Exercises
Data Mining: Learning from Large Data Sets
FS 2016

Series 1, Oct 6th, 2016 (MapReduce)

LAS Group, Institute for Machine Learning
Dept. of Computer Science, ETH Zürich
Prof. Dr. Andreas Krause
Web: `http://las.ethz.ch/teaching/dm-f16`
Email questions to:
Mario Lucic, lucic@inf.ethz.ch

Problem 1 (Approximation of the English dictionary):

In this exercise you are asked to construct an approximation of the English dictionary using a number of books written in English. The goal is to obtain a sorted list of words with their counts. We also want to make sure that all words are lower-cased and contain only letters from a-z. For example, if the only provided book contains the text "This is a (very, very) short 'book'. It is only 2 sentences long.", the output should be:

word	count
a	1
book	1
is	2
it	1
long	1
only	1
sentences	1
short	1
this	1
very	2

Your task is to modify the Word Count MapReduce example shown in the recitation session to incorporate the constraints discussed above.

Solution 1:

The mapper extracts the words and emits them. The reducer aggregates the counts.

solution1.py

```
import re
pattern = re.compile(r'[a-zA-Z]+')

def map(key, value):
    # key: book id
    # value: book content as a string
    for match in pattern.finditer(value):
        word = match.group(0).lower()
        yield word, 1

def reduce(key, values):
    yield current_word, sum(values)
```

Problem 2 (A basic English dictionary):

For some Natural Language Processing tasks you have to pre-process the data set by removing the most common words (stopwords). For the English language some examples are "the", "and", "if", "which", and "on". Your second task is to construct a dictionary such that the following constraints are met:

- There are at most 30 words for each letter.
- Each word in the dictionary has appeared at least **A** times, and at most **B** times in the data set , for some predefined **A** and **B**.
- For each letter the words are sorted alphabetically.

For example, if the subset of the output of the first exercise had been

word	count
a	788
all	123
antenna	9
auto	33
ball	15
beach	30
by	211

then for $\mathbf{A} = 10, \mathbf{B} = 35$ the final output of your MapReduce program should be

word	count
auto	33
ball	15
beach	30

You task is to write a `map` function and a `reduce` function in Python to solve this problem.

Solution 2:

The mapper runs on the output of Problem 1 and emits key-value pairs where the key is the first character of the word and the value is the count. This is done only for the words for which the count is in the interval $[A, B]$. The reducer then sorts the words alphabetically and emits the top 30 words for each letter. [1]

solution2.py

```python
def map(key, value):
    # key: word
    # value: total count of the word.
    if value >= A and value <= B:
        yield key[0], key

def reduce(key, values):
    # key: initial letter of a word
    # values: words with counts in [A, B] with initial letter given by key
    top30 = sorted(values)[:30]
    for word in top30:
        yield word
```

[1]This can be done in $\mathcal{O}(n \log k)$ instead of $\mathcal{O}(n \log n)$ by using some ordered set data structure (k is the number of words we wish to preserve).