DATASCI W261: Machine Learning at Scale

Nick Hamlin

- nickhamlin@gmail.com
- Section 3

- Week 1 Async Assignment
- Submitted: January 6, 2016

This notebook provides a poor man's Hadoop through command-line and python.

Map

The mapper file iterates through each line in the chunk sent to it and uses python's re library to count all instances of the desired string in each line. These overall total number of occurrences for the chunk is then printed to stdout

In [25]:
```
%%writefile mapper.py
#!/usr/bin/python
import sys
import re
count = 0
WORD_RE = re.compile(r"[\w']+")
filename = sys.argv[2]
findword = sys.argv[1]
with open (filename, "r") as myfile:
    for i in myfile.readlines():
        line=i.lower() #make our search case insensitive
        temp_count=len(re.findall(findword.lower(),line))
        count+=temp_count
print count #We can just print the result to stdout and redirect it
in the shell script
```

```
Overwriting mapper.py
```

In [27]:
```
!chmod a+x mapper.py
```

Reduce

Since the mapper file does most of the work in this instance, the reducer can be very simple. Here, all we need to do it extract the intermediate total for each chunk and add it to our overall running total.

In [28]:
```
%%writefile reducer.py
#!/usr/bin/python
import sys
sum = 0
for line in sys.stdin:
    sum+=int(line)
print sum
```

```
Overwriting reducer.py
```

In [29]:
```
!chmod a+x reducer.py
```

Write script to file

The rest of the script is already complete, so no changes are needed to any of the code below to make the mapreduce process work. However, it's convenient to add a final line to this script to clean up all the temp files that we create.

In [54]:
```
%%writefile pGrepCount.sh
ORIGINAL_FILE=$1
FIND_WORD=$2
BLOCK_SIZE=$3
CHUNK_FILE_PREFIX=$ORIGINAL_FILE.split
SORTED_CHUNK_FILES=$CHUNK_FILE_PREFIX*.sorted
usage()
{
    echo Parallel grep
    echo usage: pGrepCount filename word chuncksize
    echo greps file file1 in $ORIGINAL_FILE and counts the number o
f lines
    echo Note: file1 will be split in chunks up to $ BLOCK_SIZE chu
nks each
    echo $FIND_WORD each chunk will be grepCounted in parallel
}
#Splitting $ORIGINAL_FILE INTO CHUNKS
split -b $BLOCK_SIZE $ORIGINAL_FILE $CHUNK_FILE_PREFIX
#DISTRIBUTE
for file in $CHUNK_FILE_PREFIX*
do
    #grep -i $FIND_WORD $file|wc -l >$file.intermediateCount &
    ./mapper.py $FIND_WORD $file >$file.intermediateCount &
done
wait
#MERGEING INTERMEDIATE COUNT CAN TAKE THE FIRST COLUMN AND TOTOL...
#numOfInstances=$(cat *.intermediateCount | cut -f 1 | paste -sd+ -
|bc)
numOfInstances=$(cat *.intermediateCount | ./reducer.py)
echo "found [$numOfInstances] [$FIND_WORD] in the file [$ORIGINAL_F
ILE]"

#Finally, clean up all temp files (file names containing a ".spli
t")
find . -type f -name \*.split\* -exec rm {} \;
```

```
Overwriting pGrepCount.sh
```

Run the file

In [56]:
```
!chmod a+x pGrepCount.sh
```

Usage: usage: pGrepCount filename word chuncksize

In [57]:
```
!./pGrepCount.sh License.txt COPYRIGHT 4k
```

```
found [59] [COPYRIGHT] in the file [License.txt]
```

Trying again with a different word and chunksize to make sure everything works

In [59]:
```
!./pGrepCount.sh License.txt Warranty 8k
```

```
found [9] [Warranty] in the file [License.txt]
```