Big Data Computing - 4th Homework Report

Boem Davide, ID: 1176946, `davide.boem@studenti.unipd.it`
Boscaro Nicola, ID: 1181356, `nicola.boscaro.1@studenti.unipd.it`
Faccin Dario, ID: 1177736, `dario.faccin@studenti.unipd.it`[*]

In this report we've described the results obtained by running our code, on *Cloud Veneto*, that we've written for the 4th homework.

1 Results

In the Tables 1 - 6, you can see what we have obtained running our code on *Cloud Veneto*[1].

	Cores used by application	**Cores used for each executor**	**numBlocks**	**k**	**Coreset construction (ms)**	**Computation final solution (ms)**	**Average distance**	**Dataset (Approximate size)**
1	20	4	12	10	26519	24	10,5483	`all`
2	20	4	12	20	49519	40	9,9642	
3	20	4	12	30	222942	70	9,7861	
4	20	4	12	40	52159	155	9,6639	
5	20	4	12	50	23727	297	9,5462	
6	20	4	12	60	30821	449	9,4961	
7	20	4	12	70	25678	726	9,3317	
8	20	4	12	80	27404	1060	9,3204	
9	20	4	12	90	37561	1527	9,2578	
10	20	4	12	100	32148	2122	9,1879	
11	20	4	12	110	68265	2818	9,1743	
12	20	4	12	120	73367	3578	9,1061	
13	20	4	12	130	59042	4454	9,0946	
14	20	4	12	140	62462	5745	9,0689	
15	20	4	12	150	48574	8027	9,0431	
16	20	4	12	160	36865	12089	9,0115	
17	20	4	12	170	56385	11927	8,9946	
18	20	4	12	180	52730	12659	8,9458	
19	20	4	12	190	69069	14400	8,9466	
20	20	4	12	200	82165	16747	8,9113	

Table 1: Results obtained on *Cloud Veneto*, using dataset `vectors-50-all.txt.bz2` and changing k.

[*] Contact email

[1] We decided to round the average distances to 4 decimal places.

	Cores used by application	**Cores used for each executor**	**numBlocks**	**k**	**Coreset construction (ms)**	**Computation final solution (ms)**	**Average distance**	**Dataset (Approximate size)**
1	20	4	12	10	12636	43	10,2092	`2000000`
2	20	4	12	20	9732	103	9,5775	
3	20	4	12	30	14897	109	9,2798	
4	20	4	12	40	12534	172	9,0856	
5	20	4	12	50	14074	484	8,9951	
6	20	4	12	60	18627	455	8,9671	
7	20	4	12	70	23357	876	8,9588	
8	20	4	12	80	24393	1141	8,9301	
9	20	4	12	90	30550	1790	8,8520	
10	20	4	12	100	22076	2093	8,8300	

Table 2: Results obtained on *Cloud Veneto*, using dataset `vectors-50-2000000.txt.bz2` and changing k.

	Cores used by application	**Cores used for each executor**	**numBlocks**	**k**	**Coreset construction (ms)**	**Computation final solution (ms)**	**Average distance**	**Dataset (Approximate size)**
1	20	4	10	20	11781	49	10,1422	`all`
2	20	4	20	20	33848	56	10,0583	
3	20	4	30	20	32841	130	9,9066	
4	20	4	40	20	6576	198	9,9467	
5	20	4	50	20	29907	335	9,7920	
6	20	4	60	20	7354	510	9,7920	
7	20	4	70	20	8124	642	9,9569	
8	20	4	80	20	8689	814	9,8959	
9	20	4	90	20	6466	1082	9,9125	
10	20	4	100	20	6641	1235	9,9389	
11	20	4	110	20	7016	1631	9,7920	
12	20	4	120	20	6636	2041	9,7920	
13	20	4	130	20	7867	3309	9,9125	
14	20	4	140	20	7058	2955	9,7920	
15	20	4	150	20	6314	2992	9,7920	
16	20	4	160	20	9296	4145	9,7920	
17	20	4	170	20	8206	4552	9,7920	
18	20	4	180	20	6821	6880	9,7920	
19	20	4	190	20	8734	9796	9,7920	
20	20	4	200	20	8407	9755	9,7920	

Table 3: Results obtained on *Cloud Veneto*, using dataset `vectors-50-all.txt.bz2` and changing *numBlocks*.

	Cores used by application	**Cores used for each executor**	**numBlocks**	**k**	**Coreset construction (ms)**	**Computation final solution (ms)**	**Average distance**	**Dataset (Approximate size)**
1	20	4	10	20	5160	66	9,5939	`2000000`
2	20	4	20	20	3659	118	9,6041	
3	20	4	30	20	3598	224	9,5847	
4	20	4	40	20	3077	273	9,4245	
5	20	4	50	20	3033	448	9,5375	
6	20	4	60	20	3634	678	9,7332	
7	20	4	70	20	3129	909	9,4245	
8	20	4	80	20	3238	913	9,4245	
9	20	4	90	20	3097	1709	9,4245	
10	20	4	100	20	4176	1596	9,4245	

Table 4: Results obtained on *Cloud Veneto*, using dataset `vectors-50-2000000.txt.bz2` and changing *numBlocks*.

	Cores used by application	**Cores used for each executor**	**numBlocks**	**k**	**Coreset construction (ms)**	**Computation final solution (ms)**	**Average distance**	**Dataset (Approximate size)**
1	30	4	12	10	12951	34	10,5483	`all`
2	30	4	12	20	16580	42	10,1264	
3	30	4	12	30	18378	129	9,8571	
4	30	4	12	40	25320	181	9,6896	
5	30	4	12	50	24738	350	9,5100	
6	30	4	12	60	37329	465	9,4490	
7	30	4	12	70	40478	842	9,3427	
8	30	4	12	80	37355	1170	9,3359	
9	30	4	12	90	33338	1926	9,1828	
10	30	4	12	100	36045	2570	9,1724	

Table 5: Results obtained on *Cloud Veneto*, using dataset `vectors-50-all.txt.bz2` and changing X.

	Cores used by application	**Cores used for each executor**	**numBlocks**	**k**	**Coreset construction (ms)**	**Computation final solution (ms)**	**Average distance**	**Dataset (Approximate size)**
1	20	8	12	10	47584	54	10,6549	`all`
2	20	8	12	20	47944	48	10,1581	
3	20	8	12	30	60405	101	9,8100	
4	20	8	12	40	51350	146	9,7196	
5	20	8	12	50	50555	266	9,5379	
6	20	8	12	60	61062	735	9,4922	
7	20	8	12	70	52542	932	9,3342	
8	20	8	12	80	59816	1614	9,3254	
9	20	8	12	90	62616	2101	9,2225	
10	20	8	12	100	59043	2146	9,2218	

Table 6: Results obtained on *Cloud Veneto*, using dataset `vectors-50-all.txt.bz2` and changing Y.

2 Plots

3 Conclusions

Looking at the results (see Tables 1 - 6), we can see that, while the measured time for the *Construction of the coreset* stayed inside a time interval, the time spent by the program for the *Computation of the final solution* had risen proportionally with the values of k and of $numBlocks$.

On the contrary, the *Average distance* started with bigger values and converged after a while to lower values.

In the case of the total number of cores used by the application (X) we observed that there was a little improvement in the measured time of the *Construction of the coreset*, meanwhile, modifying the number of cores used for each executor (Y) we didn't see any enhancement.