LDSA Assignment-3 Part 2

- **Uvais Karni**

Section C:

C.1) The reason she did not get line split is because she used transformation flatMap instead of map, transformation flatMap takes each element and gives out one or more output (one to many transformation).

C.2) Immutable data are data that cannot be altered. The advantage of using immutable data in spark is that it removes the hassle of updating changes to the data as they distributed as partitions across cluster. It also helps with performance as it allows simplicity and high concurrency (utilisation of large no resources without the need to worry of issues arising due to read and writes). Immutability also allows for data to easily replicated and shared.

C.3) When collect is called the driver fetches the entire RDD to be loaded in its memory. If there is not enough memory to load RDD it will lead to insufficient memory in driver leading to condition out of memory. The default memory size allocated for driver is usually around 1 GB which is not much to store entire RDD.

C.4) RDD are resilient because they are fault tolerant which is possible as they utilise data lineage to reconstruct data that is corrupted or lost. Each RDD stores all the transformation it has undergone so that it can recovered as mentioned above as data lineage. The above process is possible even though its distributed environment because of another property of RDD which is immutability.

Section D:

The performance of the spark application can be improved by following:

- **Choosing the right data abstraction for the job at hand**: That is between DataFrames, RDDs etc. It makes a great deal in choosing the right data abstraction because each of them has their own set features like DataFrames are general preferred for most tasks because it has low garbage collection overhead, better optimisation through Catalyst and more benefits. RDDs are only supported by Spark legacy API.[1]
- **Choosing the right deployment mode:** There are two deployment mode client and cluster mode. It generally preferred to run client mode when the job machine is close to the spark infrastructure else cluster mode is used. The reason behind these decisions is to maintain low latency and therefore maintain high performance.[2]
- **Choosing right operator and sequence of operation:** Usually jobs with join or shuffle operation have a high overhead. So, it's always better to have the data filtered before running join or shuffle operation.
- **Efficient Use of Cache:** It is always better to cache data as it will improve performance if there is need to run multiple jobs on same data.

Apart from the advice above, you can run job on small subset of the data to test everything and then run job on the full data.

Reference:

[1] https://docs.microsoft.com/en-us/azure/hdinsight/spark/apache-spark-perf

[2] https://techvidvan.com/tutorials/spark-modes-of-deployment/