Systems for Data-science M3

Gengler Damien 272798

June 2021

1 Baseline: Prediction based on Global Average Deviation

- Question 3.2.1 :

  With our new implementation we find that the MAE for k=100 is 0.7562 and the MAE for k=200 is 0.74845.

- Question 3.2.2 :

  Over 5 runs, we find the following statistics for the computation time of all k-nearest neighbours for k=100 (the one given in run args).

Statistics	time (in microseconds)
min	1565
max	2720
avg	2023
std	480.24

Table 1: Min, Max, Average and Standard deviations of the time to compute all 100-NN

- Question 3.2.3 :

  Over 5 runs, we find the following statistics for the computation time of the 20000 predictions of the testing set for k=100 (the one given in run args).

Statistics	time (in microseconds)
min	4776
max	6312
avg	5208
std	586.72

Table 2: Min, Max, Average and Standard deviations of the time to compute 20000 predictions

- Question 3.2.4 :

  In Milestone 2 we reported an average computation time for all similarities of 25624.4 ms. With our optimized implementation we get and average computation time of the kNN similarities of 2023ms. We thus get a speedup of :

  $$speedup = \frac{t_{old}}{t_{new}} = \frac{25624.4}{2023} = 12.66$$

  Hence we managed to obtain a speed up for almost x13.

  The main reason why it is so fast is because we don't iterate without thinking on all the data. We avoid running iterations on useless data and as the matrix is sparse, we save a lot of time doing so.

- Question 4.1.1 :

  Our Spark implementation running on the cluster on the 1M dataset yields a MAE of 0.7346.

- Question 4.1.2 :

Method	**knn**			**pred**		
	min	max	avg	min	max	avg
Non-Spark	252480	270317	262998	113410	122134	116598
Spark	220798	256769	235450.2	90706	103161	99723

Table 3: Min, Max and Average computation time for knn and prediction computation over 5 iteration with one executor on the cluster

  From the results in Table 3 it seems that while similar, the execution time of the Non-Spark version is slower than the one of the Spark version. This might also come from the fact that both algorithms are slighlty different of that several people were on the cluster at the same time. It might also come from the fact that my computer went into sleep mode during the computation.

- Question 4.1.3 :

  The times for computing knn and prediction times for different number of executors are displayed in Table 4.

  Figure 1 shows the average execution time w.r.t the number of executors used. We can see a huge speedup in the computations. The computation time decreases exponentially for both KNN and predictions.

- Question 5.1.1 :

  It takes 1716 days of renting the ICC.M7 machine to make buying it less expensive. That represents 4.7 years.

Nb executors	knn			pred		
	min	max	avg	min	max	avg
1	211946	235079	222503.3	90706	103161	95800.3
2	113236	138329	126947.3	48420	53494	51380.3
4	63353	71588	66698	28793	29317	29055
8	35882	37678	37001.6	14571	16913	15520
16	22380	23578	22889	9336	9647	9466.3

Table 4: Min, Max and Average computation time for knn and prediction computation with different number of executors (time in ms)

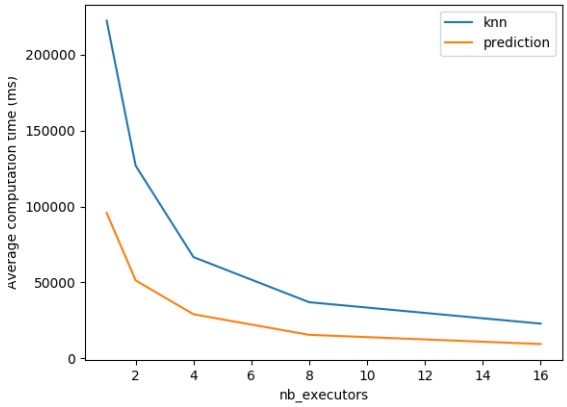

Figure 1: This is an image from a text that uses color to teach music.

- Question 5.1.2 :

  The daily cost of an IC container with the same amount of RAM and CPUs as the ICC.M7 machine is 20.896CHF/day.

  The ratio $\frac{ICC.M7}{Container}$ is 0.97. The container is not cheaper than the ICC.M7 cluster but the difference is negligible.

- Question 5.1.3 :

  The daily cost of an IC container with the same amount of RAM and CPUs as 4 Raspberry Pi 4 is 0.472CHF/day.

  The ratio $\frac{4RPis}{Container}$ is 0.4576 when running at max power and 0.091 when running at min power. The container is not cheaper than the 4RPis.

- Question 5.1.4 :

  After between 885 and 1482 days, the cost of renting a container is higher than buying and running 4 Raspberry Pis.

- Question 5.1.5 :

  For the same buying price as an ICC.M7, you can buy 369 Raspberry Pis. You would obtain a throughput 3.294 times higher with the RPis. You would obtain a total amount of RAM 1.921 times higher with the RPis.

- Question 5.1.6 :

  Assuming that we want to keep 50% of the memory available for other tasks, we can store 555555 users pers GB. Which means you can store 4444440 users per RPi and 853332480 users per ICC.M7.

- Question 5.1.7 : I think it really depends on your needs and how long you will use the hardware. If you need it for a short period of time using the cloud or renting is the preferable solution as it is easier to do and requires less effort. In case you know that you will use it for a long time (several years), or need extra privacy for your computations then you might want to invest in your own hardware.

  From here the two options of either buying and ICC.M7 machine or a given number of RPi4 are available. Once again it depends on your means and your needs. If you need a straight up simpler implementation that has great power and have the budget you should go for the ICC.M7 Machine. If you can afford setting up the whole cluster yourself, and need performances but don't know how much yet, then going for the Rpis is a good alternative. It allows you to have flexible Computing power, namely you can start with a given amount and buy more RPis to add to your distributed systems as you go. The good part about this is that this solution, for a similar investment as for ICC.M7, will give you higher throughput and RAM, also the RPis consome less energy overall. The problem is that you need to setup you whole architecture and maintain it which can be quite complicated and time consuming.

2 Notes

I think for this milestone as well as the previous ones I computed time in milliseconds rather than microseconds. As I did the same for all milestones and it didn't give me negative point in milestone 1, I kept it as is for M3 so that I could compare the time. I can anyway observe the same variations with both units so I guess it's not too important but I wanted to point it out, just in case.