**HU Extension Assignment 07 E185 Big Data Analytics**

**Handed out: 03/30/2013 Due by 5:30PM on Saturday, 04/05/2013**

**The National Bureau of Economic Research (www.nber.org) offers an interesting set of econometric and sociological datasets. From the page http://data.nber.org/patents/, please download file cite75\_99.zip which contains ASCII, comma separated values describing citation data for US patents between years 1975 and 1999. The first column contains ordered list of CITING patents and the second column contains CITED patents. For the initial analysis you might want to extract a small section of cite75\_99.txt file you will obtain when you decompress the archive.**

**My recommendation is that you open an interactive Pig session on Amazon’s Elastic Map Reduce and work on the Master node. Record Pig commands as you develop them so that you could interrupt your AWS sessions. When you shut a grunt session all of your data structures and commands are lost. Keep the text file cite75\_99.txt you extract from the above archive in one of your S3 buckets.**

**Problem 1) Using Pig create a file that contains an ordered list of patents where every patent is followed by the patents that cite it. The list that we want to create looks like the following:**

**1000026 4043055  
1000033 4190903, 4975983  
1000043 4091523  
1000044 4082383, 4055371  
1000045 4290571  
1000046 5918892, 5525001**

**The above table tells us, for example, that patent 1000033 is cited by patents: 4190903 and 4975983. In a way you are inverting the original data. You can organize your output in any data structure you like. A nested data structure would look nice, though. Could you write one or more SQL commands that would accomplish the same objective.**

**As you are working, please describe for us all the intermediary relations (data structures) and transformations you are using.**

We create an Pig Interactive session to work at in AWS and connect to it. As this has been extensively covered in previous lectures, I will not bore you with the details. :) When the instance is up and running, we connect to the master instance from our workstation.

**$ ssh -i ~/.ec2/ec2-keypair hadoop@ec2-174-129-105-253.compute-1.amazonaws.com**

Linux (none) 3.2.30-49.59.amzn1.x86\_64 #1 SMP Wed Oct 3 19:54:33 UTC 2012 x86\_64

--------------------------------------------------------------------------------

Welcome to Amazon Elastic MapReduce running Hadoop and Debian/Squeeze.

Hadoop is installed in /home/hadoop. Log files are in /mnt/var/log/hadoop. Check

/mnt/var/log/hadoop/steps for diagnosing step failures.

The Hadoop UI can be accessed via the following commands:

JobTracker lynx http://localhost:9100/

NameNode lynx http://localhost:9101/

--------------------------------------------------------------------------------

hadoop@ip-10-145-206-54:~$

and before starting to play with the data I copy it to the local machine, after creating an appropriate directory for it.

And we start up the grunt console

**hadoop@ip-10-145-206-54:~$ pig**

2013-04-05 16:43:11,177 [main] INFO org.apache.pig.Main - Logging error messages to: /home/hadoop/pig\_1365180191175.log

2013-04-05 16:43:11,210 [main] INFO org.apache.pig.impl.util.Utils - Default bootup file /home/hadoop/.pigbootup not found

2013-04-05 16:43:11,462 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: maprfs:///

2013-04-05 16:43:11,719 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to map-reduce job tracker at: maprfs:///

In order to get started, we need to load the text file into pig, and we use the LOAD command to do so. The citation file had been uploaded previously to an S3 bucket, and since it is a very nicely formatted CSV, we can cut some corners by using PigStorage() to load it.

**grunt> cit\_list = LOAD 's3://alexcp-e185/Assign07/input/cite75\_99.txt' using PigStorage(',') as (citing:chararray, cited:chararray);**

**grunt> ILLUSTRATE cit\_list**

(...)

-------------------------------------------------------------

| cit\_list | citing:chararray | cited:chararray |

-------------------------------------------------------------

| | 3858721 | 3256975 |

-------------------------------------------------------------

Next, we group our data according to the cited column.

**grunt> cit\_grp = GROUP cit\_list BY cited;**

**grunt> ILLUSTRATE cit\_grp;**

(...)

-------------------------------------------------------------

| cit\_list | citing:chararray | cited:chararray |

-------------------------------------------------------------

| | 3859500 | 3372305 |

| | 3859501 | 3372305 |

-------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------

| cit\_grp | group:chararray | cit\_list:bag{:tuple(citing:chararray,cited:chararray)} |

--------------------------------------------------------------------------------------------------------------

| | 3372305 | {(3859500, 3372305), (3859501, 3372305)} |

--------------------------------------------------------------------------------------------------------------

And we select only the citing information from our tuples that had been created by the GROUP command:

**grunt> cit\_citing = FOREACH cit\_grp GENERATE group, cit\_list.citing;**

**grunt> ILLUSTRATE cit\_citing;**

(...)

-------------------------------------------------------------

| cit\_list | citing:chararray | cited:chararray |

-------------------------------------------------------------

| | 3859804 | 2857744 |

| | 3859806 | 2857744 |

-------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------

| cit\_grp | group:chararray | cit\_list:bag{:tuple(citing:chararray,cited:chararray)} |

--------------------------------------------------------------------------------------------------------------

| | 2857744 | {(3859804, 2857744), (3859806, 2857744)} |

--------------------------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------

| cit\_citing | group:chararray | :bag{:tuple(citing:chararray)} |

-------------------------------------------------------------------------------------

| | 2857744 | {(3859804), (3859806)} |

-------------------------------------------------------------------------------------

And we order it by the cited patent for good measure:

**grunt> cit\_citing\_order = ORDER cit\_citing BY group ASC;**

**grunt> ILLUSTRATE cit\_citing\_order;**

(...)

-------------------------------------------------------------

| cit\_list | citing:chararray | cited:chararray |

-------------------------------------------------------------

| | 3858800 | 2881616 |

| | 3858801 | 2881616 |

| | 3860043 | 2254558 |

-------------------------------------------------------------

--------------------------------------------------------------------------------------------------------------

| cit\_grp | group:chararray | cit\_list:bag{:tuple(citing:chararray,cited:chararray)} |

--------------------------------------------------------------------------------------------------------------

| | 2254558 | {(3860043, 2254558)} |

| | 2881616 | {(3858800, 2881616), (3858801, 2881616)} |

--------------------------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------

| cit\_citing | group:chararray | :bag{:tuple(citing:chararray)} |

-------------------------------------------------------------------------------------

| | 2254558 | {(3860043)} |

| | 2881616 | {(3858800), (3858801)} |

-------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------

| cit\_citing\_order | group:chararray | :bag{:tuple(citing:chararray)} |

-------------------------------------------------------------------------------------------

| | 2254558 | {(3860043)} |

| | 2881616 | {(3858800), (3858801)} |

-------------------------------------------------------------------------------------------

Looks exactly like what we were looking for. Let's export it to an output bucket.

**grunt> STORE cit\_citing\_order INTO 's3://alexcp-e185/Assign07/output-p1';**

(...)

Success!

Job Stats (time in seconds):

JobId Maps Reduces MaxMapTime MinMapTIme AvgMapTime MaxReduceTime MinReduceTime AvgReduceTime Alias Feature Outputs

job\_201304051618\_0011 8 1 90 35 50 72 72 72 cit\_citing,cit\_grp,cit\_list GROUP\_BY

job\_201304051618\_0012 1 1 14 14 14 5 5 5 cit\_citing\_order SAMPLER

job\_201304051618\_0013 1 1 47 47 47 195 195 195 cit\_citing\_order ORDER\_BY s3://alexcp-e185/Assign07/output-p1,

Input(s):

Successfully read 16522439 records (1541399192 bytes) from: "s3://alexcp-e185/Assign07/input/cite75\_99.txt"

Output(s):

Successfully stored 3258984 records in: "s3://alexcp-e185/Assign07/output-p1"

Counters:

Total records written : 3258984

Total bytes written : 0

Spillable Memory Manager spill count : 0

Total bags proactively spilled: 0

Total records proactively spilled: 0

Job DAG:

job\_201304051618\_0011 -> job\_201304051618\_0012,

job\_201304051618\_0012 -> job\_201304051618\_0013,

job\_201304051618\_0013

2013-04-05 18:30:36,730 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!

So, we check the final output out by copying it over to the local machine and checking its head and tail

**hadoop@ip-10-145-206-54:~/Assign07$ mkdir output-p1**

**hadoop@ip-10-145-206-54:~/Assign07$ hadoop fs -copyToLocal s3://alexcp-e185/Assign07/output-p1/\* output-p1/**

13/04/05 18:39:58 INFO metrics.MetricsSaver: MetricsSaver FsShell root:hdfs:///mnt/var/lib/hadoop/metrics/ period:60 instanceId:i-973744f8 jobflow:j-1JKGMPFYP2GTG

13/04/05 18:39:58 INFO metrics.MetricsUtil: supported product mapr-m3

13/04/05 18:39:58 INFO metrics.MetricsSaver: Disable MetricsSaver due to MapR cluster

13/04/05 18:39:59 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p1/\_SUCCESS' for reading

13/04/05 18:40:02 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p1/\_logs/history/ip-10-145-206-54.ec2.internal\_1365178749834\_job\_201304051618\_0013\_hadoop\_PigLatin%3ADefaultJobName' for reading

13/04/05 18:40:02 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p1/part-r-00000' for reading

13/04/05 18:41:06 INFO metrics.MetricsSaver: Inside MetricsSaver Shutdown Hook

**hadoop@ip-10-145-206-54:~/Assign07$ head output-p1/part-r-00000**

"CITED" {("CITING")}

1 {(4647229),(3964859)}

10000 {(4539112)}

100000 {(5031388)}

1000006 {(4714284)}

1000007 {(4766693)}

1000011 {(5033339)}

1000017 {(3908629)}

1000026 {(4043055)}

1000033 {(4190903),(4975983)}

**hadoop@ip-10-145-206-54:~/Assign07$ tail output-p1/part-r-00000**

999961 {(5782495),(5738381),(4262874),(4171117),(4437639),(5878901),(4832301),(4871140),(5048788)}

999965 {(5052613)}

999968 {(3916735)}

999971 {(3965843)}

999972 {(4038129)}

999973 {(4900344),(5427610)}

999974 {(5464105),(4560073),(4728158)}

999977 {(4092587)}

999978 {(3915443)}

999983 {(5806555),(5143114),(5394715)}

The output looks like we expected and is correct.

An equivalent SQL statement would be the following

SELECT c1.cited,

CONCAT(SELECT citing FROM citations c2 WHERE c1.cited = c2.cited, ",")

AS list\_citing

FROM citations c1

ORDER BY cited ASC;

Where CONCAT() is a function that receives the result of a SELECT query and a Separator, and concatenates it into a single string. All existing RDBMS have different implementations of this and a good reference for that would be this Stack Overflow topic:

<http://stackoverflow.com/questions/194852/concatenate-many-rows-into-a-single-text-string>

**Problem 2) Once you are satisfied with the output of your Pig process, place all of your code into a Pig script and demonstrate that you can run that script from the command prompt where the source file or source bucket and the output file or the output bucket are specified as input parameters to the script.**

Here is the generated Pig Script with the commands that were issued before:

**hadoop@ip-10-145-206-54:~/Assign07$ cat p2.pig**

cit\_list = LOAD '$INPUT' using PigStorage(',') as (citing:chararray, cited:chararray);

cit\_grp = GROUP cit\_list BY cited;

cit\_citing = FOREACH cit\_grp GENERATE group, cit\_list.citing;

cit\_citing\_order = ORDER cit\_citing BY group ASC;

STORE cit\_citing\_order INTO '$OUTPUT';

And we can run it from the command line as follows:

**hadoop@ip-10-145-206-54:~/Assign07$ pig -p INPUT=s3://alexcp-e185/Assign07/input -p OUTPUT=s3://alexcp-e185/Assign07/output-p2 file:/home/hadoop/Assign07/p2.pig**

2013-04-05 19:05:58,152 [main] INFO org.apache.pig.Main - Logging error messages to: /home/hadoop/Assign07/pig\_1365188758151.log

2013-04-05 19:05:58,173 [main] INFO org.apache.pig.impl.util.Utils - Default bootup file /home/hadoop/.pigbootup not found

2013-04-05 19:05:58,429 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: maprfs:///

2013-04-05 19:05:58,627 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to map-reduce job tracker at: maprfs:///

(...)

Success!

Job Stats (time in seconds):

JobId Maps Reduces MaxMapTime MinMapTIme AvgMapTime MaxReduceTime MinReduceTime AvgReduceTime Alias Feature Outputs

job\_201304051618\_0014 8 1 69 38 47 70 70 70 cit\_citing,cit\_grp,cit\_list GROUP\_BY

job\_201304051618\_0015 1 1 14 14 14 5 5 5 cit\_citing\_order SAMPLER

job\_201304051618\_0016 1 1 47 47 47 71 71 71 cit\_citing\_order ORDER\_BY s3://alexcp-e185/Assign07/output-p2,

Input(s):

Successfully read 16522439 records (1541399192 bytes) from: "s3://alexcp-e185/Assign07/input"

Output(s):

Successfully stored 3258984 records in: "s3://alexcp-e185/Assign07/output-p2"

Counters:

Total records written : 3258984

Total bytes written : 0

Spillable Memory Manager spill count : 0

Total bags proactively spilled: 0

Total records proactively spilled: 0

Job DAG:

job\_201304051618\_0014 -> job\_201304051618\_0015,

job\_201304051618\_0015 -> job\_201304051618\_0016,

job\_201304051618\_0016

2013-04-05 19:11:42,013 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!

2013-04-05 19:11:42,038 [Thread-4] INFO amazon.emr.metrics.MetricsSaver - Inside MetricsSaver Shutdown Hook

We then can copy the file over and verify that it is the same output as the one that was produced for Problem 1

**hadoop@ip-10-145-206-54:~/Assign07$ mkdir output-p2**

**hadoop@ip-10-145-206-54:~/Assign07$ hadoop fs -copyToLocal s3://alexcp-e185/Assign07/output-p2/\* output-p2/**

13/04/05 19:13:38 INFO metrics.MetricsSaver: MetricsSaver FsShell root:hdfs:///mnt/var/lib/hadoop/metrics/ period:60 instanceId:i-973744f8 jobflow:j-1JKGMPFYP2GTG

13/04/05 19:13:38 INFO metrics.MetricsUtil: supported product mapr-m3

13/04/05 19:13:38 INFO metrics.MetricsSaver: Disable MetricsSaver due to MapR cluster

13/04/05 19:13:38 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p2/\_SUCCESS' for reading

13/04/05 19:13:39 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p2/\_logs/history/ip-10-145-206-54.ec2.internal\_1365178749834\_job\_201304051618\_0016\_hadoop\_PigLatin%3Ap2.pig' for reading

13/04/05 19:13:39 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p2/part-r-00000' for reading

13/04/05 19:13:45 INFO metrics.MetricsSaver: Inside MetricsSaver Shutdown Hook

**hadoop@ip-10-145-206-54:~/Assign07$ head output-p2/part-r-00000**

"CITED" {("CITING")}

1 {(3964859),(4647229)}

10000 {(4539112)}

100000 {(5031388)}

1000006 {(4714284)}

1000007 {(4766693)}

1000011 {(5033339)}

1000017 {(3908629)}

1000026 {(4043055)}

1000033 {(4190903),(4975983)}

**hadoop@ip-10-145-206-54:~/Assign07$ tail output-p2/part-r-00000**

999961 {(5782495),(5738381),(4262874),(4171117),(4437639),(5878901),(4832301),(4871140),(5048788)}

999965 {(5052613)}

999968 {(3916735)}

999971 {(3965843)}

999972 {(4038129)}

999973 {(4900344),(5427610)}

999974 {(5464105),(4560073),(4728158)}

999977 {(4092587)}

999978 {(3915443)}

999983 {(5806555),(5143114),(5394715)}

**Problem 3) Create another Pig script that transform results of the previous program and produce a data set which will tell us how many times a patent has been cited. For example, the small section of cited patents list displayed above in Problem 1 will turn into something like:**

**1000026 1**

**1000033 2**

**1000043 1**

**1000044 2**

**1000045 1**

**1000046 2**

**This new list is telling us that patent 1000026 is cited once, patent 1000033 is cited twice, and so on. Store results of this program in your file system. Please record and present in your solution all the transformations that you are using and the structure of all relations (data structures) used along the way. Take a small (20 rows) section of the result and display in your solution.**

**On this problem we start out by reading the output from Problem 1. We do have to be careful to import the data properly as a list:**

**hadoop@ip-10-145-206-54:~/Assign07$ pig**

**grunt> citations = LOAD 's3://alexcp-e185/Assign07/output-p2/part-r-00000' as (cited, list:{(citing)});**

**grunt> citations\_count = FOREACH citations GENERATE $0, COUNT($1);**

**grunt> ILLUSTRATE citations\_count;**

(...)

------------------------------------------------------------------------------------

| citations | cited:bytearray | list:bag{:tuple(citing:bytearray)} |

------------------------------------------------------------------------------------

| | 1035483 | {(4827567), (5720081), (5314221)} |

------------------------------------------------------------------------------------

-------------------------------------------------------

| citations\_count | cited:bytearray | :long |

-------------------------------------------------------

| | 1035483 | 3 |

-------------------------------------------------------

It looks good. Let's create a Pig script and run it against our dataset.

**hadoop@ip-10-145-206-54:~/Assign07$ cat p3.pig**

citations = LOAD '$INPUT' as (cited, list:{(citing)});

citations\_count = FOREACH citations GENERATE $0, COUNT($1);

STORE citations\_count INTO '$OUTPUT';

**hadoop@ip-10-145-206-54:~/Assign07$ pig -p INPUT=s3://alexcp-e185/Assign07/output-p2/part-r-00000 -p OUTPUT=s3://alexcp-e185/Assign07/output-p3 file:/home/hadoop/Assign07/p3.pig**

(...)

Success!

Job Stats (time in seconds):

JobId Maps Reduces MaxMapTime MinMapTIme AvgMapTime MaxReduceTime MinReduceTime AvgReduceTime Alias Feature Outputs

job\_201304051618\_0017 6 0 47 28 38 0 0 0 citations,citations\_count MAP\_ONLY s3://alexcp-e185/Assign07/output-p3,

Input(s):

Successfully read 3258984 records from: "s3://alexcp-e185/Assign07/output-p2/part-r-00000"

Output(s):

Successfully stored 3258984 records in: "s3://alexcp-e185/Assign07/output-p3"

Counters:

Total records written : 3258984

Total bytes written : 0

Spillable Memory Manager spill count : 0

Total bags proactively spilled: 0

Total records proactively spilled: 0

Job DAG:

job\_201304051618\_0017

2013-04-05 19:32:27,967 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!

2013-04-05 19:32:27,971 [Thread-4] INFO amazon.emr.metrics.MetricsSaver - Inside MetricsSaver Shutdown Hook

Let's copy the output over and check it out:

**hadoop@ip-10-145-206-54:~/Assign07$ mkdir output-p3**

**hadoop@ip-10-145-206-54:~/Assign07$ hadoop fs -copyToLocal s3://alexcp-e185/Assign07/output-p3/\* output-p3/**

13/04/05 19:35:02 INFO metrics.MetricsSaver: MetricsSaver FsShell root:hdfs:///mnt/var/lib/hadoop/metrics/ period:60 instanceId:i-973744f8 jobflow:j-1JKGMPFYP2GTG

13/04/05 19:35:02 INFO metrics.MetricsUtil: supported product mapr-m3

13/04/05 19:35:02 INFO metrics.MetricsSaver: Disable MetricsSaver due to MapR cluster

13/04/05 19:35:02 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p3/\_SUCCESS' for reading

13/04/05 19:35:02 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p3/\_logs/history/ip-10-145-206-54.ec2.internal\_1365178749834\_job\_201304051618\_0017\_hadoop\_PigLatin%3Ap3.pig' for reading

13/04/05 19:35:03 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p3/part-m-00000' for reading

13/04/05 19:35:03 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p3/part-m-00001' for reading

13/04/05 19:35:04 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p3/part-m-00002' for reading

13/04/05 19:35:04 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p3/part-m-00003' for reading

13/04/05 19:35:05 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p3/part-m-00004' for reading

13/04/05 19:35:05 INFO s3native.NativeS3FileSystem: Opening 's3://alexcp-e185/Assign07/output-p3/part-m-00005' for reading

13/04/05 19:35:06 INFO metrics.MetricsSaver: Inside MetricsSaver Shutdown Hook

**hadoop@ip-10-145-206-54:~/Assign07$ ls -l output-p3/**

total 32152

drwxr-xr-x 3 hadoop hadoop 4096 Apr 5 19:35 \_logs

-rw-r--r-- 1 hadoop hadoop 9142120 Apr 5 19:35 part-m-00000

-rw-r--r-- 1 hadoop hadoop 5516555 Apr 5 19:35 part-m-00001

-rw-r--r-- 1 hadoop hadoop 4147969 Apr 5 19:35 part-m-00002

-rw-r--r-- 1 hadoop hadoop 3960443 Apr 5 19:35 part-m-00003

-rw-r--r-- 1 hadoop hadoop 4132888 Apr 5 19:35 part-m-00004

-rw-r--r-- 1 hadoop hadoop 5958638 Apr 5 19:35 part-m-00005

-rw-r--r-- 1 hadoop hadoop 0 Apr 5 19:35 \_SUCCESS

**hadoop@ip-10-145-206-54:~/Assign07$ head -20 output-p3/part-m-00000**

"CITED" 1

1 2

10000 1

100000 1

1000006 1

1000007 1

1000011 1

1000017 1

1000026 1

1000033 2

1000043 1

1000044 2

1000045 1

1000046 2

1000049 1

1000051 1

1000054 1

1000065 1

1000067 3

1000070 2

The result seems good as well.

**Problem 4) Create a Pig script that will read the result of the previous problem and create a “histogram” of number of citations. We want to know how many patents are not cited at all, how many patents are cited once, cited twice and so on. Create one ordered list where the number of patents with the largest number of citations is on the top and another where the number of patents with 0 or 1 (the smallest number of) citations is on the top of the list. Store both results on your file system.**

**Could you write one or more SQL commands that would accomplish the same objectives. For us, make a small extract from the top of both files.**

For this final problem, we start out by reading the output from our previous result

**hadoop@ip-10-145-206-54:~/Assign07$ pig**

**grunt> cit\_count = LOAD 's3://alexcp-e185/Assign07/output-p3' as (cited, count:int);**

**grunt> ILLUSTRATE cit\_count**

---------------------------------------------------

| cit\_count | cited:bytearray | count:int |

---------------------------------------------------

| | 1012106 | 2 |

---------------------------------------------------

We then need to group by the citation counts we have in order to figure out how many patents have an specific number of citations:

**grunt> cit\_count\_grp = GROUP cit\_count BY count;**

**grunt> ILLUSTRATE cit\_count\_grp;**

-----------------------------------------------------

| cit\_count | cited:bytearray | count:int |

-----------------------------------------------------

| | 1037469 | 15 |

| | 1006722 | 15 |

-----------------------------------------------------

---------------------------------------------------------------------------------------------------

| cit\_count\_grp | group:int | cit\_count:bag{:tuple(cited:bytearray,count:int)} |

---------------------------------------------------------------------------------------------------

| | 15 | {(1037469, 15), (1006722, 15)} |

---------------------------------------------------------------------------------------------------

And we then can count how many patents we have in each citation count:

**grunt> cit\_hist = FOREACH cit\_count\_grp GENERATE $0, COUNT($1);**

**grunt> ILLUSTRATE cit\_hist;**

-----------------------------------------------------

| cit\_count | cited:bytearray | count:int |

-----------------------------------------------------

| | 1037469 | 15 |

| | 1006722 | 15 |

-----------------------------------------------------

---------------------------------------------------------------------------------------------------

| cit\_count\_grp | group:int | cit\_count:bag{:tuple(cited:bytearray,count:int)} |

---------------------------------------------------------------------------------------------------

| | 15 | {(1037469, 15), (1006722, 15)} |

---------------------------------------------------------------------------------------------------

------------------------------------------

| cit\_hist | group:int | :long |

------------------------------------------

| | 15 | 2 |

------------------------------------------

We seem to be good to go. We now create a Pig Script that has these commands and outputs to an descending and an ascending list using the ORDER command. We also hard code the input and output paths for simplicity.

**hadoop@ip-10-145-206-54:~/Assign07$ cat p4.pig**

cit\_count = LOAD 's3://alexcp-e185/Assign07/output-p3' as (cited, count:int);

cit\_count\_grp = GROUP cit\_count BY count;

cit\_hist = FOREACH cit\_count\_grp GENERATE $0, COUNT($1);

cit\_hist\_asc = ORDER cit\_hist BY $1 ASC;

cit\_hist\_desc = ORDER cit\_hist BY $1 DESC;

STORE cit\_hist\_asc INTO 's3://alexcp-e185/Assign07/output-p4-asc';

STORE cit\_hist\_desc INTO 's3://alexcp-e185/Assign07/output-p4-desc';

And we execute the Pig script.

**hadoop@ip-10-145-206-54:~/Assign07$ pig file:/home/hadoop/Assign07/p4.pig**

(...)

Success!

Job Stats (time in seconds):

JobId Maps Reduces MaxMapTime MinMapTIme AvgMapTime MaxReduceTime MinReduceTime AvgReduceTime Alias Feature Outputs

job\_201304051618\_0018 1 1 25 25 25 5 5 5 cit\_count,cit\_count\_grp,cit\_hist GROUP\_BY,MULTI\_QUERY,COMBINER

job\_201304051618\_0019 1 1 3 3 3 5 5 5 cit\_hist\_asc SAMPLER

job\_201304051618\_0020 1 1 2 2 2 5 5 5 cit\_hist\_desc SAMPLER

job\_201304051618\_0021 1 1 4 4 4 7 7 7 cit\_hist\_desc ORDER\_BY s3://alexcp-e185/Assign07/output-p4-desc,

job\_201304051618\_0022 1 1 3 3 3 6 6 6 cit\_hist\_asc ORDER\_BY s3://alexcp-e185/Assign07/output-p4-asc,

Input(s):

Successfully read 3258984 records (34156 bytes) from: "s3://alexcp-e185/Assign07/output-p3"

Output(s):

Successfully stored 258 records in: "s3://alexcp-e185/Assign07/output-p4-asc"

Successfully stored 258 records in: "s3://alexcp-e185/Assign07/output-p4-desc"

Counters:

Total records written : 516

Total bytes written : 0

Spillable Memory Manager spill count : 0

Total bags proactively spilled: 0

Total records proactively spilled: 0

Job DAG:

job\_201304051618\_0018 -> job\_201304051618\_0019,job\_201304051618\_0020,

job\_201304051618\_0019 -> job\_201304051618\_0022,

job\_201304051618\_0022

job\_201304051618\_0020 -> job\_201304051618\_0021,

job\_201304051618\_0021

2013-04-05 20:06:14,098 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!

2013-04-05 20:06:14,142 [Thread-4] INFO amazon.emr.metrics.MetricsSaver - Inside MetricsSaver Shutdown Hook

Finally, we check the output results:

**hadoop@ip-10-145-206-54:~/Assign07$ mkdir output-p4-asc**

**hadoop@ip-10-145-206-54:~/Assign07$ hadoop fs -copyToLocal hadoop@ip-10-145-206-54:~/Assign07$ mkdir output-p4-desc**

**hadoop@ip-10-145-206-54:~/Assign07$ hadoop fs -copyToLocal s3://alexcp-e185/Assign07/output-p4-desc/\* output-p4-desc/**

**hadoop@ip-10-145-206-54:~/Assign07$ head output-p4-asc/part-r-00000**

215 1

631 1

210 1

208 1

779 1

206 1

613 1

605 1

411 1

200 1

**hadoop@ip-10-145-206-54:~/Assign07$ head output-p4-desc/part-r-00000**

1 921128

2 552246

3 380319

4 278438

5 210814

6 163149

7 127941

8 102155

9 82126

10 66634

As far as equivalent SQL statements would go, I would not know how to do it without creating a temporary table. The pseudo-SQL below should do the trick for the descending output, and for the other one we would just need to replace the word DESC on the last line by ASC.

CREATE TABLE temp\_count (cited varchar(255), cite\_count int);

INSERT INTO temp\_count

(SELECT cited, count(\*) as cite\_count FROM citations GROUP BY citing);

SELECT cite\_count, count(\*) as cite\_hist

FROM temp\_count

GROUP BY cite\_count

ORDER BY cite\_hist DESC;

DROP TABLE temp\_count;