sbajoria 50208005
vidhijat 50207090

LAB 5: PROGRAMMING THE DATAFLOW FOR BIG DATA ANALYTICS USING APACHE SPARK

Saurabh Bajoria 50208005
Vidhi Shah 50207090

sbajoria 50208005
vidhijat 50207090

Preparation:

Apache Spark Environment: Jupyter with Python (Pyspark)

Downloaded required spark-2.1.1-bin-hadoop2.7 package.

1. Understand Apache Spark with Titanic data analysis

Ran the notebook Welcome to Spark with Python in the Jupyter environment. Please find the notebook with all the outputs along with the Lab submission.

2. Featured activity: Analysis of Latin documents for word-co-occurrence

a. Bigram

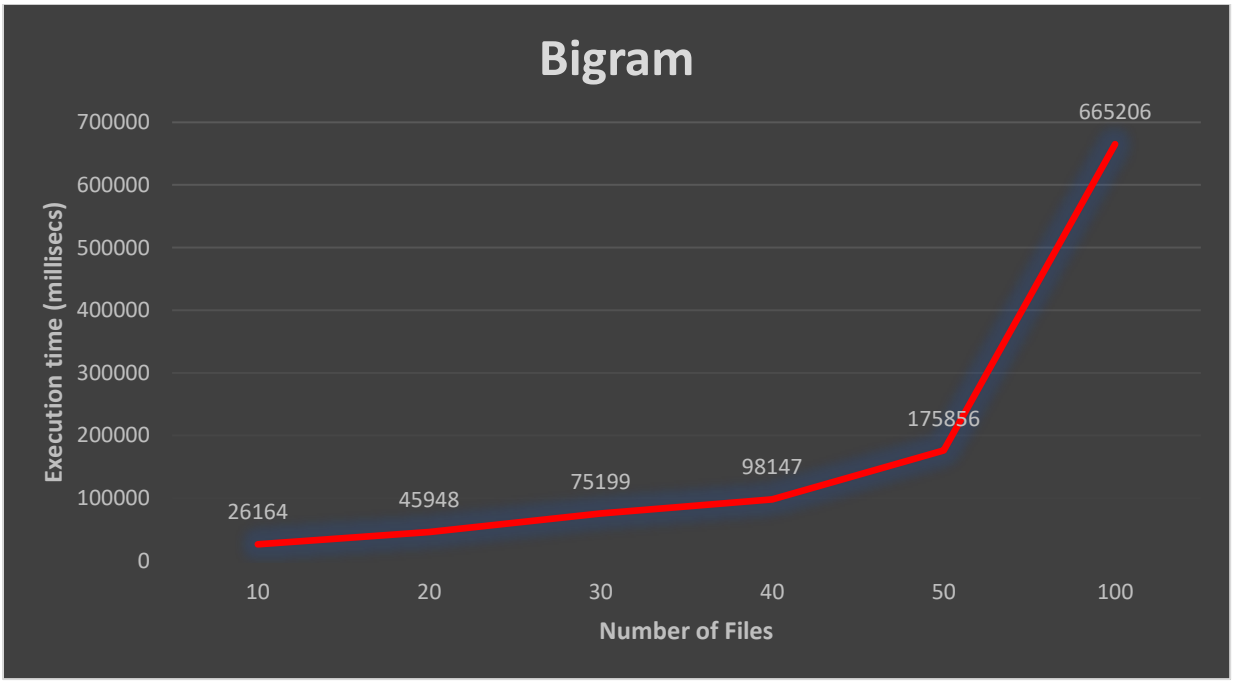
Output Format:

If two words Word1 and Word2 are neighbors and have the following lemmas, the output format is as below:

Word1 → Lemma1 and Lemma2
Word2 → Lemma3 and Lemma4

n-gram (n=2)	Location
Word1 Word2	<Location1><Location2>
Lemma1 Lemma3	<Location1><Location2>
Lemma2 Lemma3	<Location1><Location2>
Lemma1 Lemma4	<Location1><Location2>
Lemma1 Lemma4	<Location1><Location2>

Plot for Number of files Vs Execution Time:



- Seen above is the execution time vs Number of files for Word Cooccurrence-Bigram on Classical Latin text using lemmatization in **Spark**.
- The execution time increases with increase in the number of input files
- Also, we can see that the overall execution time is lesser for Bigrams compared to the Trigrams discussed below.

b. Trigram

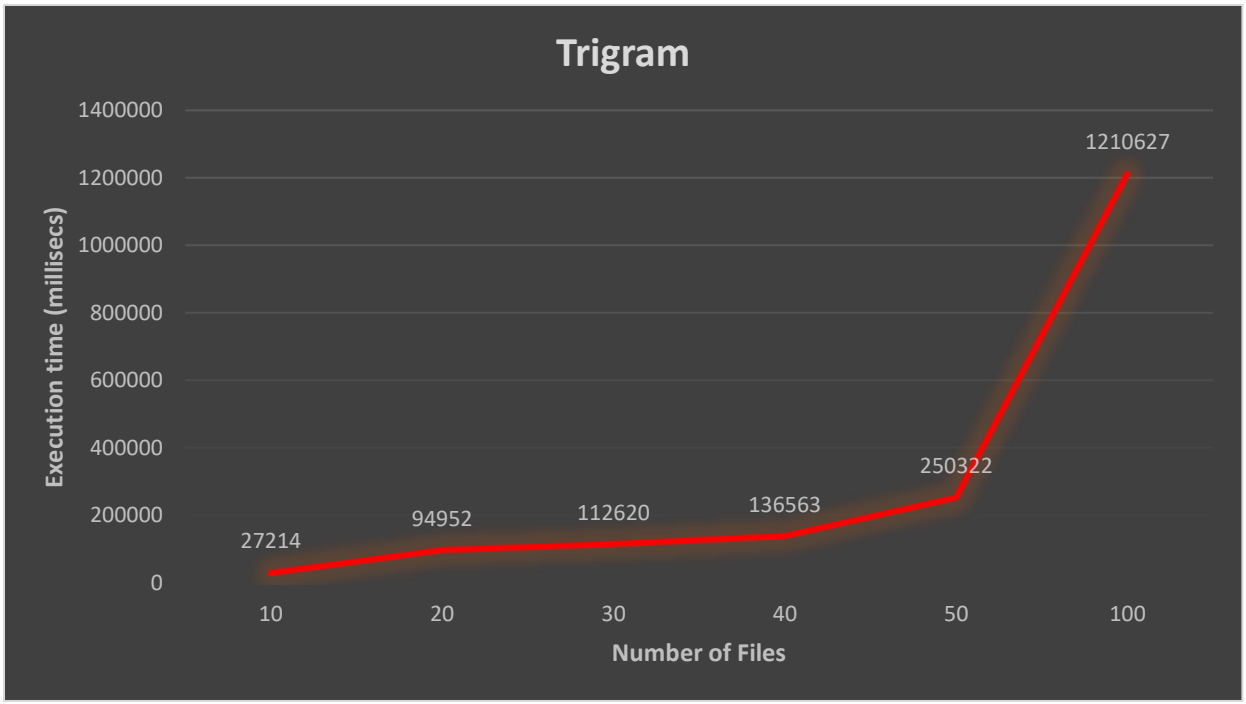
Output Format:

If two words Word1 and Word2 are neighbors and have the following lemmas, the output format is as below:

Word1 → Lemma1 and Lemma2
Word2 → Lemma3 and Lemma4
Word3 → Lemma5 and Lemma6

n-gram (n=3)	Location
Word1 Word2 Word3	<Location1><Location2>
Lemma1 Lemma3 Lemma5	<Location1><Location2>
Lemma2 Lemma3 Lemma5	<Location1><Location2>
Lemma1 Lemma4 Lemma5	<Location1><Location2>
Lemma1 Lemma4 Lemma6	<Location1><Location2>
Lemma2 Lemma4 Lemma5	<Location1><Location2>
Lemma2 Lemma4 Lemma6	<Location1><Location2>
Lemma1 Lemma3 Lemma6	<Location1><Location2>
Lemma2 Lemma3 Lemma6	<Location1><Location2>

Plot for Number of files Vs Execution Time:



- Seen above is the execution time vs Number of files for Word Cooccurrence-Trigram on Classical Latin text using lemmatization in **Spark**.
- As with Bigrams, the execution time increases with increase in the number of input files.
- Also, we can see that the overall execution time is greater for Trigrams compared to the Bigrams as more time is spent for calculating the two neighbors for each word.
- Also, as the number of input files increases, the execution time shoots up to a large number. This is as expected, because more number of files increases the computation drastically.

Conclusion:

- The execution time remains almost constant if the output of the reducers is not saved in a text file, i.e. most of the time is spent in writing the output obtained to the memory.
- Also, with increase in number of files at the input (Mapper), the number of output files obtained also increases, i.e. the output is split into n number files equaling the number of reducers used.
- The overall time taken for execution in Spark is very less compared to that in Hadoop.
- Also, with increase in number of files the execution does not increase exponentially like in Hadoop.