Hadoop Course

Dissecting MapReduce

Several file formats for sequence etc.

* File formats are very important. There are also custom file format

OutputFormat

* Validate output specifications
* RecordWriter implementation to write files of the job
* The directory doesn’t exist - > Hadoop creates it
* It writes it, to write the output files
* Output format has output write format
* Comes with many implimentations: you can also write custom implimentations

Specify mapper class and reducer class

* MaxClosePriceReducer.class
* setMapperClass // this sets the mapper
* the output key from mapper and reducer is a type “Text”
* from the mapper is a “float”

Writeable wrappers

* writable implementable is intWritable etc.
* why new data types? Writable are used whenever there is a need to transfer data between texts.
* When data is given as data. Distributed contributed. Have mappers and reducers, have a LOT of data transferred data between nodes. The object has to be turned into a byte (serialization)
* Millions of records are processed, so cedialization / serialization needs to be fast and precise

Java serialization is something else

* 3 instances of employee object, the first one has the com.employee
* The space, cannot be compact
* Reference handle, during the shuffle phase, since only the first record, it must be given special care. By not giving the class name to the string, how would you know the type? The CLIENT should know the type (usually true)

Writeables vs standard Java types: when deserialization things happen, new instances of object has to be made, but with writables, they are already there.

Submit job: see progress of job

Mapreduce Part 2

Driver Program was looked in detail

* Input and output format

Look at mapper program and execute the mapreduce

Mapper

* Mapper is divided into multiple parts -> input splits
* Each mapper processes and input split
* Each mapper can be called multiple times depending on the content of input split
* Mapper will emit key value par as output
* One or more mapper in a mapreduce job

Program

* Each line is a record to the mapper
* Output from mapper: trying to find out the maximum closing price for each stock symbol
* Group by symbol. Output the stock symbol as the key
* Key: stock symbol close price: value
* We know the input / output
* Import org.apache.hadoop.mapreduce.Mapper
* Look at the signature: 4 parameters
  + Mapper<longwritable, text, text, floatwriteable)
  + Output key: text, output value: floatwriteable
* First two arguments: argument is long writable: represents is a byte of the record from the beginning of the file. Ignore the argument when processing: we are interested in the argument
* Delimited as a value
* So we extract that from the record: separated from a value
* “line.split (“,”)
* Context.write(…)
  + List of things , reducer is called once per key

Reducer

* Take key value from multiple map functions and reduce them to output
* Keys are grouped with VALUES
* Reduce function is called once per key and its values
* Each symbol has a list of values
* Some cases, you won’t need a reducer
* Some times reducers are not necessary
* Reduce program:
  + 4 type parameters
  + 3 or 4
  + Reduce a floatWritable : list of closing prices, easy to find
  + Use Math.max function
  + Sum the output by calling the context
  + Run as a job
  + Export project out of java
* Click on project: jar file -> give location -> finish
  + It is now a jar file
* WINSCP ->