Discussion Session
Week 8

INF 141: Information Retrieval
Winter 2008

Yasser Ganjisaffar
yganjisa@ics.uci.edu
Some Terminology

• Running “Word Count” across 20 files is one job
• 20 files to be mapped imply 20 map tasks + some number of reduce tasks
• At least 20 map task attempts will be performed... more if a machine crashes.
Task Attempts

• A particular task will be attempted at least once, possibly more times if it crashes

• If the same input causes crashes over and over, that input will eventually be abandoned.
MapReduce: High Level
Nodes, Trackers, Tasks

• Master node runs *JobTracker* instance, which accepts *Job* requests from clients

• *TaskTracker* instances run on slave nodes

• TaskTracker forks separate Java process for task instances
Partitioners

• Partitioners are application code that define how keys are assigned to reduces.
• Default partitioning spreads keys evenly, but randomly
  – Uses key.hashCode() % num_reduces
Default Partitioning in PostingLists

part-00000
a
is
sample
this

part-00001
another
test
example
just
that

sort -m part-* > merged-result.txt
A Better Approach: Custom Partitioning

cat part-* > merged-result.txt
Partitioners

/*Support alphabetical reduce buckets */
public static class MyPartitioner implements Partitioner<Text, Text> {

    public void configure(JobConf arg0) {
    }

    public int getPartition(Text key, Text value, int numPartitions) {
        String s = key.toString();
        if(s.length() > 1){
            return(s.charAt(0) % numPartitions);
        }
        else{
            return(0);
        }
    }
}

/*Write the outputs into alphabetical part-00000 files */
conf.setPartitionerClass(MyPartitioner.class);
conf.setNumReduceTasks(26);
Hadoop Job Scheduling

• FIFO queue matches incoming jobs to available nodes
  – No notion of fairness
  – Never switches out running job

• Warning! Start your job as soon as possible.
Counters

- public static enum Counters{MAPPER,REDUCER};
- reporter.incrCounter(Counters.MAPPER,1);
- reporter.incrCounter(Counters.REDUCER,1);

- [http://palantir.ics.uci.edu:1755/jobtracker.jsp](http://palantir.ics.uci.edu:1755/jobtracker.jsp)
Some Performance Tweaks
Split versus StringTokenizer

Pattern pattern = java.util.regex.Pattern.compile("[^\s]+");
String [] responses = pattern.split(text);

OR?

StringTokenizer st = new StringTokenizer(text, "\t\r\n,/.?`~!@#$%^&\*'()-_=+\|;:'\"<>"");
Which one is better?

```java
StringTokenizer st = new StringTokenizer(content, " ");
while (st.hasMoreTokens()) {
    String token = st.nextToken();
    Text word = new Text(token);
    output.collect(word, docid);
}
```

```java
Text word = new Text(""");
StringTokenizer st = new StringTokenizer(content, " ");
while (st.hasMoreTokens()) {
    String token = st.nextToken();
    word.set(token);
    output.collect(word, docid);
}
```

1. Less Garbage Collection.
Compression?

• Compressing the outputs and intermediate data will often yield huge performance gains
  – Set `mapred.compress.map.output` to true to compress map outputs

• Compression Types (`mapred.output.compression.type`) for SequenceFiles
  – “block” - Group of keys and values are compressed together
  – “record” - Each value is compressed individually
  – Block compression is almost always best

• Compression Codecs (`mapred(.map)?.output.compression.codec`)
  – Default (zlib) - slower, but more compression
  – LZO - faster, but less compression
Valid Terms?

- 14 kilometres
- 1345–1346
Distributed File Cache

• The Distributed Cache facility allows you to transfer files from the distributed file system to the local filesystem (for reading only) of all participating nodes before the beginning of a job.

• Boost efficiency: a node can read the file from its local filesystem.

• DistributedCache.addCacheFile(new URI("/path/to/file_to_distribute"), conf);
More Performance Tweaks

• Hadoop defaults to heap cap of 200 MB
  – Set: mapred.child.java.opts = -Xmx512m
  – 1024 MB / process may also be appropriate

• Use Combiners where possible.
Links

- https://eee.uci.edu/wiki/index.php/INF141_ASS06_01