CS 4175 Fall 2023
Shirley Moore, Instructor

MapReduce, Hadoop, and Spark

Please refer to the following articles and YouTube videos for information about MapReduce, Hadoop, and Spark:

Big Data with PySpark
https://nyu-cds.github.io/python-bigdata/01-introduction/
https://nyu-cds.github.io/python-bigdata/02-mapreduce/
https://nyu-cds.github.io/python-bigdata/03-spark/

Big Data and Machine Learning
https://www.youtube.com/watch?v=qYONJbXNvK0&list=PLdkRteUOw2X-YKqommnuGWqNfEEUG6P2E Videos 1 and 2

Please refer to the Software section of the PSC Bridges-2 User Guide for information about using Spark on Bridges-2.

We will watch selected portions of the Big Data and Machine Learning youtube videos during class:

Video 1. Brief History of Big Data
33:00-end Hadoop vs. Spark

Video 2. Intro to Spark
0:00-1:00
2:50-7:00
8:00-10:00 Spark context
15:00-17:45 Transformations
19:15-21:00 Actions
26:15-26:45 Transformations vs. Actions
28:00-29:30 Pair RDD Transformations (key-value)
31:45
32:30-35:00 Shakespeare exercise
57:00

Now let's look at the MapReduce example explained in https://nyu-cds.github.io/python-bigdata/02-mapreduce/ .

Now let's use Spark on Bridges-2 to do the Shakespeare exercise. You can find the data file Complete_Shakespeare.txt in our shared project directory /ocean/projects/cis230018p/shared.

1) Count the number of lines
2) Count the number of words
3) Count the number of distinct words

4) Count the occurrence of each word
5) Show the top 5 most frequent words

```
$ interact -p GPU-shared --gres=gpu:v100-32:1 -t 30:00
$ module load spark
$ pyspark
In [1]: lines_rdd = sc.textFile("Complete_Shakespeare.txt")

In [2]: lines_rdd.count()
[Stage 0:>                                                          Out[2]: 196386

In [3]: words_rdd = lines_rdd.flatMap(lambda x: x.split())

In [4]: words_rdd.count()
Out[4]: 966501

In [5]: words_rdd.distinct().count()
[Stage 2:>                                                          Out[5]: 71594

In [6]: key_value_rdd = words_rdd.map(lambda x: (x,1))

In [7]: key_value_rdd.take(5)
Out[7]: [('The', 1), ('Project', 1), ('Gutenberg', 1), ('eBook', 1), ('of', 1)]

In [8]: word_counts_rdd = key_value_rdd.reduceByKey(lambda x,y: x+y)

In [10]: word_counts_rdd.take(5)
Out[10]:
[('The', 4631),
 ('Project', 79),
 ('of', 17111),
 ('Shakespeare', 5),
 ('ebook', 2)]
In [11]: flipped_rdd = word_counts_rdd.map(lambda x: (x[1],x[0]
    ...: ))

In [12]: flipped_rdd.take(5)
Out[12]:
[(4631, 'The'),
 (79, 'Project'),
 (17111, 'of'),
 (5, 'Shakespeare'),
 (2, 'ebook')]
```

```
In[13]: results_rdd = flipped_rdd.sortByKey(False)

In [14]: results_rdd.take(5)
Out[14]: [(25689, 'the'), (20717, 'I'), (19849, 'and'), (17111, 'of'), (17075, 'to')]
```