Q1. Implement a Map Reduce program for counting the number of lines in a document.

Result Figures:

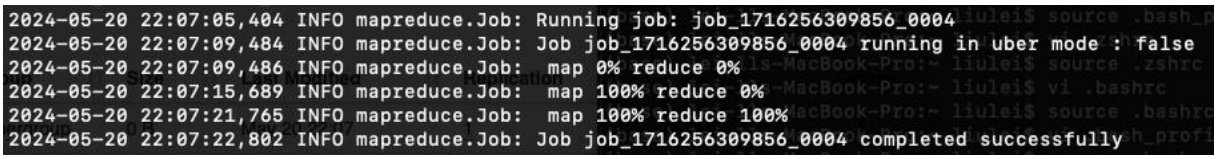

Figure 1

Figure1 shows the MapReduce job completed successfully.

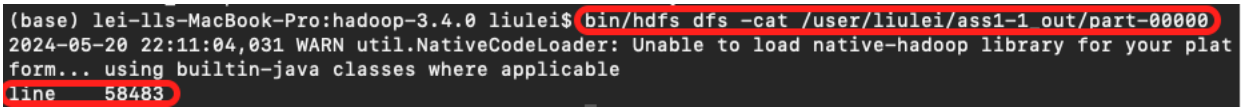

Figure 2

Figure 2 shows the final result: our total line number is 58483.
Please check mapper.py and reducer.py along with the submission. The process for all the commands to create the result can also be found in cmd.txt.

Q2. Apply K-means clustering on Map Reduce using k = 5 and k = 8 clusters on the given dataset, list the cluster labels or centroids, the number of iterations for convergence or use maximum iterations = 15 and time/duration.

For k = 5

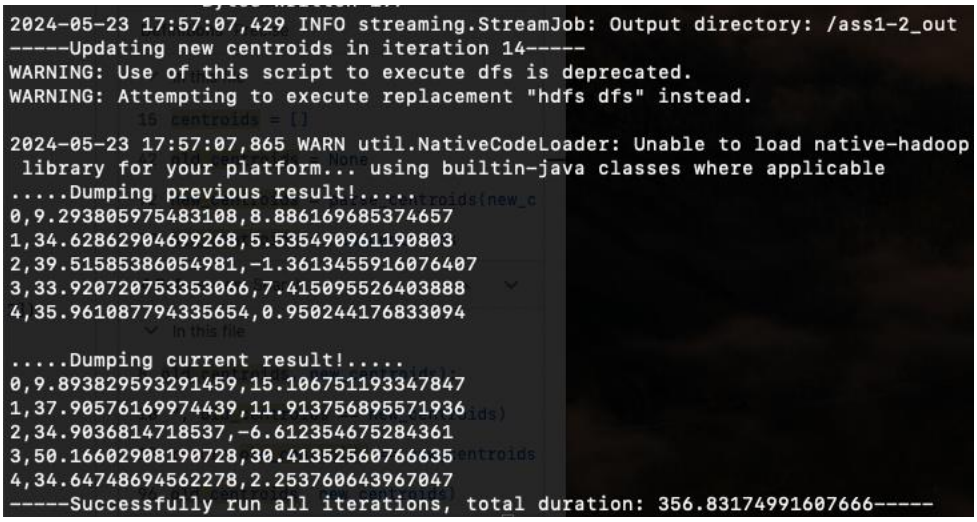

Figure 3

Figure 3 shows Mapreduce completed after 15 iterations with total duration time around 356.8 seconds.

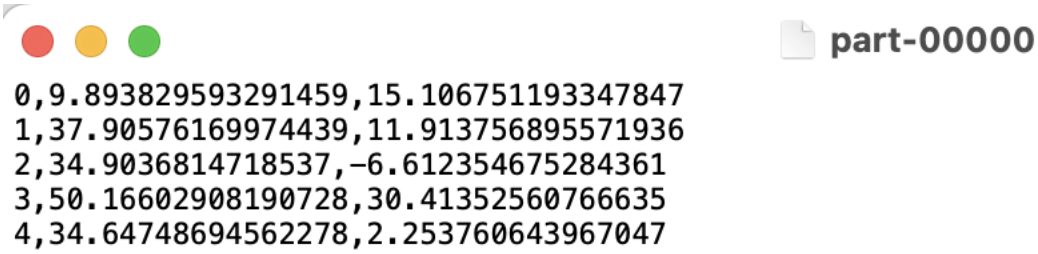

Figure 4

Figure 4 shows the final result for k = 5.

For k = 8

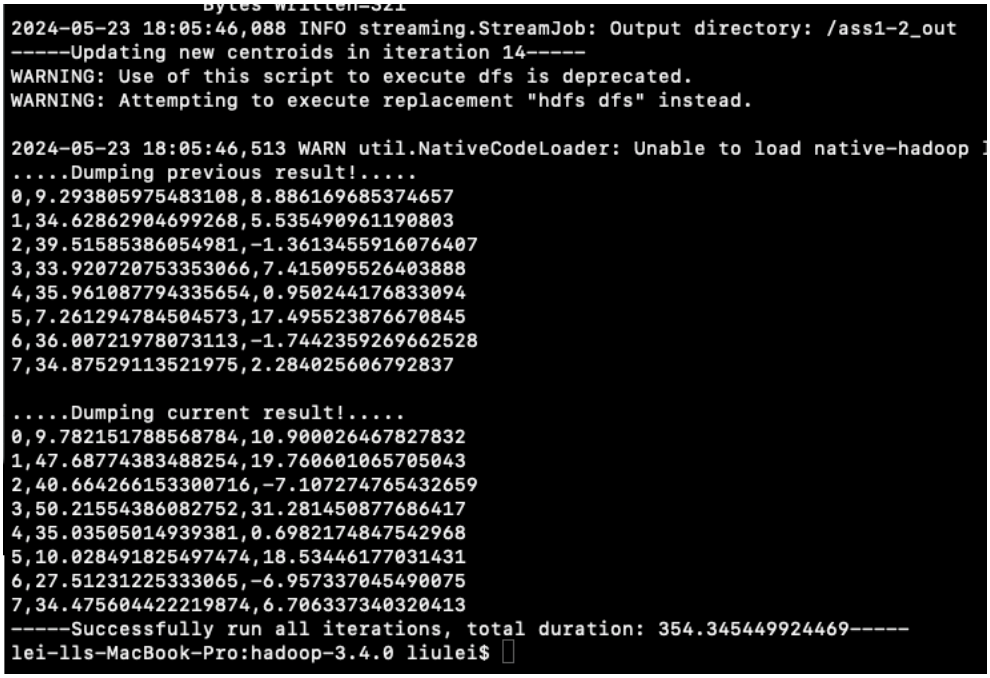

Figure 5

Figure 5 shows Mapreduce completed after 15 iterations with total duration time around 354.3 seconds.

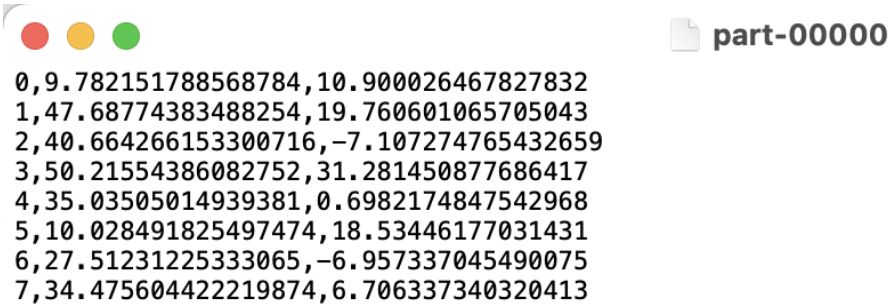

Figure 6

Figure 6 shows the final result for k = 8.
Please check automation.py, mapper.py and reducer.py along with the submission.
Automation.py serves as the script to automate running Mapreduce for 15 iterations, the Hadoop commands are included in this automation script.

Q3. Explain the advantages and disadvantages of using K-Means Clustering with MapReduce.

Advantages:

1. Parallelization to speed up the computation
   From lecture, we know that MapReduce can be operated by increasing the servers horizontally. Each server in the cluster can compute an independent subprocesses. This can significantly speed up the computation process and benefits more if using K-Means in a very large data set. You can even further pipeline the subprocess to accelerate the computation.
2. Flexibility in processing the data
   By performing K-Means Clustering with MapReduce in parallel, it provides flexibility to handle the data during the computation process. For example, you can easily remove, modify a subprocess within MapReduce, or extract a subprocess and integrate it with other computations.

Disadvantages:

1. Complexity in data overhead
   It could be a headache to find the correct format to input data into the MapReduce. Data should be properly formatted before inputting into the Mapper function and the Reduce function in order to perform the K-Means Clustering computation. The designer needs to understand the concept of both K-means Clustering and the concept of the MapReduce.
2. No intermediate result
   MapReduce is designed to process a series of batches. Therefore, it could be a bit hard to access the intermediate data result or some real-time analysis result.

Q4. Can we reduce the number of distance comparisons by applying the Canopy Selection? Which distance metric should we use for the canopy clustering and why?

Yes, we can reduce the number of distance comparisons. In Canopy Selection, we do not need to compare the points that are not within a same canopy, which saves a large amount of distance comparisons needed.
Since the canopies may overlap with each other and we only calculate the distance within a canopy, the overall algorithm may contain the tolerance to inaccuracy. We can choose inexpensive distance metric for canopy clustering as long as the clustering accuracy is still guaranteed. Some algorithms the paper mentioned are Greedy Agglomerative Clustering, K-means, and Expectation-Maximization.

Q5. Is it possible to apply Canopy Selection on MapReduce? If yes, then explain in words, how would you implement it

Yes, we can apply Canopy Selection on MapReduce.

Step 1: Create Canopies.

This can be created by having similarity metrics. Initialize different Canopies with different standards. (e.g. data share the same similarity 1 go to Canopy 1, data share the same similarity 2 go to Canopy 2, etc). For example, we can use distance threshold.

Step 2: Canopy clustering/map process

Assign all datapoints to different potential Canopies based on the threshold we set in Step 1. Key: to indicate which canopy being assigned, Value: to indicate which datapoints.

Step3: calculation within canopy/reducer process

Perform calculation within each canopy and then filter out the data points that is below the distance threshold. This is exactly what reducer aims: merge values to form a possibly smaller set of values.

Q6. Is it possible to combine the Canopy Selection with K-Means on MapReduce? If yes, then explain in words, how would you do that.

Yes. Combining with the implementation idea in Q5.

Step1: Create Canopies as in Q5

Step2: Canopy clustering as in Q5

Step3: K-means

Canopy clustering indicates which canopies(clusters) the datapoints belong. This can be treated as the input to the K-means. We perform K-means until the result converges.