04BIG DATA COMPUTING 2019-20 – HOMEWORK 3 – GROUP 04

Required Tests (Java)

Dataset	K	L	num-executors	Init	T1	T2	AvgDist
Glove2M300d.txt	100	16	16	21690	17289	19242	29.04073384782192
Glove2M300d.txt	100	16	8	32978	29811	19350	29.04073384782192
Glove2M300d.txt	100	16	4	56114	46917	20881	29.04073384782192
Glove2M300d.txt	100	4	4	52307	46084	1173	29.029582748194713

Observations

We have run 4 experiments:

1) `.cache()` inside initialization + `.repartition(L)` inside Round1: only for this we weren't be able to run the 4th test due to errors 52 and 413 that cause Out of Memory.
2) `.cache()` at the end of Round 1 + `.repartition(L)` inside Round1: slower T1, faster and constant T2
3) **`.cache()` at the end of Round 1 + `.repartition(L)` inside initialization: slower Init, Faster T1 and constant T2.**
4) `.cache()` inside initialization + `.repartition(L)` inside initialization: slower T2.

The best runtimes are achieved by the 2nd and 3rd experiments. Between these two **the fastest is the 3rd** that we reported in the table above.

In particular from the results of the tests we can see that T1 increases as the number of executors decreases. This is to be expected since for round 1 we are partitioning the input and distributing it across multiple workers.

In round 2 instead we see a constant execution time, besides some minor variation. This is caused by the fact that we are running a sequential algorithm on the L*K points extracted from round 1.

We also report the other 3 experiments below as reference.

1) `.cache()` inside initialization + `.repartition(L)` inside Round1

Dataset	K	L	num-executors	Init	T1	T2	AvgDist
Glove2M300d.txt	100	16	16	15886	26700	36390	29.04073384782192
Glove2M300d.txt	100	16	8	22481	34677	42817	29.04073384782192
Glove2M300d.txt	100	16	4	41284	74432	65456	29.04073384782192
Glove2M300d.txt	100	4	4	*You will not be able to run this test*			

2) `.cache()` at the end of Round 1 + `.repartition(L)` inside Round1

Dataset	K	L	num-executors	Init	T1	T2	AvgDist
Glove2M300d.txt	100	16	16	14527	34806	19763	29.04073384782192
Glove2M300d.txt	100	16	8	21402	57782	19275	29.04073384782192
Glove2M300d.txt	100	16	4	31642	87786	19255	29.04073384782192
Glove2M300d.txt	100	4	4	31613	90222	1154	29.029582748194713

4) `.cache()` inside initialization + `.repartition(L)` inside initialization

Dataset	K	L	num-executors	Init	T1	T2	AvgDist
Glove2M300d.txt	100	16	16	23636	14044	43658	29.04073384782192
Glove2M300d.txt	100	16	8	35719	28239	53100	29.04073384782192
Glove2M300d.txt	100	16	4	56890	42189	59072	29.04073384782192
Glove2M300d.txt	100	4	4	56981	49107	48130	29.038756561329237