapper. In the mapper, it will call the recorderreader to from a recorder, which cantains key and values, the keys are the offset of the inputfile, and value is the text of each recorder block. In this process, the block will aline to the multiply of lines, it will implement to the end of a line automatically, and will always read one more line. Meanwhile, it will skip the first line if it is not start from 0 position.

Then the mappers will taker over record concurrently. The master will send the check signal to make sure every client is working every 5 second, if the client is down, then the master will choose a worker whose workload is lowest and allot the task to this new worker. For each map, it will receive the message containing a request to show the command and a message including the reducetask. It will contains the ID and inputsplit information. And the mapper can create the recorder by it self. Also, in the maptask, it will contains the instance of mapper, and the mapper can call the map method defined by the user through this way. The master will transfer the instance of mapper to the warker, by calling the forname()method, we can call the map method in the mapper. It will be same for the reducer. When the map method runs, it will read the input text line by line.

The mapper also takes the task to emit the output to sever splitpaths and then we use the merge sort to merge all the splitpaths. Then the mapper will also do the partition job by create a partioner instance. Out partition sorted file by hashcode value of key, and will yield a hasmap cantains key and value.

After the map, it will save the output into a path. And send the feed back to the master. Then when all the mappers are done, The task will handover to the reducer. Then the reducers will use merge sort these outputs files after partition, which put all the key value pairs into one file and then combine these pairs by key. It means put the values of the same key togather. Then it will call the reduce method defined by the user.

After the reducers done their jobs, they just output the result into the target path file in give format. Then the task will be shown to the users.

As for the tolerance , if one worker is down, then we just allocate this task to another woker whose workload is the lowest. And we also has the exception detector, which will return exception during running.

2.Workflow

In the Config file, it contains the IP of each machine, the role is either “MASTER” or “SLAVE”, indicating whether it is a master or slave in this system, and Portnum is to refer to the port number that you would like to connect to for this system. You can only have one master in the system, but any number of slaves. Just make sure that no two systems are running on the same ipaddress and portnum.

Now to run the system, start on the master and run the masterterminal. This will run a master instance, and if necessary slave instances on the same machine.

After this, go onto every slave, and run workerterminal. This will instantiate the slaves, and establish a connection with the master.

Finally, once the system is up and running, you can send a user job request the map-reduce system. To do this, modify the configration, and specify all the necessary details including your input source file, your output source file, the type of input you are using (whether it is key-value type, or text type). You must also specify the details of the Map Reduce classes, implement the reducer and mapper interface. The input for Map is the key and value pair, where key is

3. Example and how to run

The users should define the mapper and reducer interface to define these two method.

Two examples have been provided in the Examples folder.

WordCount: which takes in a large corpus of text and returns the number of times each word occurred in the corpus. The map split the values and erase punctuation. The reduce sums up the values.

Count odd number: which count each odd number and sum them up to the users. The map recoginize the ood number, the reduce sums up the amount for each odd number.

4.Use Introduction

An example of how to run WordCount is shown below:

1.Open several machine, and set this IP into the configuration.

2.First we "make" all the files in all the machines.

3.In the master machine, we just tape "make master"

4.In the client terminal,We just tape "make worker_1" it will automatically runs up the client, the workerID is work_1, we can starts up similarly to start other workers.

5.If you want more worker , you can change the configuration and add it to the make file as the same format of worker 1 to worker 3, then run the make worker_No.

6.Then we start the master by type “start” in the comindline.
In the terminal , there are other commands, such as stop, which will stop the coordinator; monitor, which will check the worker every 5 seconds, and quit to quit this user task.

7.Then we follow the usage massage:
Usage: submit <MapClass> <InputFile> <ReduceClass> <OutputFolder>

8.We will tape like:
“Submit WorldCountMapper Inputfilepath
WorldCountReducer OutputFolderpath”

9.Then the terminal will shows: the task is running .
And the task is finished.

10.When the task is done the output will be laid in the output file, you can view it.

5.Furture Work

1. We can change the two way merge sort to multi-way merge sort, which will increase the efficience when merge sort.

2. Set abondunt master to run if the former master is down

3. Set the checkpoint, to restart and recover the task when the whole system is down.