University of Augsburg, Institute of Computer Science
Prof. Dr. P. Fischer
J. Kastner, L. Rudenko

SS 2019
03. May 2019
Sheet 2

Analyzing Massive Data Sets

Note: In order to avoid the complexity of setting up an actual Map-Reduce environment for the exercises, but still have consistent answers, we expect you to work with the following function signatures in Python:

```
def map(key, val): # key, val are each a single value
        return [] # A List of 2-values tuples [(k1,v1),(k2,v2),...]
def reduce (key, val): # key single value, val a list
         return [] # A List of 2-values tuples [(k1,v1),(k2,v2),...]
```

As an example, here is the word count from the lecture:

```
def map (key,val):
    res = []
    words = val.split("␣") # Split on whitespace
    for w in words:
        res.append((w,1))
    return res

def reduce (key,val):
    res = []
    res.append((key,len(val)))
    return res
```

The framework performing the invocations and the processing does not have to be modelled.

Exercise 1: MapReduce - Find and List Duplicate Files (homework)

Assume you are given a list of **[md5hash: string, filename: string]** pairs. How would you find the names of duplicate files where you schould report only **distinct file names**? Two files are duplicate if their md5hash values are equal.

Possible example for the execution:
given list: $[123, Name1], [123, Name2], [456, Name1]$
result: $[123 : [Name1, Name2]]$
file $[456, Name1]$ does not have any duplicates and is not in the result set.

Provide the Python code for map and reduce functions to find and list the duplicate files.

Exercise 2: MapReduce - Vector Length (live)

A measurement method records the surface of objects as a set of $3D$ vectors and stores them in the database table $S(\underline{vectorID}, \underline{dimension}, value)$. Due to measurement problems, not all vectors are completely recorded so that some dimensions are missing. The task is to compute the vector lengths *length* of all **fully captured** vectors: $length = \sqrt{(x^2 + y^2 + z^2)}$.

a) Use Python to write *map* and *reduce* functions to create a table $T(\underline{vectorId}, length)$ in which vector *length* of **each fully captured** vector is stored.

b) The table below shows a distribution of S to three MapReduce nodes. For this example, show the output of the two (map/reduce) phases of the MapReduce job on each of the three nodes.

Node 1		
id	*dim*	*val*
1	x	2
2	z	4
1	y	3
3	y	2
4	y	4

Node 2		
id	*dim*	*val*
3	z	4
4	z	2
2	x	2
2	y	4

Node 3		
id	*dim*	*val*
5	y	3
3	x	4
4	x	4

Exercise 3: MapReduce - Find the Common Friends (live)

Facebook has a feature which lists common friends with a person when you visit his or her profile page. We want to list the common friends for each pair of persons. We would like to implement this feature using MapReduce. Assume that you are given a file which consists of millions of lines in the following format (adjancency list):

$(PersonA, [PersonB, PersonC, PersonD, ...])$
...
where each line lists a person's name followed by list of his/her friends. Given this friendship list file, please provide the Python code for map and reduce functions for finding common friends among all pairs of persons listed in the file. Two possible cases should be considered - for now individually:

- The two users are connected, so they form a "triangle" with each common friend.
- The two users are not connected directly, but share common friends.