Spark RDD

Connection to Metacentrum cluster

```
ssh username@hador.ics.muni.cz
```

PySpark launching

```
pyspark --master yarn --num-executors 2 --executor-memory 4G
```

Python useful stuff

```
my_string = "to be Or NOT to be"
my_list = my_string.split()
print 'String length is:', len(my_string)
print 'Words in the string: ', my_string.split()
print 'Unique words in the string: ', set(my_string.split())
print 'String to lowercase: ', my_string.lower()
print 'Number of words in the string: ', len(my_list)
print 'First item in the list: ', my_list[0]
print 'Last item in the list: ', my_list[-1]
```

Example tryout

Try to run the Example 1 from the Spark RDD lecture.

```
# reading RDD
lines = sc.textFile("/user/pascepet/data/bible/bible.txt")

# line to words splitting
words = lines.flatMap(lambda line: line.split(" "))

# transformation to (key, value) pair
pairs = words.map(lambda word: (word, 1))

# summing 1s to every key
counts = pairs.reduceByKey(lambda a, b: a + b)

# look at result
counts.take(5)
```

Making the example better

Enrich the initial example:

2.1. The beginning of each line has to be ignored (each text line starts by book name and chapter:verse descriptor, then tab character followed by a text of the verse). So split the line by the tab and keep the second part only.

2.2. Convert all words to lowercase.

2.3. Get rid other than alphanumerical character from the text (replace them by an empty string. If you don’t know how to replace all non-alphanumerical characters at once, try to replace some of them one by one (e. g. comma, period, dash etc.).

2.4. Process only words not in stopwords (they are in the file `/user/pascepet/data/stopwords.txt` on HDFS).

Hint: stopwords can be got into set variable:

```
sw = set(sc.textFile('/user/pascepet/data/stopwords.txt').collect())
```

2.5. Cache some RDD.

2.6. Order words from the most often to the most rare (use sortBy method).

2.7. Find the longest word.

Text-mining

We will use the original text (not adjusted) in this section.

3.1. Find number of words for each verse (1 verse = 1 line) and find verses with biggest and least number of words.

3.2. Do the same computation as in 3.1 but take unique words only inside 1 verse.

Numeric computations

We will use hourly data of temperatures on USA weather stations (we used it for Hive training recently). Check if you have files *teplota1-6.csv* and *teplota7-12.csv* in some subdirectory of your user directory on HDFS.

CSV soubor is delimited by comma (','). It has header with field names, we need to remove them when processing data. Fields: *id_stanice, mesic, den, hodina, teplota, flag, latitude, longitude, vyska, stat, nazev*.

The temperature is in 10×°F. Some record have empty string on temperature place – it means “not available”.

Note: Spark method *textFile* can read all files in a directory to one RDD.

4.1. Look into few rows of data to understand them.

4.2. Find a state with the highest average temperature in months 6–8. Convert the result to °C.

4.3. For each month find the state with the highest average temperature in this month.