DATSCIW261 ASSIGNMENT 3

MIDS UC Berkeley, Machine Learning at Scale

AUTHOR : Rajesh Thallam
EMAIL : rajesh.thallam@ischool.berkeley.edu
WEEK : 3
DATE : 22-Sep-15

HW3.0

a. What is a merge sort? Where is it used in Hadoop?

A merge-sort is a comparison based sorting algorithm with following steps

- **input list of length n is divided into n sublists with each sublist containing one element.**
- **repeatedly merge the sorted sublists into new sublists until there is only one sublist.**

The merge-sort algorithm is used during Hadoop shuffle phase specifically on the reducer side of the shuffling. Merge-sort is used to sort the outputs from the mapper outputs and later to be merged.

b. How is a combiner function in the context of Hadoop? Give an example where it can be used and justify why it should be used in the context of this problem.

A combiner is an optional phase used in Hadoop Reduce specifically on the mapper-side to reduce the network traffic between mapper and reducer. Combiner essentially acts as a reducer aggregating local values from the mapper outputs with the same key before sending to the shuffle phase.

Word count would be a simple example to explain use of combiner. With combiner in place, every document instead of emitting (word, 1) for every word from the mapper, combiner can locally aggregate the count for each word and emit (word, count_of_words) for each distinct word in the document.

This largely reduces the network traffic as we now have fewer number of words to transfer from the mapper to reducer. Due to local aggregation it also reduces the number of disk spills.

c. What is the Hadoop shuffle?

Hadoop shuffle is the core process of transferring data from mapper to reducers. It consists of three steps/phases: partition, sort and combine.

- **On the mapper side**
    - **Map outputs are buffered in memory in a circular buffer**
    - **When buffer reaches threshold, sorted contents are spilled to disk.**
    - **Spills are merged into a single, partition file (sorted within each partition).**
- **On the reducer side**
    - **All map outputs are copied over to the reducer machines**
    - **A multi-pass merge, or merge-sort happens in memory and on disk**
    - **Finally, the last merge pass goes directly into the reducer**

d. What is the Apriori algorithm? Describe an example use in your domain of expertise. Define confidence and lift.

Apriori algorithm is a data mining algorithm to find out the association among item sets using support and confidence. These two inputs help to discriminate the frequent and infrequent item sets. The algorithm is based on the principle that if an item does not fulfill minimum support constraint or not frequent then its descendants are also not frequent so this item is removed from the basket because this item does not contribute in the construction of association rules.

Appliation of apriori in my domain In the healthcare domain, this could be used for disease classification for a patient based on set of rules such as diagnosis or procedure codes performed, age, gender and BMI. For example, if a person has plasma-glucose levels high and age is between [40, 60] with BMI as severely obese then we can classify patient having type-II diabetes with confidence of X%. I may have given a crude example but the idea is to classify disease condition in a patient. Similar condition could be used for drug abuse detection similar to credit card fraud detection by finding infrequent item sets.

Confidence is defined as the measure of certainty associated with each associated pattern given the occurrence of the antecedent. For example, beer might appear in 5 transactions; 3 of the 5 might also include diaper. The rule confidence would be: beer implies diapers with 60% confidence

Lift measures how many times more often item sets occur together than expected if they where statistically independent.

```
Lift = Support/(Support(X) * Support(Y))
```

Preparation for HW3_*

In [3]:
```
# stop hadoop
!ssh hduser@rtubuntu /usr/local/hadoop/sbin/stop-yarn.sh
!ssh hduser@rtubuntu /usr/local/hadoop/sbin/stop-dfs.sh
```

```
stopping yarn daemons
no resourcemanager to stop
localhost: no nodemanager to stop
no proxyserver to stop
15/09/19 17:59:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
Stopping namenodes on [localhost]
localhost: no namenode to stop
localhost: no datanode to stop
Stopping secondary namenodes [0.0.0.0]
0.0.0.0: no secondarynamenode to stop
15/09/19 17:59:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
```

In [4]:
```
# start hadoop
!ssh hduser@rtubuntu /usr/local/hadoop/sbin/start-yarn.sh
!ssh hduser@rtubuntu /usr/local/hadoop/sbin/start-dfs.sh
```

```
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-hduser-resourcemanager-rtubuntu.out
localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-hduser-nodemanager-rtubuntu.out
15/09/19 17:59:41 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hduser-namenode-rtubuntu.out
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hduser-datanode-rtubuntu.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-hduser-secondarynamenode-rtubuntu.out
15/09/19 18:00:01 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
```

In [6]:
```
# create necessary directories
!hdfs dfs -mkdir /hw3
```

```
15/09/19 18:00:58 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
```

HW3.1

Suppose we want to recommend new products to the customer based on the products they have already browsed on the online website. Do some exploratory data analysis of this dataset. Report your findings such as number of unique products; largest basket, etc. using Hadoop Map-Reduce.

Data Exploration

As part of data exploration, following analysis will be conducted for this problem

- Number of transactions
- Number of unique products
- Largest basket i.e. the transaction with maximum products
- Average products in a transaction

```
In [8]: !head ProductPurchaseData.txt
```

```
FRO11987 ELE17451 ELE89019 SNA90258 GRO99222
GRO99222 GRO12298 FRO12685 ELE91550 SNA11465 ELE26917 ELE52966 FRO90334 SNA30755 ELE17451 FRO84225 SNA80192
ELE17451 GRO73461 DAI22896 SNA99873 FRO86643
ELE17451 ELE37798 FRO86643 GRO56989 ELE23393 SNA11465
ELE17451 SNA69641 FRO86643 FRO78087 SNA11465 GRO39357 ELE28573 ELE11375 DAI54444
ELE17451 GRO73461 DAI22896 SNA99873 FRO18919 DAI50921 SNA80192 GRO75578
ELE17451 ELE59935 FRO18919 ELE23393 SNA80192 SNA85662 SNA91554 DAI22177
ELE17451 SNA69641 FRO18919 SNA90258 ELE28573 ELE11375 DAI14125 FRO78087
ELE17451 GRO73461 DAI22896 SNA80192 SNA85662 SNA90258 DAI46755 FRO81176 ELE66810 DAI49199 DAI91535 GRO94758 ELE94711 DAI22
177
ELE17451 SNA69641 DAI91535 GRO94758 GRO99222 FRO76833 FRO81176 SNA80192 DAI54690 ELE37798 GRO56989
```

Mapper

This mapper emits product in each transaction with transaction id and number of products in that transaction. Mapper sends redundant for each line i.e. transaction id (or row number) and number of products

```
In [26]: %%writefile mapper.py
#!/usr/bin/env python
import sys

rownum = 0
for transactions in sys.stdin:
    rownum += 1
    products = transactions.strip().split()
    for product in products:
        print '{} {} {}'.format(rownum, len(products), product)
```

```
Overwriting mapper.py
```

Reducer

Reducer reads each line from mapper output and maintains list of unique products and transactions with number of produts. At the end of the reducer stage, it prints data exploration summary.

```
In [33]: %%writefile reducer.py
#!/usr/bin/env python
import sys

products = {}
transactions = {}
no_of_transactions = 0
no_of_products = 0

for lines in sys.stdin:
    no_of_products += 1

    line = lines.strip().split()
    transaction_id = line[0]
    total_products = int(line[1])
    product = line[2]

    products[product] = products.get(product, 0) + 1
    transactions[transaction_id] = transactions.get(transaction_id, total_products)

print "-" * 60
print 'Data Exploration Summary'
print "-" * 60
print "{0: <40} | {1}".format("MEASURE", "VALUE")
print "{0: <40}-+-{1}".format("-" * 40, "-" * 17)

print '{0: <40} | {1:d}'.format('Number of transactions', len(transactions))
print '{0: <40} | {1:d}'.format('Total products', sum(transactions.values()))
print '{0: <40} | {1:d}'.format('Unique products', len(products))
print '{0: <40} | Row# {1} = {2}'.format('Transaction with maximum products', max(transactions, key=lambda key: transactio
ns[key]), max(transactions.values()))
print '{0: <40} | Row# {1} = {2}'.format('Transaction with minimum products', min(transactions, key=lambda key: transactio
ns[key]), min(transactions.values()))
print '{0: <40} | {1:0.2f}'.format('Average products per transaction', sum(transactions.values())/len(transactions))
print '{0: <40} | Product {1} browsed {2} times'.format('Most popular product (most browsed)', max(products, key=lambda ke
y: products[key]), max(products.values()))

print "-" * 60
```

```
Overwriting reducer.py
```

Preparing to run the job

In [16]:

```
# Use chmod for permissions
!chmod a+x mapper.py
!chmod a+x reducer.py
```

In [17]:

```
!hdfs dfs -mkdir /hw3/hw3_1
!hdfs dfs -mkdir /hw3/hw3_1/src
!hdfs dfs -put ./ProductPurchaseData.txt /hw3/hw3_1/src
```

```
15/09/20 10:13:50 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
15/09/20 10:13:53 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
15/09/20 10:13:55 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
```

Driver Function

Driver function calls the hadoop streaming job after purging previously generated target files (to avoid the `'File Already Exists'` error). Few points to notice

- number of mappers is set to 10
- number of reducers is set to 1
- output data exploration summary

In [34]:

```
# HW 3.1: exploratory data analysis of the data set
def hw3_1():
    # cleanup target directory
    !hdfs dfs -rm -R /hw3/hw3_1/tgt

    !echo "sample input data"
    !hdfs dfs -cat /hw3/hw3_1/src/ProductPurchaseData.txt | head

    # run map reduce job
    !hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.6.0.jar \
    -Dmapreduce.job.maps=10 \
    -Dmapreduce.job.reduces=1 \
    -files mapper.py,reducer.py \
    -mapper mapper.py \
    -reducer reducer.py \
    -input /hw3/hw3_1/src/ProductPurchaseData.txt \
    -output /hw3/hw3_1/tgt

    print "\n"
    !echo "partial output data"
    !hdfs dfs -cat /hw3/hw3_1/tgt/part-00000

hw3_1()
```

```
15/09/22 04:20:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
15/09/22 04:20:25 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 0 minutes, Emptier interva
l = 0 minutes.
Deleted /hw3/hw3_1/tgt
sample input data
15/09/22 04:20:27 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
FRO11987 ELE17451 ELE89019 SNA90258 GRO99222
GRO99222 GRO12298 FRO12685 ELE91550 SNA11465 ELE26917 ELE52966 FRO90334 SNA30755 ELE17451 FRO84225 SNA80192
ELE17451 GRO73461 DAI22896 SNA99873 FRO86643
ELE17451 ELE37798 FRO86643 GRO56989 ELE23393 SNA11465
ELE17451 SNA69641 FRO86643 FRO78087 SNA11465 GRO39357 ELE28573 ELE11375 DAI54444
ELE17451 GRO73461 DAI22896 SNA99873 FRO18919 DAI50921 SNA80192 GRO75578
ELE17451 ELE59935 FRO18919 ELE23393 SNA80192 SNA85662 SNA91554 DAI22177
ELE17451 SNA69641 FRO18919 SNA90258 ELE28573 ELE11375 DAI14125 FRO78087
ELE17451 GRO73461 DAI22896 SNA80192 SNA85662 SNA90258 DAI46755 FRO81176 ELE66810 DAI49199 DAI91535 GRO94758 ELE94711 DAI22
177
ELE17451 SNA69641 DAI91535 GRO94758 GRO99222 FRO76833 FRO81176 SNA80192 DAI54690 ELE37798 GRO56989
cat: Unable to write to output stream.
15/09/22 04:20:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
15/09/22 04:20:30 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.session-id
15/09/22 04:20:30 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
15/09/22 04:20:30 INFO jvm.JvmMetrics: Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already ini
tialized
15/09/22 04:20:31 INFO mapred.FileInputFormat: Total input paths to process : 1
15/09/22 04:20:31 INFO mapreduce.JobSubmitter: number of splits:1
15/09/22 04:20:31 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1511436974_0001
15/09/22 04:20:31 INFO mapred.LocalDistributedCacheManager: Localized file:/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARN
ING-AT-SCALE/week3/hw3/mapper.py as file:/app/hadoop/tmp/mapred/local/1442920831535/mapper.py
15/09/22 04:20:31 INFO mapred.LocalDistributedCacheManager: Localized file:/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARN
ING-AT-SCALE/week3/hw3/reducer.py as file:/app/hadoop/tmp/mapred/local/1442920831536/reducer.py
15/09/22 04:20:31 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
15/09/22 04:20:31 INFO mapreduce.Job: Running job: job_local1511436974_0001
15/09/22 04:20:31 INFO mapred.LocalJobRunner: OutputCommitter set in config null
```

```
15/09/22 04:20:31 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapred.FileOutputCommitter
15/09/22 04:20:32 INFO mapred.LocalJobRunner: Waiting for map tasks
15/09/22 04:20:32 INFO mapred.LocalJobRunner: Starting task: attempt_local1511436974_0001_m_000000_0
15/09/22 04:20:32 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
15/09/22 04:20:32 INFO mapred.MapTask: Processing split: hdfs://localhost:54310/hw3/hw3_1/src/ProductPurchaseData.txt:0+34
58517
15/09/22 04:20:32 INFO mapred.MapTask: numReduceTasks: 1
15/09/22 04:20:32 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
15/09/22 04:20:32 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
15/09/22 04:20:32 INFO mapred.MapTask: soft limit at 83886080
15/09/22 04:20:32 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
15/09/22 04:20:32 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
15/09/22 04:20:32 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
15/09/22 04:20:32 INFO streaming.PipeMapRed: PipeMapRed exec [/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARNING-AT-SCALE/
week3/hw3/./mapper.py]
15/09/22 04:20:32 INFO Configuration.deprecation: mapred.tip.id is deprecated. Instead, use mapreduce.task.id
15/09/22 04:20:32 INFO Configuration.deprecation: mapred.local.dir is deprecated. Instead, use mapreduce.cluster.local.dir
15/09/22 04:20:32 INFO Configuration.deprecation: map.input.file is deprecated. Instead, use mapreduce.map.input.file
15/09/22 04:20:32 INFO Configuration.deprecation: mapred.skip.on is deprecated. Instead, use mapreduce.job.skiprecords
15/09/22 04:20:32 INFO Configuration.deprecation: map.input.length is deprecated. Instead, use mapreduce.map.input.length
15/09/22 04:20:32 INFO Configuration.deprecation: mapred.work.output.dir is deprecated. Instead, use mapreduce.task.output
.dir
15/09/22 04:20:32 INFO Configuration.deprecation: map.input.start is deprecated. Instead, use mapreduce.map.input.start
15/09/22 04:20:32 INFO Configuration.deprecation: mapred.job.id is deprecated. Instead, use mapreduce.job.id
15/09/22 04:20:32 INFO Configuration.deprecation: user.name is deprecated. Instead, use mapreduce.job.user.name
15/09/22 04:20:32 INFO Configuration.deprecation: mapred.task.is.map is deprecated. Instead, use mapreduce.task.ismap
15/09/22 04:20:32 INFO Configuration.deprecation: mapred.task.id is deprecated. Instead, use mapreduce.task.attempt.id
15/09/22 04:20:32 INFO Configuration.deprecation: mapred.task.partition is deprecated. Instead, use mapreduce.task.partiti
on
15/09/22 04:20:32 INFO streaming.PipeMapRed: R/W/S=1/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:32 INFO streaming.PipeMapRed: R/W/S=10/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:32 INFO streaming.PipeMapRed: R/W/S=100/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:32 INFO streaming.PipeMapRed: R/W/S=1000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:32 INFO streaming.PipeMapRed: Records R/W=1216/1
15/09/22 04:20:32 INFO mapreduce.Job: Job job_local1511436974_0001 running in uber mode : false
15/09/22 04:20:32 INFO mapreduce.Job:  map 0% reduce 0%
15/09/22 04:20:33 INFO streaming.PipeMapRed: R/W/S=10000/121229/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:33 INFO streaming.PipeMapRed: MRErrorThread done
15/09/22 04:20:33 INFO streaming.PipeMapRed: mapRedFinished
15/09/22 04:20:33 INFO mapred.LocalJobRunner:
15/09/22 04:20:33 INFO mapred.MapTask: Starting flush of map output
15/09/22 04:20:33 INFO mapred.MapTask: Spilling map output
15/09/22 04:20:33 INFO mapred.MapTask: bufstart = 0; bufend = 7012667; bufvoid = 104857600
15/09/22 04:20:33 INFO mapred.MapTask: kvstart = 26214396(104857584); kvend = 24691104(98764416); length = 1523293/6553600
15/09/22 04:20:35 INFO mapred.MapTask: Finished spill 0
15/09/22 04:20:35 INFO mapred.Task: Task:attempt_local1511436974_0001_m_000000_0 is done. And is in the process of committ
ing
15/09/22 04:20:35 INFO mapred.LocalJobRunner: Records R/W=1216/1
15/09/22 04:20:35 INFO mapred.Task: Task 'attempt_local1511436974_0001_m_000000_0' done.
15/09/22 04:20:35 INFO mapred.LocalJobRunner: Finishing task: attempt_local1511436974_0001_m_000000_0
15/09/22 04:20:35 INFO mapred.LocalJobRunner: map task executor complete.
15/09/22 04:20:35 INFO mapred.LocalJobRunner: Waiting for reduce tasks
15/09/22 04:20:35 INFO mapred.LocalJobRunner: Starting task: attempt_local1511436974_0001_r_000000_0
15/09/22 04:20:35 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
15/09/22 04:20:35 INFO mapred.ReduceTask: Using ShuffleConsumerPlugin: org.apache.hadoop.mapreduce.task.reduce.Shuffle@43d
6de10
15/09/22 04:20:35 INFO reduce.MergeManagerImpl: MergerManager: memoryLimit=363285696, maxSingleShuffleLimit=90821424, merg
eThreshold=239768576, ioSortFactor=10, memToMemMergeOutputsThreshold=10
15/09/22 04:20:35 INFO reduce.EventFetcher: attempt_local1511436974_0001_r_000000_0 Thread started: EventFetcher for fetch
ing Map Completion Events
15/09/22 04:20:35 INFO reduce.LocalFetcher: localfetcher#1 about to shuffle output of map attempt_local1511436974_0001_m_0
00000_0 decomp: 7774317 len: 7774321 to MEMORY
15/09/22 04:20:35 INFO reduce.InMemoryMapOutput: Read 7774317 bytes from map-output for attempt_local1511436974_0001_m_000
000_0
15/09/22 04:20:35 INFO reduce.MergeManagerImpl: closeInMemoryFile -> map-output of size: 7774317, inMemoryMapOutputs.size(
) -> 1, commitMemory -> 0, usedMemory ->7774317
15/09/22 04:20:35 INFO reduce.EventFetcher: EventFetcher is interrupted.. Returning
15/09/22 04:20:35 INFO mapred.LocalJobRunner: 1 / 1 copied.
15/09/22 04:20:35 INFO reduce.MergeManagerImpl: finalMerge called with 1 in-memory map-outputs and 0 on-disk map-outputs
15/09/22 04:20:35 INFO mapred.Merger: Merging 1 sorted segments
15/09/22 04:20:35 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 7774302 bytes
15/09/22 04:20:35 INFO reduce.MergeManagerImpl: Merged 1 segments, 7774317 bytes to disk to satisfy reduce memory limit
15/09/22 04:20:35 INFO reduce.MergeManagerImpl: Merging 1 files, 7774321 bytes from disk
15/09/22 04:20:35 INFO reduce.MergeManagerImpl: Merging 0 segments, 0 bytes from memory into reduce
15/09/22 04:20:35 INFO mapred.Merger: Merging 1 sorted segments
15/09/22 04:20:35 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 7774302 bytes
15/09/22 04:20:35 INFO mapred.LocalJobRunner: 1 / 1 copied.
15/09/22 04:20:35 INFO streaming.PipeMapRed: PipeMapRed exec [/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARNING-AT-SCALE/
week3/hw3/./reducer.py]
15/09/22 04:20:35 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
15/09/22 04:20:35 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
15/09/22 04:20:35 INFO streaming.PipeMapRed: R/W/S=1/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:35 INFO streaming.PipeMapRed: R/W/S=10/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:35 INFO streaming.PipeMapRed: R/W/S=100/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:35 INFO streaming.PipeMapRed: R/W/S=1000/0/0 in:NA [rec/s] out:NA [rec/s]
```

```
15/09/22 04:20:35 INFO streaming.PipeMapRed: R/W/S=10000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:35 INFO mapreduce.Job:  map 100% reduce 0%
15/09/22 04:20:36 INFO streaming.PipeMapRed: R/W/S=100000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:20:37 INFO streaming.PipeMapRed: R/W/S=200000/0/0 in:200000=200000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:20:37 INFO streaming.PipeMapRed: R/W/S=300000/0/0 in:300000=300000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:20:37 INFO streaming.PipeMapRed: Records R/W=380824/1
15/09/22 04:20:37 INFO streaming.PipeMapRed: MRErrorThread done
15/09/22 04:20:37 INFO streaming.PipeMapRed: mapRedFinished
15/09/22 04:20:37 INFO mapred.Task: Task:attempt_local1511436974_0001_r_000000_0 is done. And is in the process of committ
ing
15/09/22 04:20:37 INFO mapred.LocalJobRunner: 1 / 1 copied.
15/09/22 04:20:37 INFO mapred.Task: Task attempt_local1511436974_0001_r_000000_0 is allowed to commit now
15/09/22 04:20:37 INFO output.FileOutputCommitter: Saved output of task 'attempt_local1511436974_0001_r_000000_0' to hdfs:
//localhost:54310/hw3/hw3_1/tgt/_temporary/0/task_local1511436974_0001_r_000000
15/09/22 04:20:37 INFO mapred.LocalJobRunner: Records R/W=380824/1 > reduce
15/09/22 04:20:38 INFO mapred.Task: Task 'attempt_local1511436974_0001_r_000000_0' done.
15/09/22 04:20:38 INFO mapred.LocalJobRunner: Finishing task: attempt_local1511436974_0001_r_000000_0
15/09/22 04:20:38 INFO mapred.LocalJobRunner: reduce task executor complete.
15/09/22 04:20:38 INFO mapreduce.Job:  map 100% reduce 100%
15/09/22 04:20:38 INFO mapreduce.Job: Job job_local1511436974_0001 completed successfully
15/09/22 04:20:39 INFO mapreduce.Job: Counters: 38
        File System Counters
                FILE: Number of bytes read=15762262
                FILE: Number of bytes written=24048931
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=6917034
                HDFS: Number of bytes written=723
                HDFS: Number of read operations=13
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
        Map-Reduce Framework
                Map input records=31101
                Map output records=380824
                Map output bytes=7012667
                Map output materialized bytes=7774321
                Input split bytes=112
                Combine input records=0
                Combine output records=0
                Reduce input groups=380821
                Reduce shuffle bytes=7774321
                Reduce input records=380824
                Reduce output records=13
                Spilled Records=761648
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=97
                CPU time spent (ms)=0
                Physical memory (bytes) snapshot=0
                Virtual memory (bytes) snapshot=0
                Total committed heap usage (bytes)=335683584
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=3458517
        File Output Format Counters
                Bytes Written=723
15/09/22 04:20:39 INFO streaming.StreamJob: Output directory: /hw3/hw3_1/tgt
```

```
partial output data
15/09/22 04:20:40 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
------------------------------------------------------------
Data Exploration Summary
------------------------------------------------------------
MEASURE                                 | VALUE
----------------------------------------+-------------------
Number of transactions                  | 31101
Total products                          | 380824
Unique products                         | 12592
Transaction with maximum products       | Row# 7034 = 37
Transaction with minimum products       | Row# 26069 = 2
Average products per transaction        | 12.00
Most popular product (most browsed)     | Product DAI62779 browsed 6667 times
------------------------------------------------------------
```

HW3.2

List the top 5 rules with corresponding confidence scores in decreasing order of confidence score for frequent (100>count) itemsets of size 2. A rule is of the form: (item1) ⇒ item2. Fix the ordering of the rule lexicographically (left to right), and break ties in confidence (between rules, if any exist) by taking the first ones in lexicographically increasing order. Use Hadoop MapReduce to complete this part of the assignment; use a single mapper and single reducer; use a combiner if you think it will help and justify.

Assumptions

1. Based on the disussion in the LMS and requirement, algorithm in the solution considers ordering of the rules lexicographically whih would ignore half of the possible rules. Solution reports the rule a->b but not the rule b->a and if a and b occur at different frequencies, the rule confidences are unequal. However to validate, I have enabled flag on the reducer to turn the lexicographial ordering on or off.
2. The algorithm uses pairs method with combiner to enable distributed processing and local aggregation before sending output to reduer. The combiner task is totally optional and the output remains same irrespective of combiner. The signature of combiner and reduer are same even though outputs are different

Mapper

- The mapper reads the input file and emits item sets of k = 1 and k = 2 in the format
  - for k = 1, emits ((item1), count)
  - for k = 2, emits ((item1, item2), count)
- Mapper removes duplicate products in a single transaction and sorts alphabetically before emitting the output.

In [35]:

```
%%writefile mapper.py
#!/usr/bin/python
import traceback
import itertools
import sys
import re

try:
    item_sets = {}
    item_counts = {}

    for transaction in sys.stdin:
        items = sorted(list(set(transaction.split())))

        for item in items:
            item_counts[item] = item_counts.get(item, 0) + 1

        # using pairs and k = 2
        for pair in itertools.combinations(items, 2):
            key = ','.join(pair)
            item_sets[key] = item_sets.get(key, 0) + 1

    for k, v in item_sets.iteritems():
        print "{0}\t{1}".format(k, v)

    for k, v in item_counts.iteritems():
        print "{0}\t{1}".format(k, v)

except Exception:
    traceback.print_exc()
```

```
Overwriting mapper.py
```

Reducer

- The reducer reads the mapper or combiner output emitting frequent item sets in the format below
  - for k = 1, emits ((item1), count)
  - for k = 2, emits ((item1, item2), count)
- Reducer combines mapper or combiner task outputs to form a final frequent item set i.e. local aggregation and reports frequent item set sizes after pruning based on support threshold of s = 100
- Reducer takes lexicographical requirement into account to break the ties (and it can be enabled on or off)
- Association rules are reported in the form a => b, c where a and b represent frequent item set pair and c represents confidene score. Only top-5 rules with confidence scores are reported

```
In [40]: %%writefile reducer.py
#!/usr/bin/python
import traceback
import itertools
import sys
import ast
import re

try:
    # define variables
    SUPPORT_THRESHOLD = 100
    LEXIC_ORDERING = 1
    rules = []
    final_item_set_0 = {}
    final_item_set_1 = {}
    supps = []

    # read each map output
    for line in sys.stdin:
        key, value = line.strip().split('\t')
        value = int(value)

        k = len(key.split(','))
        if k == 1:
            final_item_set_0[key] = final_item_set_0.get(key, 0) + value
        if k == 2:
            final_item_set_1[key] = final_item_set_1.get(key, 0) + value

    supps.append({k:v for k,v in final_item_set_0.iteritems() if v >= SUPPORT_THRESHOLD})
    print "|C{}| = {}".format(1, str(len(final_item_set_0)))
    print "|L{}| = {}".format(1, str(len(supps[0])))

    supps.append({k:v for k,v in final_item_set_1.iteritems() if v >= SUPPORT_THRESHOLD})
    print "|C{}| = {}".format(2, str(len(final_item_set_1)))
    print "|L{}| = {}".format(2, str(len(supps[1])))

    if LEXIC_ORDERING == 0:
        remove_chars = ['(', ')', ',', '\'']
        rx = '[' + re.escape(''.join(remove_chars)) + ']'

        for key, value in supps[1].iteritems():
            key = key.split(',')
            for a in itertools.combinations(key, 1):
                b = tuple([w for w in key if w not in a])
                conf = float(value) / float(supps[0][','.join(a)])

                a = re.sub(rx, '', str(a))
                b = re.sub(rx, '', str(b))
                rules.append((a, b, conf))
    else:
        for key, value in supps[1].iteritems():
            key = key.split(',')
            a = key[0]
            b = key[1]
            conf = float(value) / float(supps[0][a])
            rules.append((a, b, conf))

    rules = sorted(rules, key=lambda x: (x[0], x[1]))
    rules = sorted(rules, key=lambda x: (x[2]), reverse=True)

    print "-" * 60
    print 'Top-5 association rules with confidence scores'
    print "-" * 60

    for rule in rules[:5]:
        print ("{} => {}, conf = {:.4f}").format(rule[0], rule[1], rule[2])

except Exception:
    traceback.print_exc()
```

```
Overwriting reducer.py
```

Combiner

- **The combiner reads the mapper output emitting frequent item sets in the format below**
  - **for k = 1, emits ((item1), count)**
  - **for k = 2, emits ((item1, item2), count)**
- **Combiner is in place to locally aggregate same item set pairs to identify frequent item sets. Pruning is performed at the reduer after collecting frequent item sets from all the combiner stages**
- **Combiner is an optional task to improve performance by reducing network transfer and the map reduce code should report the same output with or without combiner**

In [38]:

```
%%writefile combiner.py
#!/usr/bin/python
import traceback
import itertools
import sys
import ast

try:
    # define variables
    final_item_set_0 = {}
    final_item_set_1 = {}
    supps = []

    # read each map output
    for line in sys.stdin:
        key, value = line.strip().split('\t')
        value = int(value)

        k = len(key.split(','))
        if k == 1:
            final_item_set_0[key] = final_item_set_0.get(key, 0) + value
        if k == 2:
            final_item_set_1[key] = final_item_set_1.get(key, 0) + value

    for k, v in final_item_set_0.iteritems():
        print "{0}\t{1}".format(k, v)

    for k, v in final_item_set_1.iteritems():
        print "{0}\t{1}".format(k, v)

except Exception:
    traceback.print_exc()
```

```
Overwriting combiner.py
```

Preparing to run the job

In [105]:

```
# move source file to hdfs
!hdfs dfs -mkdir /hw3/hw3_2
```

```
15/09/20 18:59:52 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
```

Driver Function

In [47]:

```
# HW 3.2  Report top-5 rules with confidence scores in
#         the browsing produt data using mapreduce code

import time

def hw3_2():
    start_time = time.time()

    # cleanup target directory
    !hdfs dfs -rm -R /hw3/hw3_2/tgt

    # run map reduce job
    !hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.6.0.jar \
    -Dmapreduce.job.maps=10 \
    -Dmapreduce.job.combines=10 \
    -Dmapreduce.job.reduces=1 \
    -files mapper.py,reducer.py,combiner.py \
    -mapper mapper.py \
    -reducer reducer.py \
    -combiner combiner.py \
    -input /hw3/hw3_1/src/ProductPurchaseData.txt \
    -output /hw3/hw3_2/tgt

    end_time = time.time()

    print "\nOUTPUT"
    # display count on the screen
    print "output from mapper/reducer to determine the number of occurrences of word assistance"
    !hdfs dfs -cat /hw3/hw3_2/tgt/part-00000

    print "Time taken to find association rules = {:.2f} second.".format(end_time - start_time)

hw3_2()
```

```
15/09/22 04:46:28 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
15/09/22 04:46:29 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 0 minutes, Emptier interva
```

```
l = 0 minutes.
Deleted /hw3/hw3_2/tgt
15/09/22 04:46:30 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
15/09/22 04:46:31 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.session-id
15/09/22 04:46:31 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
15/09/22 04:46:31 INFO jvm.JvmMetrics: Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already ini
tialized
15/09/22 04:46:31 INFO mapred.FileInputFormat: Total input paths to process : 1
15/09/22 04:46:32 INFO mapreduce.JobSubmitter: number of splits:1
15/09/22 04:46:32 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local2002659329_0001
15/09/22 04:46:32 INFO mapred.LocalDistributedCacheManager: Localized file:/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARN
ING-AT-SCALE/week3/hw3/mapper.py as file:/app/hadoop/tmp/mapred/local/1442922392667/mapper.py
15/09/22 04:46:32 INFO mapred.LocalDistributedCacheManager: Localized file:/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARN
ING-AT-SCALE/week3/hw3/reducer.py as file:/app/hadoop/tmp/mapred/local/1442922392668/reducer.py
15/09/22 04:46:32 INFO mapred.LocalDistributedCacheManager: Localized file:/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARN
ING-AT-SCALE/week3/hw3/combiner.py as file:/app/hadoop/tmp/mapred/local/1442922392669/combiner.py
15/09/22 04:46:33 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
15/09/22 04:46:33 INFO mapreduce.Job: Running job: job_local2002659329_0001
15/09/22 04:46:33 INFO mapred.LocalJobRunner: OutputCommitter set in config null
15/09/22 04:46:33 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapred.FileOutputCommitter
15/09/22 04:46:33 INFO mapred.LocalJobRunner: Waiting for map tasks
15/09/22 04:46:33 INFO mapred.LocalJobRunner: Starting task: attempt_local2002659329_0001_m_000000_0
15/09/22 04:46:33 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
15/09/22 04:46:33 INFO mapred.MapTask: Processing split: hdfs://localhost:54310/hw3/hw3_1/src/ProductPurchaseData.txt:0+34
58517
15/09/22 04:46:33 INFO mapred.MapTask: numReduceTasks: 1
15/09/22 04:46:33 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
15/09/22 04:46:33 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
15/09/22 04:46:33 INFO mapred.MapTask: soft limit at 83886080
15/09/22 04:46:33 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
15/09/22 04:46:33 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
15/09/22 04:46:33 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
15/09/22 04:46:33 INFO streaming.PipeMapRed: PipeMapRed exec [/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARNING-AT-SCALE/
week3/hw3/./mapper.py]
15/09/22 04:46:33 INFO Configuration.deprecation: mapred.tip.id is deprecated. Instead, use mapreduce.task.id
15/09/22 04:46:33 INFO Configuration.deprecation: mapred.local.dir is deprecated. Instead, use mapreduce.cluster.local.dir
15/09/22 04:46:33 INFO Configuration.deprecation: map.input.file is deprecated. Instead, use mapreduce.map.input.file
15/09/22 04:46:33 INFO Configuration.deprecation: mapred.skip.on is deprecated. Instead, use mapreduce.job.skiprecords
15/09/22 04:46:33 INFO Configuration.deprecation: map.input.length is deprecated. Instead, use mapreduce.map.input.length
15/09/22 04:46:33 INFO Configuration.deprecation: mapred.work.output.dir is deprecated. Instead, use mapreduce.task.output
.dir
15/09/22 04:46:33 INFO Configuration.deprecation: map.input.start is deprecated. Instead, use mapreduce.map.input.start
15/09/22 04:46:33 INFO Configuration.deprecation: mapred.job.id is deprecated. Instead, use mapreduce.job.id
15/09/22 04:46:33 INFO Configuration.deprecation: user.name is deprecated. Instead, use mapreduce.job.user.name
15/09/22 04:46:33 INFO Configuration.deprecation: mapred.task.is.map is deprecated. Instead, use mapreduce.task.ismap
15/09/22 04:46:33 INFO Configuration.deprecation: mapred.task.id is deprecated. Instead, use mapreduce.task.attempt.id
15/09/22 04:46:33 INFO Configuration.deprecation: mapred.task.partition is deprecated. Instead, use mapreduce.task.partiti
on
15/09/22 04:46:33 INFO streaming.PipeMapRed: R/W/S=1/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:33 INFO streaming.PipeMapRed: R/W/S=10/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:33 INFO streaming.PipeMapRed: R/W/S=100/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:33 INFO streaming.PipeMapRed: R/W/S=1000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:34 INFO mapreduce.Job: Job job_local2002659329_0001 running in uber mode : false
15/09/22 04:46:34 INFO mapreduce.Job:  map 0% reduce 0%
15/09/22 04:46:34 INFO streaming.PipeMapRed: R/W/S=10000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:36 INFO streaming.PipeMapRed: Records R/W=31101/1
15/09/22 04:46:37 INFO streaming.PipeMapRed: MRErrorThread done
15/09/22 04:46:37 INFO streaming.PipeMapRed: mapRedFinished
15/09/22 04:46:37 INFO mapred.LocalJobRunner:
15/09/22 04:46:37 INFO mapred.MapTask: Starting flush of map output
15/09/22 04:46:37 INFO mapred.MapTask: Spilling map output
15/09/22 04:46:37 INFO mapred.MapTask: bufstart = 0; bufend = 17724293; bufvoid = 104857600
15/09/22 04:46:37 INFO mapred.MapTask: kvstart = 26214396(104857584); kvend = 22655652(90622608); length = 3558745/6553600
15/09/22 04:46:39 INFO mapred.LocalJobRunner: Records R/W=31101/1 > sort
15/09/22 04:46:39 INFO streaming.PipeMapRed: PipeMapRed exec [/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARNING-AT-SCALE/
week3/hw3/./combiner.py]
15/09/22 04:46:39 INFO Configuration.deprecation: mapred.skip.map.auto.incr.proc.count is deprecated. Instead, use mapredu
ce.map.skip.proc-count.auto-incr
15/09/22 04:46:39 INFO streaming.PipeMapRed: R/W/S=1/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:39 INFO streaming.PipeMapRed: R/W/S=10/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:39 INFO streaming.PipeMapRed: R/W/S=100/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:39 INFO streaming.PipeMapRed: R/W/S=1000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:39 INFO streaming.PipeMapRed: R/W/S=10000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:39 INFO streaming.PipeMapRed: R/W/S=100000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:40 INFO streaming.PipeMapRed: R/W/S=200000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:40 INFO mapreduce.Job:  map 67% reduce 0%
15/09/22 04:46:40 INFO streaming.PipeMapRed: R/W/S=300000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:40 INFO streaming.PipeMapRed: R/W/S=400000/0/0 in:400000=400000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:46:40 INFO streaming.PipeMapRed: R/W/S=500000/0/0 in:500000=500000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:46:41 INFO streaming.PipeMapRed: R/W/S=600000/0/0 in:600000=600000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:46:41 INFO streaming.PipeMapRed: R/W/S=700000/0/0 in:700000=700000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:46:41 INFO streaming.PipeMapRed: R/W/S=800000/0/0 in:400000=800000/2 [rec/s] out:0=0/2 [rec/s]
15/09/22 04:46:41 INFO streaming.PipeMapRed: Records R/W=889687/1
15/09/22 04:46:42 INFO mapred.LocalJobRunner: Records R/W=889687/1 > sort
15/09/22 04:46:43 INFO streaming.PipeMapRed: MRErrorThread done
```

```
15/09/22 04:46:43 INFO streaming.PipeMapRed: mapRedFinished
15/09/22 04:46:43 INFO mapred.MapTask: Finished spill 0
15/09/22 04:46:43 INFO mapred.Task: Task:attempt_local2002659329_0001_m_000000_0 is done. And is in the process of committ
ing
15/09/22 04:46:43 INFO mapred.LocalJobRunner: Records R/W=889687/1
15/09/22 04:46:43 INFO mapred.Task: Task 'attempt_local2002659329_0001_m_000000_0' done.
15/09/22 04:46:43 INFO mapred.LocalJobRunner: Finishing task: attempt_local2002659329_0001_m_000000_0
15/09/22 04:46:43 INFO mapred.LocalJobRunner: map task executor complete.
15/09/22 04:46:43 INFO mapred.LocalJobRunner: Waiting for reduce tasks
15/09/22 04:46:43 INFO mapred.LocalJobRunner: Starting task: attempt_local2002659329_0001_r_000000_0
15/09/22 04:46:43 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
15/09/22 04:46:43 INFO mapred.ReduceTask: Using ShuffleConsumerPlugin: org.apache.hadoop.mapreduce.task.reduce.Shuffle@28c
2dc68
15/09/22 04:46:43 INFO reduce.MergeManagerImpl: MergerManager: memoryLimit=363285696, maxSingleShuffleLimit=90821424, merg
eThreshold=239768576, ioSortFactor=10, memToMemMergeOutputsThreshold=10
15/09/22 04:46:43 INFO reduce.EventFetcher: attempt_local2002659329_0001_r_000000_0 Thread started: EventFetcher for fetch
ing Map Completion Events
15/09/22 04:46:43 INFO reduce.LocalFetcher: localfetcher#1 about to shuffle output of map attempt_local2002659329_0001_m_0
00000_0 decomp: 19503669 len: 19503673 to MEMORY
15/09/22 04:46:43 INFO reduce.InMemoryMapOutput: Read 19503669 bytes from map-output for attempt_local2002659329_0001_m_00
0000_0
15/09/22 04:46:43 INFO reduce.MergeManagerImpl: closeInMemoryFile -> map-output of size: 19503669, inMemoryMapOutputs.size
() -> 1, commitMemory -> 0, usedMemory ->19503669
15/09/22 04:46:43 INFO reduce.EventFetcher: EventFetcher is interrupted.. Returning
15/09/22 04:46:43 INFO mapred.LocalJobRunner: 1 / 1 copied.
15/09/22 04:46:43 INFO reduce.MergeManagerImpl: finalMerge called with 1 in-memory map-outputs and 0 on-disk map-outputs
15/09/22 04:46:43 INFO mapred.Merger: Merging 1 sorted segments
15/09/22 04:46:43 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 19503658 bytes
15/09/22 04:46:43 INFO reduce.MergeManagerImpl: Merged 1 segments, 19503669 bytes to disk to satisfy reduce memory limit
15/09/22 04:46:43 INFO reduce.MergeManagerImpl: Merging 1 files, 19503673 bytes from disk
15/09/22 04:46:43 INFO reduce.MergeManagerImpl: Merging 0 segments, 0 bytes from memory into reduce
15/09/22 04:46:43 INFO mapred.Merger: Merging 1 sorted segments
15/09/22 04:46:43 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 19503658 bytes
15/09/22 04:46:43 INFO mapred.LocalJobRunner: 1 / 1 copied.
15/09/22 04:46:43 INFO streaming.PipeMapRed: PipeMapRed exec [/media/sf_shared/GitHub/MIDS-W261-MACHINE-LEARNING-AT-SCALE/
week3/hw3/./reducer.py]
15/09/22 04:46:43 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
15/09/22 04:46:43 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
15/09/22 04:46:44 INFO streaming.PipeMapRed: R/W/S=1/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:44 INFO streaming.PipeMapRed: R/W/S=10/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:44 INFO streaming.PipeMapRed: R/W/S=100/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:44 INFO streaming.PipeMapRed: R/W/S=1000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:44 INFO streaming.PipeMapRed: R/W/S=10000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:44 INFO mapreduce.Job:  map 100% reduce 0%
15/09/22 04:46:44 INFO streaming.PipeMapRed: R/W/S=100000/0/0 in:NA [rec/s] out:NA [rec/s]
15/09/22 04:46:45 INFO streaming.PipeMapRed: R/W/S=200000/0/0 in:200000=200000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:46:45 INFO streaming.PipeMapRed: R/W/S=300000/0/0 in:300000=300000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:46:45 INFO streaming.PipeMapRed: R/W/S=400000/0/0 in:400000=400000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:46:45 INFO streaming.PipeMapRed: R/W/S=500000/0/0 in:500000=500000/1 [rec/s] out:0=0/1 [rec/s]
15/09/22 04:46:46 INFO streaming.PipeMapRed: R/W/S=600000/0/0 in:300000=600000/2 [rec/s] out:0=0/2 [rec/s]
15/09/22 04:46:46 INFO streaming.PipeMapRed: R/W/S=700000/0/0 in:350000=700000/2 [rec/s] out:0=0/2 [rec/s]
15/09/22 04:46:46 INFO streaming.PipeMapRed: R/W/S=800000/0/0 in:400000=800000/2 [rec/s] out:0=0/2 [rec/s]
15/09/22 04:46:47 INFO streaming.PipeMapRed: Records R/W=889687/1
15/09/22 04:46:47 INFO streaming.PipeMapRed: MRErrorThread done
15/09/22 04:46:47 INFO streaming.PipeMapRed: mapRedFinished
15/09/22 04:46:47 INFO mapred.Task: Task:attempt_local2002659329_0001_r_000000_0 is done. And is in the process of committ
ing
15/09/22 04:46:47 INFO mapred.LocalJobRunner: 1 / 1 copied.
15/09/22 04:46:47 INFO mapred.Task: Task attempt_local2002659329_0001_r_000000_0 is allowed to commit now
15/09/22 04:46:47 INFO output.FileOutputCommitter: Saved output of task 'attempt_local2002659329_0001_r_000000_0' to hdfs:
//localhost:54310/hw3/hw3_2/tgt/_temporary/0/task_local2002659329_0001_r_000000
15/09/22 04:46:47 INFO mapred.LocalJobRunner: Records R/W=889687/1 > reduce
15/09/22 04:46:47 INFO mapred.Task: Task 'attempt_local2002659329_0001_r_000000_0' done.
15/09/22 04:46:47 INFO mapred.LocalJobRunner: Finishing task: attempt_local2002659329_0001_r_000000_0
15/09/22 04:46:47 INFO mapred.LocalJobRunner: reduce task executor complete.
15/09/22 04:46:48 INFO mapreduce.Job:  map 100% reduce 100%
15/09/22 04:46:48 INFO mapreduce.Job: Job job_local2002659329_0001 completed successfully
15/09/22 04:46:48 INFO mapreduce.Job: Counters: 38
        File System Counters
                FILE: Number of bytes read=39224872
                FILE: Number of bytes written=59244075
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=6917034
                HDFS: Number of bytes written=411
                HDFS: Number of read operations=13
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
        Map-Reduce Framework
                Map input records=31101
                Map output records=889687
                Map output bytes=17724293
                Map output materialized bytes=19503673
                Input split bytes=112
                Combine input records=889687
                Combine output records=889687
                Reduce input groups=889687
                Reduce shuffle bytes=19503673
                Reduce input records=889687
                Reduce output records=12
                Spilled Records=1779374
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=35
                CPU time spent (ms)=0
                Physical memory (bytes) snapshot=0
                Virtual memory (bytes) snapshot=0
                Total committed heap usage (bytes)=335683584
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=3458517
        File Output Format Counters
                Bytes Written=411
15/09/22 04:46:48 INFO streaming.StreamJob: Output directory: /hw3/hw3_2/tgt

OUTPUT
output from mapper/reducer to determine the number of occurrences of word assistance
15/09/22 04:46:49 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
|C1| = 12592
|L1| = 647
|C2| = 877095
|L2| = 1334
------------------------------------------------------------
Top-5 association rules with confidence scores
------------------------------------------------------------
DAI93865 => FRO40251, conf = 1.0000
ELE12951 => FRO40251, conf = 0.9906
DAI88079 => FRO40251, conf = 0.9867
DAI43868 => SNA82528, conf = 0.9730
DAI23334 => DAI62779, conf = 0.9545
Time taken to find association rules = 21.84 second.
```

HW3.3

Benchmark your results using the pyFIM implementation of the Apriori algorithm (Apriori - Association Rule Induction / Frequent Item Set Mining implemented by Christian Borgelt). Comment on the results from both implementations (your Hadoop MapReduce of apriori versus pyFIM) in terms of results and execution times.

Assumptions

1. The solution does not use results of pyFim directly to account for lexicographical requirement
2. neessary transformation has been made to source the input the data to pyFim apriori implementation

Results

- Results from both the implementations i.e. mapreduce and apriori using pyfim are same.
- There is huge performance difference between the implementations. Mapreduce took ~21 seconds and pyfim took less than second. Following could be attributed for this difference
  - Mapreduce framework has initialization time of 5-6 seconds before mapper begins streaming
  - Mapreduce framework require I/O transfers and handshakes between the different tasks
  - Mapreduce framework performs sort and shuffle whih might be taking time
  - The data structures constructed in the mapreduce may not be efficient
  - pyfim implementation is C code running entirely in a single thread and mostly in memory making it very fast as there is no or minimal I/O transfer required
  - Mapreduce code would perform better with large data sets compared to smaller ones.

In [42]:

```
%%writefile apriori_fim.py
#!/usr/bin/python
import traceback
import fim

try:
    item_counts = {}
    frequencies = []
    rules = []
    # read input file by line and split to
    # store each line as list of items
    # fim apriori expects this data structure as input
    baskets = [ line.split() for line in open('ProductPurchaseData.txt').read().strip().split('\n')]

    # target = 's'       -> frequent item sets
    # supp   = negative  -> minimum support of an item set
    # zmax   = number    -> maximum number of items per item set
    item_sets = fim.apriori(baskets, target='s', supp=-100, zmax=2)

    for r in item_sets:
        # apriori reports in the format ((itemset), support)
        item_set, item_count = r
        # k = 1
        if len(item_set) == 1:
            item_counts[item_set[0]] = item_count
        # k = 2
        elif len(item_set) == 2:
            item1, item2 = item_set
            # lexicographial ordering of the rules
            # report the rule a->b but not the rule b->a
            if item1 < item2:
                frequencies.append(((item1, item2), float(item_count)))

    # calculate confidence
    for rule, count in frequencies:
        conf = count / item_counts[rule[0]]
        rules.append((rule[0], rule[1], conf))

    rules = sorted(rules, key=lambda x: (x[0], x[1]))
    rules = sorted(rules, key=lambda x: (x[2]), reverse=True)

    for rule in rules[:5]:
        print ("{} => {}, conf = {:.4f}").format(rule[0], rule[1], rule[2])

except Exception:
    traceback.print_exc()
```

```
Overwriting apriori_fim.py
```

In [1]:

```
# Use chmod for permissions
!chmod a+x apriori_fim.py
```

Driver Function

In [43]:

```
# HW 3.3  Apriori implementation using pyFim module

def hw3_3():
    import time
    start_time = time.time()

    # run apriori implementation using pyfim
    !./apriori_fim.py

    end_time = time.time()

    print "Time taken to find association rules and report top-5 confidence scores = {:.2f} second.".format(end_time - sta
rt_time)

hw3_3()
```

```
DAI93865 => FRO40251, conf = 1.0000
ELE12951 => FRO40251, conf = 0.9906
DAI88079 => FRO40251, conf = 0.9867
DAI43868 => SNA82528, conf = 0.9730
DAI23334 => DAI62779, conf = 0.9545
Time taken to find association rules and report top-5 confidence scores = 0.64 second.
```

HW3.4

Suppose that you wished to perform the Apriori algorithm once again, though this time now with the goal of listing the top 5 rules with corresponding confidence scores in decreasing order of confidence score for itemsets of size 3 using Hadoop MapReduce. A rule is now of the form:

(item1, item2) ⇒ item3

Recall that the Apriori algorithm is iterative for increasing itemset size, working off of the frequent itemsets of the previous size to explore ONLY the NECESSARY subset of a large combinatorial space. Describe how you might design a framework to perform this exercise.

Apriori algoithm for increasing itemset size

The Apriori algorithm takes advantage of the fact that any subset of a frequent itemset is also a frequent itemset. The algorithm can therefore, reduce the number of candidates being considered by only exploring the itemsets whose support count is greater than the minimum support count. All infrequent itemsets can be pruned if it has an infrequent subset.

Following are teh prerequisites to obtain k-itemsets

- **transaction data**
- **frequent (k-1) itemsets satisfy target threshold such as support threshold in the problem**
- **filter k-itemsets such that (k+1) itemsets are generated with the same steps**

Since this is a recursive approach, it makes fit for use in the MapReduce framework, where between mappers, we share nothing, and likewise for reducers. We need the transaction data and the frequent (k-1)-itemsets to be available in distributed cache for each mapper to read from.

Pseudocode of the data pipeline

```
**Mapper1:**
    input: transaction data
    output:
**Reducer1:**
    input: aggregate the counts for each 1-itemset and filter
    output: <(itemset, count)> prune with target threshold
    store and emit output to next stage for k = 2 frequent itemsets
**Mapper2:**
    input: transaction data and frequent (k-1) itemset to enumerate k-itemset
    output: <(itemset, count)>
**Reducer2:**
    input: <(k-itemset, count)> aggregate the counts for each k-itemset and prune
    output: <(itemset, count)> prune with target threshold
    store and emit output to next stage for k + 1 frequent itemsets
[[Recursive Mapper 2 and Reducer 2 until k]]

**Reducer:**
    input: <k-itemsets, count>
    output: rules with confidence scores
```

Finally, to create rules a final reducer will be called. In this step, reduer reads frequent 3-itemsets and frequent 2-itemsets and computes confidence for all possible 2-item subsets. For top-5 rules, a sorted list that is sliced to get max 5 confidence and the corresponding rules.

**** -- END OF ASSIGNMENT 3 -- ****