DATASCI W261: Machine Learning at Scale

Assignment Week 3

Minhchau Dang (minhchau.dang@berkeley.edu)
Miki Seltzer (miki.seltzer@berkeley.edu)
W261-2, Spring 2016
Submission: 2016-Feb-02 8:50pm

HW3.0

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

Merge sort is used to combine two pre-sorted lists. It is a very efficient sort, as it only needs to iteratively look for the smallest element in multiple sorted lists. It is utilized in Hadoop during the shuffle, when key-value pairs are shuffled to reducers, then sorted.

How is a combiner function in the context of Hadoop?

A combiner function allows for preaggregation before key-value pairs are sent from the mappers to reducers. It is similar to a reducer, but there are some key differences. One difference is that the combiner may not always be used during the job -- it may be used 0, 1, or many times. Thus, we cannot count on Hadoop actually using a combiner we have included in the job, and we must be careful about matching output types of the mapper and combiner.

Give an example where it can be used and justify why it should be used in the context of this problem.

In the classic word count example, a document is scanned, and each word is paired with the value of 1. A combiner can be used to combine values of 1 with the same key (word) before they are shuffled to reducers. This reduces the amount of data that is shuffled between the mapper and reducer, and increases efficiency.

What is the Hadoop shuffle?

The Hadoop shuffle is the process by which data from mappers is shuffled and sorted while being sent to reducers. The shuffle ensures that keys are grouped together and sorted within the reducer they are sent to.

HW3.1: Use Counters to do EDA (exploratory data analysis and to monitor progress)

In [343]:

```
# I am running this locally, so make sure that the Hadoop streaming API
is in this folder.
!wget http://central.maven.org/maven2/org/apache/hadoop/hadoop-streamin
g/2.7.1/hadoop-streaming-2.7.1.jar

# Create a folder on HDFS for this week's assignment, strip the header l
ine from Consumer_Complaints.csv
!echo "$(tail -n +2 Consumer_Complaints.csv)" > Consumer_Complaints.csv
!hdfs dfs -mkdir /user/miki/week03
!hdfs dfs -put Consumer_Complaints.csv /user/miki/week03
```

```
mkdir: `/user/miki/week03': File exists
```

In [344]:

```
%%writefile mapper_31.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.1

import sys
from csv import reader

# Our input comes from STDIN (standard input)
for line in reader(sys.stdin):
    product = line[1]
    if product == "Debt collection": sys.stderr.write("reporter:counter:
Product,Debt,1\n")
    elif product == "Mortgage": sys.stderr.write("reporter:counter:Produ
ct,Mortgage,1\n")
    else: sys.stderr.write("reporter:counter:Product,Other,1\n")
    print line
```

```
Writing mapper_31.py
```

In [345]:
```
%%writefile reducer_31.py
#!/usr/bin/python
## reducer.py
## Author: Miki Seltzer
## Description: reducer code for HW3.1

import sys
from operator import itemgetter
from csv import reader

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    print line
```

```
Writing reducer_31.py
```

In [346]:
```
# Change permissions on mapper and reducer
!chmod +x mapper_31.py
!chmod +x reducer_31.py

# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/hw3_1_output

# Run job
!hadoop jar hadoop-streaming-2.7.1.jar \
-mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_31.py \
-reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_31.py \
-input /user/miki/week03/Consumer_Complaints.csv \
-output /user/miki/week03/hw3_1_output
```

```
Deleted /user/miki/week03/hw3_1_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob7547497993532515688.jar tmpDir=null
```

Screenshot

Custom Counters

HW 3.2 Analyze the performance of your Mappers, Combiners and Reducers using Counters

In [18]:
```
%%writefile HW3_2_input.txt
foo foo quux labs foo bar quux
```

```
Overwriting HW3_2_input.txt
```

In [19]:
```
!hdfs dfs -put HW3_2_input.txt /user/miki/week03
```

In [347]:
```
%%writefile mapper_32a.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.2

import sys
from csv import reader

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Mapper,1\n")

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    line = line.split()
    for word in line:
        print '%s\t%s' % (word, 1)
```

```
Writing mapper_32a.py
```

```
In [348]: %%writefile reducer_32a.py
          #!/usr/bin/python
          ## reducer.py
          ## Author: Miki Seltzer
          ## Description: reducer code for HW3.2

          import sys
          from operator import itemgetter

          # Increment mapper counter
          sys.stderr.write("reporter:counter:Custom_Counter,Reducer,1\n")

          # Our input comes from STDIN (standard input)
          for line in sys.stdin:
              print line
```

```
Writing reducer_32a.py
```

```
In [349]: # Change permissions on mapper and reducer
          !chmod +x mapper_32a.py
          !chmod +x reducer_32a.py

          # If output folder already exists, delete it
          !hdfs dfs -rm -r /user/miki/week03/hw3_2a_output

          # Run job
          !hadoop jar hadoop-streaming-2.7.1.jar \
          -D mapred.map.tasks=1 \
          -D mapred.reduce.tasks=4 \
          -mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_32a.py \
          -reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_32a.py \
          -input /user/miki/week03/HW3_2_input.txt \
          -output /user/miki/week03/hw3_2a_output
```

```
Deleted /user/miki/week03/hw3_2a_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob7785257004505429063.jar tmpDir=null
```

What is the value of your user defined Mapper Counter, and Reducer Counter after completing this word count job? The answer should be 1 and 4 respectively. Please explain.

I had to specify the number of map tasks and reduce tasks to get 1 and 4, since the defaults produced counters of 2 and 1 respectively.

The counters were incremented each time the mapper and reducer scripts were executed.

HW3.2b: Perform a word count analysis of the Issue column of the Consumer Complaints Dataset

In [302]:
```
%%writefile mapper_32b.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.2b

import sys
from csv import reader
import string

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Mapper,1\n")

# Initialize variables
total = 0

# Our input comes from STDIN (standard input)
for line in reader(sys.stdin):
    # Format our line
    issue = line[3].lower()
    issue = issue.replace(',',' ').replace('/',' ')

    for word in issue.split():
        if len(word) > 0:
            print '%s\t%s' % (word, 1)
            total += 1

# Print total words
print '%s\t%s' % ('*total', total)
```

```
Writing mapper_32b.py
```

In [303]:
```
%%writefile reducer_32b.py
#!/usr/bin/python
## reducer.py
## Author: Miki Seltzer
## Description: reducer code for HW3.2b

import sys
from operator import itemgetter

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Reducer,1\n")

# Initialize variables
prev_word = None
prev_count = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:

    # Split line
    word, count = line.split('\t')

    # Convert count (currently a string) to int
    try:
        count = int(count)
    except ValueError:
        # Count wasn't an int, so move on
        continue

    if prev_word == word:
        # We haven't moved to a new word
        prev_count += count

    else:
        if prev_word:
            print '%s\t%s' % (prev_word, prev_count)

        prev_count = count
        prev_word = word

# Output the last line
if prev_word == word:
    print '%s\t%s' % (prev_word, prev_count)
```

```
Writing reducer_32b.py
```

```
In [350]: # Change permissions on mapper and reducer
          !chmod +x mapper_32b.py
          !chmod +x reducer_32b.py

          # If output folder already exists, delete it
          !hdfs dfs -rm -r /user/miki/week03/hw3_2b_output

          # Run job
          !hadoop jar hadoop-streaming-2.7.1.jar \
          -D mapred.reduce.tasks=4 \
          -mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_32b.py \
          -reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_32b.py \
          -input /user/miki/week03/Consumer_Complaints.csv \
          -output /user/miki/week03/hw3_2b_output
```

```
Deleted /user/miki/week03/hw3_2b_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob1465294680810041979.jar tmpDir=null
```

What is the value of your user defined Mapper Counter, and Reducer Counter after completing your word count job?

After completing this job, the counters show the following values:

- Mapper: 2
- Reducer: 4 (this is explicitly set when running the job)

HW3.2c: Perform a word count analysis of the Issue column of the Consumer Complaints Dataset (ADD: standalone combiner)

We can reuse the reducer in this case, and rename it combiner. We update the line to increment the combiner counter.

```
In [305]: %%writefile combiner_32c.py
          #!/usr/bin/python
          ## combiner.py
          ## Author: Miki Seltzer
          ## Description: combiner code for HW3.2c

          import sys
          from operator import itemgetter

          # Increment mapper counter
          sys.stderr.write("reporter:counter:Custom_Counter,Combiner,1\n")

          prev_word = None
          prev_count = 0

          # Our input comes from STDIN (standard input)
          for line in sys.stdin:
              word, count = line.split('\t')

              # Convert count (currently a string) to int
              try:
                  count = int(count)
              except ValueError:
                  # Count wasn't an int, so move on
                  continue

              # Check if we've moved to a new word
              if prev_word == word:
                  prev_count += count
              else:
                  if prev_word:
                      # We are at a new word, need to print previous word sum
                      print '%s\t%s' % (prev_word, prev_count)
                  prev_count = count
                  prev_word = word

          # Output the last line
          if prev_word == word:
              print '%s\t%s' % (prev_word, prev_count)
```

```
Writing combiner_32c.py
```

In [351]:
```
# Change permissions on mapper and reducer
!chmod +x mapper_32b.py
!chmod +x combiner_32c.py
!chmod +x reducer_32b.py

# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/hw3_2c_output

# Run job
!hadoop jar hadoop-streaming-2.7.1.jar \
-D mapred.reduce.tasks=4 \
-mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_32b.py \
-combiner /home/cloudera/Documents/W261-Fall2016/Week03/combiner_32c.py
\
-reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_32b.py \
-input /user/miki/week03/Consumer_Complaints.csv \
-output /user/miki/week03/hw3_2c_output
```

```
Deleted /user/miki/week03/hw3_2c_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob2612824031763828089.jar tmpDir=null
```

What is the value of your user defined Mapper Counter, Combiner Counter and Reducer Counter after completing your word count job?

After completing this job, the counters show the following values:

- Mapper: 2
- Combiner: 8
- Reducer: 4 (this is explicitly set when running the job)

HW3.2d: Using a single reducer, present frequency and relative frequency of top 50 and bottom 10 terms

For this section, we only need an identity mapper and an identity reducer.

In [309]:
```
%%writefile mapper_32d.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.2d

import sys
from csv import reader
import string

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Mapper,1\n")

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    word, count = line.replace('\n','').split('\t')
    print '%s\t%s' % (count, word)
```

```
Overwriting mapper_32d.py
```

In [310]:
```
%%writefile reducer_32d.py
#!/usr/bin/python
## reducer.py
## Author: Miki Seltzer
## Description: reducer code for HW3.2d

import sys
from operator import itemgetter

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Reducer,1\n")

# Initialize variables
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    fields = line.replace('\n','').split('\t')
    count = fields[0]
    word = fields[1]

    try:
        count = int(count)
    except ValueError:
        continue

    # The first word should be *total, save this as total
    if word == '*total': total = float(count)
    else: print '%s\t%s\t%s' % (word, count, count/total)
```

```
Overwriting reducer_32d.py
```

```
In [352]: # Change permissions on mapper and reducer
          !chmod +x mapper_32d.py
          !chmod +x reducer_32d.py

          # If output folder already exists, delete it
          !hdfs dfs -rm -r /user/miki/week03/hw3_2d_output

          # Run job
          !hadoop jar hadoop-streaming-2.7.1.jar \
          -D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFi
          eldBasedComparator \
          -D mapred.text.key.partitioner.options=-k1,1 \
          -D stream.num.map.output.key.fields=2 \
          -D mapred.text.key.comparator.options='-k1,1nr -k2,2n' \
          -D mapred.reduce.tasks=1 \
          -mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_32d.py \
          -reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_32d.py \
          -input /user/miki/week03/hw3_2b_output/part* \
          -output /user/miki/week03/hw3_2d_output
```

```
Deleted /user/miki/week03/hw3_2d_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob6548003399415369612.jar tmpDir=null
```

```
In [353]: !rm hw3_2d_output.txt
          !hdfs dfs -copyToLocal /user/miki/week03/hw3_2d_output/part-00000 hw3_2d
          _output.txt
```

In [354]:

```
# Function to pretty print:
# - the top x and bottom y items
# - unique items
def print_results(file, x=50, y=10):
    words = []
    special_words = []

    with open(file,'r') as myfile:
        for line in myfile:
            fields = line.replace('\n','').split('\t')
            if fields[0][0] != '*': words.append(fields)
            else: special_words.append(fields)

    print '      {:16s}{:>8s}{:>15s}'.format('word', 'count', 'relative
freq')
    print '---------------------------------------------'
    for i in range(x):
        print '[{:3d}] {:16s}{:8,d}{:15.2%}'.format(i+1,
                                                    words[i][0],
                                                    int(words[i][1]),
                                                    float(words[i][2]))
    print '...'
    for i in range(y):
        j = len(words) - 10 + i
        print '[{:3d}] {:16s}{:8,d}{:15.2%}'.format(j+1,
                                                    words[j][0],
                                                    int(words[j][1]),
                                                    float(words[j][2]))

    print '\n---------------------------------------------'
    print '      {:16s}{:>8,d}'.format('Unique words', len(words))
    for item in special_words:
        name = item[0][1:].replace('_',' ')
        print '      {:16s}{:>8,d}'.format(name, int(item[1]))

print_results('hw3_2d_output.txt')
```

```
      word               count  relative freq
--------------------------------------------
[  1] loan             119,630          8.87%
[  2] collection        72,394          5.37%
[  3] foreclosure       70,487          5.23%
[  4] modification      70,487          5.23%
[  5] account           57,448          4.26%
[  6] credit            55,251          4.10%
[  7] or                40,508          3.00%
[  8] payments          39,993          2.97%
[  9] escrow            36,767          2.73%
[ 10] servicing         36,767          2.73%
[ 11] report            34,903          2.59%
[ 12] incorrect         29,133          2.16%
[ 13] information       29,069          2.16%
[ 14] on                29,069          2.16%
[ 15] debt              27,874          2.07%
[ 16] closing           19,000          1.41%
[ 17] not               18,477          1.37%
[ 18] attempts          17,972          1.33%
[ 19] cont'd            17,972          1.33%
[ 20] collect           17,972          1.33%
[ 21] owed              17,972          1.33%
[ 22] and               16,448          1.22%
[ 23] management        16,205          1.20%
[ 24] opening           16,205          1.20%
[ 25] of                13,983          1.04%
[ 26] my                10,731          0.80%
[ 27] deposits          10,555          0.78%
[ 28] withdrawals       10,555          0.78%
[ 29] problems           9,484          0.70%
[ 30] application        8,868          0.66%
[ 31] communication      8,671          0.64%
[ 32] tactics            8,671          0.64%
[ 33] broker             8,625          0.64%
[ 34] mortgage           8,625          0.64%
[ 35] originator         8,625          0.64%
[ 36] to                 8,401          0.62%
[ 37] unable             8,178          0.61%
[ 38] billing            8,158          0.61%
[ 39] other              7,886          0.58%
[ 40] disclosure         7,655          0.57%
[ 41] verification       7,655          0.57%
[ 42] disputes           6,938          0.51%
[ 43] reporting          6,559          0.49%
[ 44] lease              6,337          0.47%
[ 45] the                6,248          0.46%
[ 46] by                 5,663          0.42%
[ 47] being              5,663          0.42%
[ 48] caused             5,663          0.42%
[ 49] funds              5,663          0.42%
[ 50] low                5,663          0.42%
...
```

```
[165] apply                 118          0.01%
[166] amount                 98          0.01%
[167] credited               92          0.01%
[168] payment                92          0.01%
[169] checks                 75          0.01%
[170] convenience            75          0.01%
[171] amt                    71          0.01%
[172] day                    71          0.01%
[173] disclosures            64          0.00%
[174] missing                64          0.00%

-------------------------------------------
      Unique words          174
```

HW3.3. Shopping Cart Analysis Exploratory Data Analysis

We can reuse the reducer from HW3.2b, but there are small changes that need to be made to the mapper:

- We do not have to format the products to lower case, assume there is no punctuation stripping needed
- Keep track of the largest basket size as we loop through baskets

In [282]:
```
!hdfs dfs -put ProductPurchaseData.txt /user/miki/week03
```

In [325]:

```
%%writefile mapper_33a.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.3a

import sys

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Mapper,1\n")

# Initialize variables
total = 0
basket_size = 0
largest_basket_size = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    # Split our line into products
    for product in line.replace('\n','').split():
        print '%s\t%s' % (product, 1)
        basket_size += 1
        total += 1
    if basket_size > largest_basket_size:
        largest_basket_size = basket_size

    basket_size = 0

# Print total words
print '%s\t%s' % ('*total', total)
print '%s\t%s' % ('*largest_basket', largest_basket_size)
```

```
Overwriting mapper_33a.py
```

In [326]:
```
# Change permissions on mapper and reducer
!chmod +x mapper_33a.py

# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/hw3_3a_output

# Run job
!hadoop jar hadoop-streaming-2.7.1.jar \
-mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_33a.py \
-reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_32b.py \
-input /user/miki/week03/ProductPurchaseData.txt \
-output /user/miki/week03/hw3_3a_output
```

```
Deleted /user/miki/week03/hw3_3a_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob4662501162287957069.jar tmpDir=null
```

Using a single reducer: Report your findings such as number of unique products; largest basket; report the top 50 most frequently purchased items, their frequency, and their relative frequency (break ties by sorting the products alphabetical order) etc. using Hadoop Map-Reduce.

We can use the mapper and reducer from HW3.2d to get the sorted frequencies and relative frequencies of the products.

In [327]:
```
# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/hw3_3b_output

# Run job
!hadoop jar hadoop-streaming-2.7.1.jar \
-D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFi
eldBasedComparator \
-D mapred.text.key.partitioner.options=-k1,1 \
-D stream.num.map.output.key.fields=2 \
-D mapred.text.key.comparator.options='-k1,1nr -k2,2n' \
-D mapred.reduce.tasks=1 \
-mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_32d.py \
-reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_32d.py \
-input /user/miki/week03/hw3_3a_output/part* \
-output /user/miki/week03/hw3_3b_output
```

```
Deleted /user/miki/week03/hw3_3b_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob7461671848105384853.jar tmpDir=null
```

In [330]:
```
!rm hw3_3b_output.txt
!hdfs dfs -copyToLocal /user/miki/week03/hw3_3b_output/part-00000 hw3_3b
_output.txt
```

```
In [341]: print_results('hw3_3b_output.txt', 50, 0)
```

```
      word             count  relative freq
------------------------------------------
[  1] DAI62779         6,667          1.75%
[  2] FRO40251         3,881          1.02%
[  3] ELE17451         3,875          1.02%
[  4] GRO73461         3,602          0.95%
[  5] SNA80324         3,044          0.80%
[  6] ELE32164         2,851          0.75%
[  7] DAI75645         2,736          0.72%
[  8] SNA45677         2,455          0.64%
[  9] FRO31317         2,330          0.61%
[ 10] DAI85309         2,293          0.60%
[ 11] ELE26917         2,292          0.60%
[ 12] FRO80039         2,233          0.59%
[ 13] GRO21487         2,115          0.56%
[ 14] SNA99873         2,083          0.55%
[ 15] GRO59710         2,004          0.53%
[ 16] GRO71621         1,920          0.50%
[ 17] FRO85978         1,918          0.50%
[ 18] GRO30386         1,840          0.48%
[ 19] ELE74009         1,816          0.48%
[ 20] GRO56726         1,784          0.47%
[ 21] DAI63921         1,773          0.47%
[ 22] GRO46854         1,756          0.46%
[ 23] ELE66600         1,713          0.45%
[ 24] DAI83733         1,712          0.45%
[ 25] FRO32293         1,702          0.45%
[ 26] ELE66810         1,697          0.45%
[ 27] SNA55762         1,646          0.43%
[ 28] DAI22177         1,627          0.43%
[ 29] FRO78087         1,531          0.40%
[ 30] ELE99737         1,516          0.40%
[ 31] ELE34057         1,489          0.39%
[ 32] GRO94758         1,489          0.39%
[ 33] FRO35904         1,436          0.38%
[ 34] FRO53271         1,420          0.37%
[ 35] SNA93860         1,407          0.37%
[ 36] SNA90094         1,390          0.36%
[ 37] GRO38814         1,352          0.36%
[ 38] ELE56788         1,345          0.35%
[ 39] GRO61133         1,321          0.35%
[ 40] DAI88807         1,316          0.35%
[ 41] ELE74482         1,316          0.35%
[ 42] ELE59935         1,311          0.34%
[ 43] SNA96271         1,295          0.34%
[ 44] DAI43223         1,290          0.34%
[ 45] ELE91337         1,289          0.34%
[ 46] GRO15017         1,275          0.33%
[ 47] DAI31081         1,261          0.33%
[ 48] GRO81087         1,220          0.32%
[ 49] DAI22896         1,219          0.32%
[ 50] GRO85051         1,214          0.32%
...
```

```
-------------------------------------------
      Unique words      12,592
      largest basket        74
```

HW3.4: Pairs

Suppose we want to recommend new products to the customer based on the products they have already browsed on the online website. Write a map-reduce program to find products which are frequently browsed together. Fix the support count (cooccurence count) to s = 100 (i.e. product pairs need to occur together at least 100 times to be considered frequent) and find pairs of items (sometimes referred to itemsets of size 2 in association rule mining) that have a support count of 100 or more.

In [360]:

```
%%writefile mapper_34a.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.4a

import sys
import itertools

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Mapper,1\n")

# Initialize variables
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    # Split our line into products
    products = line.replace('\n','').split()

    # Get all combinations of products:
    #  - Use a set to remove duplicate products
    #  - Combinations finds tuples of length 2 with no repeats
    pairs = list(itertools.combinations(set(products), 2))

    # For each pair, sort the pair alphabetically, then emit
    for pair in pairs:
        sorted_pair = sorted(pair)
        print '%s\t%s\t%s' % (sorted_pair[0], sorted_pair[1], 1)

    # Increment total number of baskets
    total += 1

# Print total words
print '%s\t%s\t%s' % ('*total', '', total)
```

```
Overwriting mapper_34a.py
```

In [365]:
```
%%writefile reducer_34a.py
#!/usr/bin/python
## reducer.py
## Author: Miki Seltzer
## Description: reducer code for HW3.4a

import sys
from operator import itemgetter

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Reducer,1\n")

# Initialize variables
prev_pair = []
prev_count = 0
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    # Define our key and value
    fields = line.replace('\n','').split('\t')
    pair = [fields[0], fields[1]]
    count = fields[2]

    # Convert count (currently a string) to int
    try:
        count = int(count)
    except ValueError:
        # Count wasn't an int, so move on
        continue

    # Check if we've moved to a new word
    if prev_pair == pair:
        prev_count += count
    else:
        if len(prev_pair) > 0:
            # We are at a new pair, need to print previous pair sum
            print '%s\t%s\t%s' % (prev_pair[0], prev_pair[1], prev_coun
t)
        prev_count = count
        prev_pair = pair

# Output the last line
if prev_pair == pair:
    print '%s\t%s\t%s' % (prev_pair[0], prev_pair[1], prev_count)
```

```
Overwriting reducer_34a.py
```

In [21]:
```
# Change permissions on mapper and reducer
!chmod +x mapper_34a.py
!chmod +x reducer_34a.py

# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/hw3_4a_output

# Run job
!time hadoop jar hadoop-streaming-2.7.1.jar \
-D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFi
eldBasedComparator \
-D mapred.text.key.partitioner.options=-k1,1 \
-D stream.num.map.output.key.fields=2 \
-D mapred.text.key.comparator.options='-k1,1 -k2,2' \
-mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_34a.py \
-reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_34a.py \
-input /user/miki/week03/ProductPurchaseData.txt \
-output /user/miki/week03/hw3_4a_output
```

```
Deleted /user/miki/week03/hw3_4a_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob5023424855138306517.jar tmpDir=null

real	0m50.284s
user	0m5.474s
sys	0m0.365s
```

Now we have each pair and the number of times that the pair co-occurs. We need to run another job to calculate the relative frequency and sort the resulting pairs

In [379]:

```
%%writefile mapper_34b.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.4b

import sys

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Mapper,1\n")

# Initialize variables
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    fields = line.replace('\n','').split('\t')
    if int(fields[2]) >= 100:
        print '%s\t%s\t%s' % (fields[2], fields[0], fields[1])
```

```
Overwriting mapper_34b.py
```

In [381]:
```
%%writefile reducer_34b.py
#!/usr/bin/python
## reducer.py
## Author: Miki Seltzer
## Description: reducer code for HW3.4b

import sys
from operator import itemgetter

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Reducer,1\n")

# Initialize variables
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    fields = line.replace('\n','').split('\t')
    count = fields[0]
    item1 = fields[1]
    item2 = fields[2]

    try:
        count = int(count)
    except ValueError:
        continue

    # The first word should be *total, save this as total
    if item1 == '*total': total = float(count)
    else: print '%s\t%s\t%s\t%s' % (item1, item2, count, count/total)
```

```
Writing reducer_34b.py
```

In [383]:

```
# Change permissions on mapper and reducer
!chmod +x mapper_34b.py
!chmod +x reducer_34b.py

# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/hw3_4b_output

# Run job
!time hadoop jar hadoop-streaming-2.7.1.jar \
-D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFi
eldBasedComparator \
-D mapred.text.key.partitioner.options=-k1,1 \
-D stream.num.map.output.key.fields=3 \
-D mapred.text.key.comparator.options='-k1,1nr -k2,2 -k3,3' \
-D mapred.reduce.tasks=1 \
-mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_34b.py \
-reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_34b.py \
-input /user/miki/week03/hw3_4a_output/part* \
-output /user/miki/week03/hw3_4b_output
```

```
Deleted /user/miki/week03/hw3_4b_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob2299545763059091234.jar tmpDir=null

real	0m26.758s
user	0m4.982s
sys	0m0.345s
```

In [386]:

```
!hdfs dfs -copyToLocal /user/miki/week03/hw3_4b_output/part-00000 hw3_4b
_output.txt
```

```
In [31]: # Function to pretty print:
         # - the top x and bottom y items
         # - unique items
         def print_results(file, x=50, y=10):
             words = []
             special_words = []

             with open(file,'r') as myfile:
                 for line in myfile:
                     fields = line.replace('\n','').split('\t')
                     if fields[0][0] != '*': words.append(fields)
                     else: special_words.append(fields)

             print '      {:10s}{:10s}{:>8s}{:>15s}'.format('item1', 'item2', 'co
         unt', 'relative freq')
             print '-------------------------------------------------------'
             for i in range(x):
                 print '[{:3d}] {:10s}{:10s}{:8,d}{:15.2%}'.format(i+1,
                                                                  words[i][0],
                                                                  words[i][1],
                                                                  int(words[i]
         [2]),
                                                                  float(words[i]
         [3]))
```

List the top 50 product pairs

```
In [395]: print_results('hw3_4b_output.txt', 50, 0)
```

```
          item1     item2        count  relative freq
------------------------------------------------------
[   1] DAI62779  ELE17451     1,592          5.12%
[   2] FRO40251  SNA80324     1,412          4.54%
[   3] DAI75645  FRO40251     1,254          4.03%
[   4] FRO40251  GRO85051     1,213          3.90%
[   5] DAI62779  GRO73461     1,139          3.66%
[   6] DAI75645  SNA80324     1,130          3.63%
[   7] DAI62779  FRO40251     1,070          3.44%
[   8] DAI62779  SNA80324       923          2.97%
[   9] DAI62779  DAI85309       918          2.95%
[  10] ELE32164  GRO59710       911          2.93%
[  11] DAI62779  DAI75645       882          2.84%
[  12] FRO40251  GRO73461       882          2.84%
[  13] DAI62779  ELE92920       877          2.82%
[  14] FRO40251  FRO92469       835          2.68%
[  15] DAI62779  ELE32164       832          2.68%
[  16] DAI75645  GRO73461       712          2.29%
[  17] DAI43223  ELE32164       711          2.29%
[  18] DAI62779  GRO30386       709          2.28%
[  19] ELE17451  FRO40251       697          2.24%
[  20] DAI85309  ELE99737       659          2.12%
[  21] DAI62779  ELE26917       650          2.09%
[  22] GRO21487  GRO73461       631          2.03%
[  23] DAI62779  SNA45677       604          1.94%
[  24] ELE17451  SNA80324       597          1.92%
[  25] DAI62779  GRO71621       595          1.91%
[  26] DAI62779  SNA55762       593          1.91%
[  27] DAI62779  DAI83733       586          1.88%
[  28] ELE17451  GRO73461       580          1.86%
[  29] GRO73461  SNA80324       562          1.81%
[  30] DAI62779  GRO59710       561          1.80%
[  31] DAI62779  FRO80039       550          1.77%
[  32] DAI75645  ELE17451       547          1.76%
[  33] DAI62779  SNA93860       537          1.73%
[  34] DAI55148  DAI62779       526          1.69%
[  35] DAI43223  GRO59710       512          1.65%
[  36] ELE17451  ELE32164       511          1.64%
[  37] DAI62779  SNA18336       506          1.63%
[  38] ELE32164  GRO73461       486          1.56%
[  39] DAI62779  FRO78087       482          1.55%
[  40] DAI85309  ELE17451       482          1.55%
[  41] DAI62779  GRO94758       479          1.54%
[  42] DAI62779  GRO21487       471          1.51%
[  43] GRO85051  SNA80324       471          1.51%
[  44] ELE17451  GRO30386       468          1.50%
[  45] FRO85978  SNA95666       463          1.49%
[  46] DAI62779  FRO19221       462          1.49%
[  47] DAI62779  GRO46854       461          1.48%
[  48] DAI43223  DAI62779       459          1.48%
[  49] ELE92920  SNA18336       455          1.46%
[  50] DAI88079  FRO40251       446          1.43%
```

Report the compute time for the Pairs job.

The 1st job (counts) reports the following compute times:

```
real    0m50.284s
user    0m5.474s
sys     0m0.365s
```

The 2nd job (sorts) reports the following compute times:

```
real    0m26.758s
user    0m4.982s
sys     0m0.345s
```

Describe the computational setup used (E.g., single computer; dual core; linux, number of mappers, number of reducers)

Cloudera QuickStart VM: single computer, 2 processors, 2 mappers (default), 1 reducer

How many times is each mapper and reducer called?

- Mapper: 2
- Reducer: 1

HW3.5: Stripes

Repeat 3.4 using the stripes design pattern for finding cooccuring pairs.

In [5]:
```
%%writefile mapper_35a.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.5a

import sys
import itertools

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Mapper,1\n")

# Initialize variables
total = 0
stripes = {}

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    # Split our line into products
    products = line.replace('\n','').split()

    # Get all combinations of products:
    #  - Use a set to remove duplicate products
    #  - Combinations finds tuples of length 2 with no repeats
    items = sorted(list(set(products)))

    for i in range(len(items)-1):
        for j in range(i+1, len(items)):
            stripes[items[j]] = 1
        print '%s\t%s' % (items[i], stripes)
        stripes = {}

    # Increment total number of baskets
    total += 1

# Print total words
print '%s\t%s' % ('*total', {'*total':total})
```

```
Overwriting mapper_35a.py
```

In [9]:
```
%%writefile reducer_35a.py
#!/usr/bin/python
## reducer.py
## Author: Miki Seltzer
## Description: reducer code for HW3.5a

import sys
from operator import itemgetter

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Reducer,1\n")

# Initialize variables
prev_word = None
prev_stripe = {}
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    # Define our key and value
    fields = line.replace('\n','').split('\t')
    word = fields[0]
    stripe = eval(fields[1])

    # Check if we've moved to a new word
    if prev_word == word:
        # We need to move through the dictionary and update counts
        for item in stripe:
            if item in prev_stripe:
                prev_stripe[item] += stripe[item]
            else:
                prev_stripe[item] = stripe[item]

    else:
        if len(prev_stripe) > 0:
            # We are at a new pair, need to print previous pair sum
            print '%s\t%s' % (prev_word, prev_stripe)
        prev_stripe = stripe
        prev_word = word

# Output the last line
if prev_stripe == stripe:
    print '%s\t%s' % (prev_word, prev_stripe)
```

```
Overwriting reducer_35a.py
```

In [22]:
```
# Change permissions on mapper and reducer
!chmod +x mapper_35a.py
!chmod +x reducer_35a.py

# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/hw3_5a_output

# Run job
!time hadoop jar hadoop-streaming-2.7.1.jar \
-mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_35a.py \
-reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_35a.py \
-input /user/miki/week03/ProductPurchaseData.txt \
-output /user/miki/week03/hw3_5a_output
```

```
Deleted /user/miki/week03/hw3_5a_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob8692232309490935031.jar tmpDir=null

real    0m44.977s
user    0m5.111s
sys     0m0.337s
```

Now we have a stripe for each product (item1). The stripe contains the second item in the pair (item2), along with the count of co-occurrences with item1. Now, we need to "unpack" the stripe, and sort the pairs.

In [26]:
```
%%writefile mapper_35b.py
#!/usr/bin/python
## mapper.py
## Author: Miki Seltzer
## Description: mapper code for HW3.5b

import sys
import itertools

# Increment mapper counter
sys.stderr.write("reporter:counter:Custom_Counter,Mapper,1\n")

# Initialize variables
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    # Define our key and value
    fields = line.replace('\n','').split('\t')
    word = fields[0]
    stripe = eval(fields[1])
    # Now we need to "unpack" the stripe and emit each pair for sorting
    for item in stripe:
        if stripe[item] >= 100:
            print '%s\t%s\t%s' % (stripe[item], word, item)
```

```
Overwriting mapper_35b.py
```

Since this mapper unpacks the pairs and emits them in the same format as the Pairs method in HW3.4, we can use the same reducer from the previous part to find the relative frequencies and top 50 pairs of items.

In [28]:
```
# Change permissions on mapper and reducer
!chmod +x mapper_35b.py

# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/hw3_5b_output

# Run job
!time hadoop jar hadoop-streaming-2.7.1.jar \
-D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFi
eldBasedComparator \
-D mapred.text.key.partitioner.options=-k1,1 \
-D stream.num.map.output.key.fields=3 \
-D mapred.text.key.comparator.options='-k1,1nr -k2,2 -k3,3' \
-D mapred.reduce.tasks=1 \
-mapper /home/cloudera/Documents/W261-Fall2016/Week03/mapper_35b.py \
-reducer /home/cloudera/Documents/W261-Fall2016/Week03/reducer_34b.py \
-input /user/miki/week03/hw3_5a_output/part* \
-output /user/miki/week03/hw3_5b_output
```

```
Deleted /user/miki/week03/hw3_5b_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob2185279117818801498.jar tmpDir=null

real	0m34.390s
user	0m5.735s
sys	0m0.335s
```

In [29]:
```
!hdfs dfs -copyToLocal /user/miki/week03/hw3_5b_output/part-00000 hw3_5b
_output.txt
```

List the top 50 product pairs

```
In [32]: print_results('hw3_5b_output.txt', 50, 0)
```

```
          item1     item2        count  relative freq
-------------------------------------------------------
[   1] DAI62779  ELE17451     1,592          5.12%
[   2] FRO40251  SNA80324     1,412          4.54%
[   3] DAI75645  FRO40251     1,254          4.03%
[   4] FRO40251  GRO85051     1,213          3.90%
[   5] DAI62779  GRO73461     1,139          3.66%
[   6] DAI75645  SNA80324     1,130          3.63%
[   7] DAI62779  FRO40251     1,070          3.44%
[   8] DAI62779  SNA80324       923          2.97%
[   9] DAI62779  DAI85309       918          2.95%
[  10] ELE32164  GRO59710       911          2.93%
[  11] DAI62779  DAI75645       882          2.84%
[  12] FRO40251  GRO73461       882          2.84%
[  13] DAI62779  ELE92920       877          2.82%
[  14] FRO40251  FRO92469       835          2.68%
[  15] DAI62779  ELE32164       832          2.68%
[  16] DAI75645  GRO73461       712          2.29%
[  17] DAI43223  ELE32164       711          2.29%
[  18] DAI62779  GRO30386       709          2.28%
[  19] ELE17451  FRO40251       697          2.24%
[  20] DAI85309  ELE99737       659          2.12%
[  21] DAI62779  ELE26917       650          2.09%
[  22] GRO21487  GRO73461       631          2.03%
[  23] DAI62779  SNA45677       604          1.94%
[  24] ELE17451  SNA80324       597          1.92%
[  25] DAI62779  GRO71621       595          1.91%
[  26] DAI62779  SNA55762       593          1.91%
[  27] DAI62779  DAI83733       586          1.88%
[  28] ELE17451  GRO73461       580          1.86%
[  29] GRO73461  SNA80324       562          1.81%
[  30] DAI62779  GRO59710       561          1.80%
[  31] DAI62779  FRO80039       550          1.77%
[  32] DAI75645  ELE17451       547          1.76%
[  33] DAI62779  SNA93860       537          1.73%
[  34] DAI55148  DAI62779       526          1.69%
[  35] DAI43223  GRO59710       512          1.65%
[  36] ELE17451  ELE32164       511          1.64%
[  37] DAI62779  SNA18336       506          1.63%
[  38] ELE32164  GRO73461       486          1.56%
[  39] DAI62779  FRO78087       482          1.55%
[  40] DAI85309  ELE17451       482          1.55%
[  41] DAI62779  GRO94758       479          1.54%
[  42] DAI62779  GRO21487       471          1.51%
[  43] GRO85051  SNA80324       471          1.51%
[  44] ELE17451  GRO30386       468          1.50%
[  45] FRO85978  SNA95666       463          1.49%
[  46] DAI62779  FRO19221       462          1.49%
[  47] DAI62779  GRO46854       461          1.48%
[  48] DAI43223  DAI62779       459          1.48%
[  49] ELE92920  SNA18336       455          1.46%
[  50] DAI88079  FRO40251       446          1.43%
```

Report the compute time for the Stripes job.

The 1st job (counts) reports the following compute times:

```
real    0m44.977s
user    0m5.111s
sys     0m0.337s
```

The 2nd job (sorts) reports the following compute times:

```
real    0m34.390s
user    0m5.735s
sys     0m0.335s
```

Describe the computational setup used (E.g., single computer; dual core; linux, number of mappers, number of reducers)

Cloudera QuickStart VM: single computer, 2 processors, 2 mappers (default), 1 reducer

How many times is each mapper and reducer called?

- Mapper: 2
- Reducer: 1

Discuss the differences in these counts between the Pairs and Stripes jobs

Below is a table showing the timings for the pairs and stripes jobs (1st job counts the co-occurrences, 2nd job sorts pairs):

item	Pairs count	Stripes count	Pairs sort	Stripes sort
real	0m50.284s	0m44.977s	0m26.758s	0m34.390s
user	0m5.474s	0m5.111s	0m4.982s	0m5.735s
sys	0m0.365s	0m0.337s	0m0.345s	0m0.335s

Indeed, the stripes job took less time to complete than the pairs job in the counting phase.

However, the pairs job took less time to complete when attempting to sort the pairs. This is likely due to the fact that in order to sort the output of the stripes job, we need to "unpack" the stripes to recover each individual pair, whereas the output of the pairs job does not need any additional unpacking.

Optional problems

HW3.7. Shopping Cart Analysis

Product Recommendations: The action or practice of selling additional products or services to existing customers is called cross-selling. Giving product recommendation is one of the examples of cross-selling that are frequently used by online retailers. One simple method to give product recommendations is to recommend products that are frequently browsed together by the customers.

Suppose we want to recommend new products to the customer based on the products they have already browsed on the online website. Write a program using the A-priori algorithm to find products which are frequently browsed together. Fix the support to s = 100 (i.e. product sets need to occur together at least 100 times to be considered frequent) and find itemsets of size 2 and 3.

In [1]:

```
%%writefile mapper_37.py
#!/usr/bin/env python
import itertools
import sys

item_count = int(sys.argv[1])
valid_items = set()

# If our item count is greater than 1, then load the corresponding model
file
# indicating the items we should care about.

if item_count > 1:
    model_id = str(item_count - 1)

    # The first k items in each model row will correspond to the product
s. We
    # can build up the set of valid items simply by iterating over the m
odel
    # and adding each of the elements in the first k columns.

    with open('apriori_model_' + model_id + '.txt') as model_file:
        for line in model_file:
            model_row = line.strip().split('\t')
            old_itemset = model_row[0:item_count - 1]
            valid_items.update(old_itemset)

"""
Emit all the itemsets for this basket.
"""
def emit_itemsets(basket):
    # First, we need to find out which items in the basket match the ite
ms
    # which we can accept in our k-itemsets. Note that we will accept ev
ery
    # item when the item count is 1.

    matching_items = []

    for item in basket:
        if item_count == 1 or item in valid_items:
            matching_items.append(item)

    # If we don't have enough items, we have no itemsets to emit.

    if len(matching_items) < item_count:
        return

    # Otherwise, emit all possible subsets. We'll use the pairs approach
to
    # make things easier to read.

    for itemset in itertools.permutations(matching_items, item_count):
        print '\t'.join(itemset) + '\t1'

    # Also emit a counter for subcombinations so that we can create a
    # tally to use for computing confidence.

    if item_count > 1:
        for sub_itemset in itertools.permutations(matching_items, item_c
ount - 1):
            print '\t'.join(sub_itemset) + '\t*\t1'

    # Finally, counter so that we can track the number of matching baske
ts.

    print ('*\t' * item_count) + str(1)

# Iterate over the baskets and emit the itemsets for each basket.

for line in sys.stdin:
    basket = line.strip().split(' ')
    emit_itemsets(basket)
```

```
Writing mapper_37.py
```

In [2]:

```
%%writefile reducer_37.py
#!/usr/bin/env python
import sys

support_threshold = int(sys.argv[1])

basket_count = 0
confidence_count = 0

current_itemset = None
current_count = 0

"""
Emit the current itemset and its count if they exceed support_threshold.
"""
def emit_count():

    # Declare that we want to use the global basket_count and confidence
_count
    # variables rather than something local to the function.

    global basket_count, confidence_count

    # Check if we haven't started counting anything yet.

    if current_itemset is None:
        return

    # Check if we are computing the basket count from the sort operatio
n.

    if current_itemset[0] == '*':
        basket_count = current_count
        return

    if current_itemset[-1] == '*':
        confidence_count = current_count
        return

    # Check if we have exceeded the necessary threshold.

    if current_count >= support_threshold:
        frequency = 1.0 * current_count / basket_count

        itemset_stats = str(current_count) + '\t' + str(frequency)

        if len(current_itemset) > 1:
            confidence = 1.0 * current_count / confidence_count
            itemset_stats += '\t' + str(confidence)

        print '\t'.join(current_itemset) + '\t' + itemset_stats

for line in sys.stdin:
    # Each line corresponds to the itemset stats. The last item will be
a count
    # value, while the first items will be the itemset.

    itemset_stats = line.strip().split('\t')

    itemset = itemset_stats[0:-1]
    count = int(itemset_stats[-1])

    # If we haven't switched itemsets, continue accumulating the counte
r.

    if current_itemset == itemset:
        current_count += count
        continue

    # If we have switched itemsets, emit the count for the old itemset a
nd then
    # switch to the new itemset.

    emit_count()
    current_itemset = itemset
    current_count = count

# We are guaranteed to not have printed the very last itemset, so emit i
t now.

emit_count()
```

```
Writing reducer_37.py
```

Test with the sample data from the slides.

In [1]:
```
!echo Beer Diaper BabyPowder Bread Umbrella > SampleSlidesData.txt
!echo Diaper BabyPowder >> SampleSlidesData.txt
!echo Beer Diaper Milk >> SampleSlidesData.txt
!echo Diaper Beer Detergent >> SampleSlidesData.txt
!echo Beer Milk CocaCola >> SampleSlidesData.txt
```

In [2]:
```
!cat SampleSlidesData.txt
```

```
Beer Diaper BabyPowder Bread Umbrella
Diaper BabyPowder
Beer Diaper Milk
Diaper Beer Detergent
Beer Milk CocaCola
```

```
In [3]: !cat SampleSlidesData.txt | python mapper_37.py 1 | sort -k1 | python re
        ducer_37.py 2 | tee apriori_model_1.txt
```

```
BabyPowder      2       0.4
Beer    4       0.8
Diaper  4       0.8
Milk    2       0.4
```

```
In [4]: !cat SampleSlidesData.txt | python mapper_37.py 2 | sort -k1 -k2 | pytho
        n reducer_37.py 2 | tee apriori_model_2.txt
```

```
BabyPowder      Diaper  2       0.4     1.0
Beer    Diaper  3       0.6     0.75
Beer    Milk    2       0.4     0.5
Diaper  BabyPowder      2       0.4     0.5
Diaper  Beer    3       0.6     0.75
Milk    Beer    2       0.4     1.0
```

Test with the actual data.

```
In [5]: !cat ProductPurchaseData.txt | python mapper_37.py 1 | sort -k1 | python
        reducer_37.py 100 > apriori_model_1.txt
```

```
In [6]: !cat ProductPurchaseData.txt | python mapper_37.py 2 | sort -k1 -k2 | py
        thon reducer_37.py 100 > apriori_model_2.txt
```

```
In [7]: !cat ProductPurchaseData.txt | python mapper_37.py 3 | sort -k1 -k2 -k3
        | python reducer_37.py 100 > apriori_model_3.txt
```

Test using Hadoop

```
In [15]: # Change permissions on mapper and reducer
         !chmod +x mapper_37.py
         !chmod +x reducer_37.py

         # If output folder already exists, delete it
         !hdfs dfs -rm -r /user/miki/week03/aprior_model_1

         # Run job
         !hadoop jar hadoop-streaming-2.7.1.jar \
         -D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFi
         eldBasedComparator \
         -D mapred.text.key.partitioner.options=-k1 \
         -D stream.num.map.output.key.fields=3 \
         -D mapred.text.key.comparator.options='-k1 -k2' \
         -D mapred.reduce.tasks=1 \
         -mapper '/home/cloudera/Documents/W261-Fall2016/Week03/mapper_37.py 1' \
         -reducer '/home/cloudera/Documents/W261-Fall2016/Week03/reducer_37.py 10
         0' \
         -input /user/miki/week03/ProductPurchaseData.txt \
         -output /user/miki/week03/apriori_model_1
```

```
Deleted /user/miki/week03/hw3_7_output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/
streamjob2467148671885451279.jar tmpDir=null
```

In [25]:
```
# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/apriori_model_2

# Run job
!hadoop jar hadoop-streaming-2.7.1.jar \
-D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFi
eldBasedComparator \
-D mapred.text.key.partitioner.options=-k1 \
-D stream.num.map.output.key.fields=3 \
-D mapred.text.key.comparator.options='-k1 -k2' \
-D mapred.reduce.tasks=1 \
-mapper '/home/cloudera/Documents/W261-Fall2016/Week03/mapper_37.py 2' \
-reducer '/home/cloudera/Documents/W261-Fall2016/Week03/reducer_37.py 10
0' \
-file /home/cloudera/Documents/W261-Fall2016/Week03/apriori_model_1.txt
\
-input /user/miki/week03/ProductPurchaseData.txt \
-output /user/miki/week03/apriori_model_2
```

```
rm: `/user/miki/week03/apriori_model_2': No such file or directory
packageJobJar: [/home/cloudera/Documents/W261-Fall2016/Week03/apriori_m
odel_1.txt] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/stream
job8901592178783306 82.jar tmpDir=null
```

In [26]:
```
# If output folder already exists, delete it
!hdfs dfs -rm -r /user/miki/week03/apriori_model_3

# Run job
!hadoop jar hadoop-streaming-2.7.1.jar \
-D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFi
eldBasedComparator \
-D mapred.text.key.partitioner.options=-k1 \
-D stream.num.map.output.key.fields=3 \
-D mapred.text.key.comparator.options='-k1 -k2 -k3' \
-D mapred.reduce.tasks=1 \
-mapper '/home/cloudera/Documents/W261-Fall2016/Week03/mapper_37.py 3' \
-reducer '/home/cloudera/Documents/W261-Fall2016/Week03/reducer_37.py 10
0' \
-file /home/cloudera/Documents/W261-Fall2016/Week03/apriori_model_2.txt
\
-input /user/miki/week03/ProductPurchaseData.txt \
-output /user/miki/week03/apriori_model_3
```

```
rm: `/user/miki/week03/apriori_model_3': No such file or directory
packageJobJar: [/home/cloudera/Documents/W261-Fall2016/Week03/apriori_m
odel_2.txt] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/stream
job3095742417588462696.jar tmpDir=null
```

```
In [28]: !hdfs dfs -cat /user/miki/week03/apriori model 3/part-00000 | head -n 20
```

```
DAI22896        DAI62779        GRO73461        101     0.0037998495109
1       0.340067340067
DAI22896        GRO73461        DAI62779        101     0.0037998495109
1       0.332236842105
DAI23334        DAI62779        ELE92920        143     0.0053799849510
9       0.52380952381
DAI23334        ELE92920        DAI62779        143     0.0053799849510
9       1.0
DAI31081        DAI62779        ELE17451        103     0.0038750940556
8       0.282967032967
DAI31081        DAI75645        FRO40251        122     0.004589917231
0.592233009709
DAI31081        ELE17451        DAI62779        103     0.0038750940556
8       0.449781659389
DAI31081        ELE32164        GRO59710        112     0.0042136945071
5       0.358974358974
DAI31081        FRO40251        DAI75645        122     0.004589917231
0.435714285714
DAI31081        FRO40251        GRO85051        102     0.0038374717833
0.364285714286
DAI31081        FRO40251        SNA80324        103     0.0038750940556
8       0.367857142857
DAI31081        GRO59710        ELE32164        112     0.0042136945071
5       0.598930481283
DAI31081        GRO85051        FRO40251        102     0.0038374717833
1.0
DAI31081        SNA80324        FRO40251        103     0.0038750940556
8       0.544973544974
DAI42083        DAI62779        DAI92600        105     0.0039503386004
5       0.905172413793
DAI42083        DAI92600        DAI62779        105     0.0039503386004
5       0.415019762846
DAI42083        DAI92600        ELE17451        117     0.0044018058690
7       0.462450592885
DAI42083        ELE17451        DAI92600        117     0.0044018058690
7       0.632432432432
DAI42493        DAI62779        ELE17451        112     0.0042136945071
5       0.363636363636
DAI42493        DAI62779        ELE92920        112     0.0042136945071
5       0.363636363636
cat: Unable to write to output stream.
```