Efficiency of KV read and write on different KV databases

Author: Zhoujie (Jason) Ding

Introduction

There are many kinds of KV databases to use for our kaas, and we are looking for the one that performs the best. In my benchmark, I set the shortest execution time for KV read and write as the metric. I currently want to compare the performance between Redis and Anna. In this report, I have three KV databases: Redis, Anna, and the local database acting like a dictionary. My inputs are different sizes of matrices, varying from 5 by 5 to 5000 by 5000 (explicitly, they are 5 by 5, 100 by 100, 1000 by 1000, and 5000 by 5000). In addition, different tests are performed using these matrices for different types of KV reads and writes in order to simulate the real-world situations.

Methods

I use the `time` function in python API to measure the duration of reading and/or writing the matrices in the KV database. I also use the `testenv` function in `utils.py` to invoke and close the Redis or Anna so that I reset the server in every test. Moreover, I repeat every test multiple times and record every time slot in a list. Then, I calculate the mean and the standard deviation of the time slots in order to reduce test error (the mean and the standard deviation are used in the boxplot which is not included in this report).

The independent variables (parameters) for my experiement are as follow:

`is_mixed`: whether I test with different sizes of matrices in a single read or write.

`is_manyRW`: whether I test with multiple reads and writes at the same time with one size of matrices.

`is_cold`: whether I test without warming up the cache, meaning whether I fill in the cache with some data of matrices before beginning my test.

`is_R`: whether I test reads.

`is_W`: whether I test writes.

The function `time_putAndget()` is the main function I use to test execution time with different parameters. Inside the main function, I set up the correct type of the KV databases by using the function `setup_mode()`. Then, if I test mixed matrices, I will time reading and/or writing all sizes of matrices at once. Otherwise, I will time them separately in different loops. To reduce as much overhead as possible, I write a function `test_putAndget()`, which times `kv.get()` and/or `kv.read()`. The time slot calculated from that function gives the result to compare and experiment.

Results

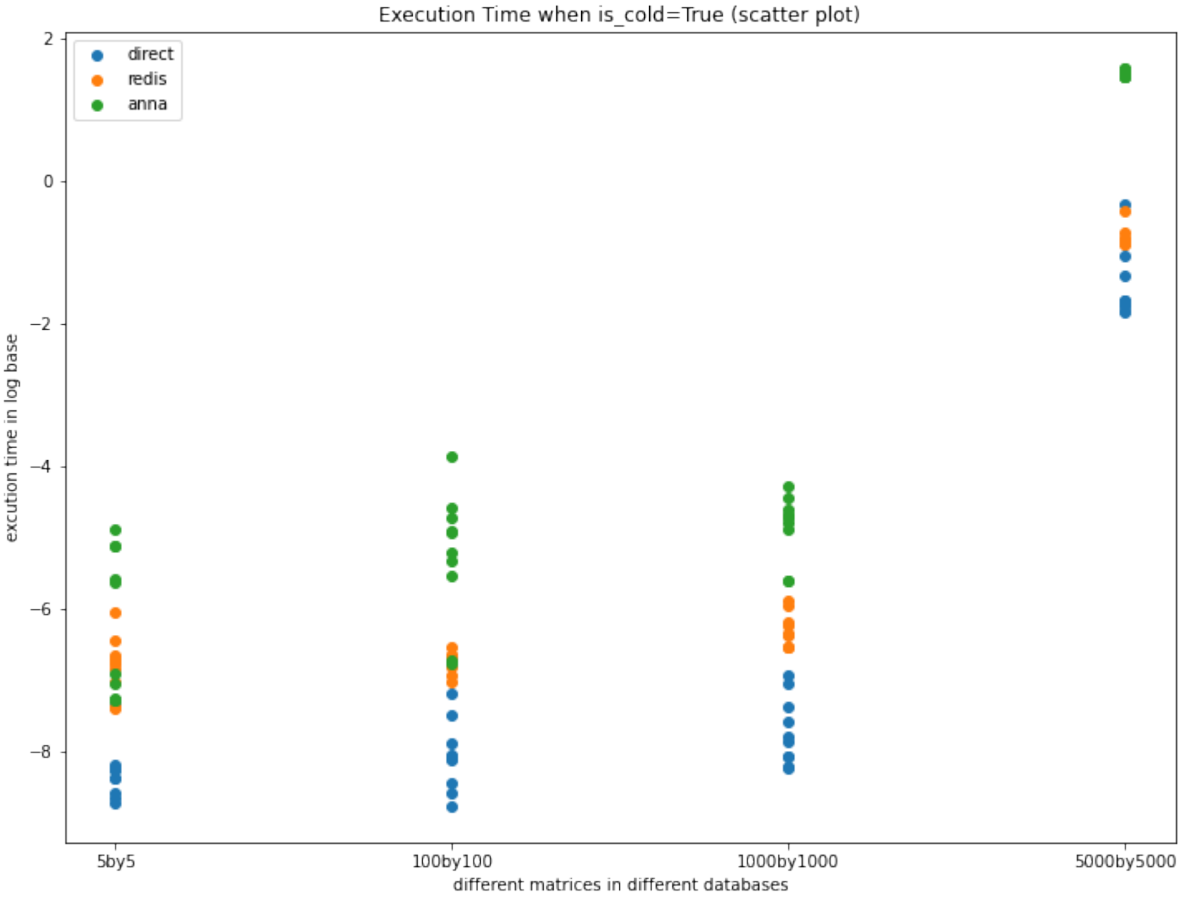
The default setting for all these tests is: `is_mixed=False`, `is_manyRW=False`, `is_cold=True`, `is_R=True`, and `is_W=True`.

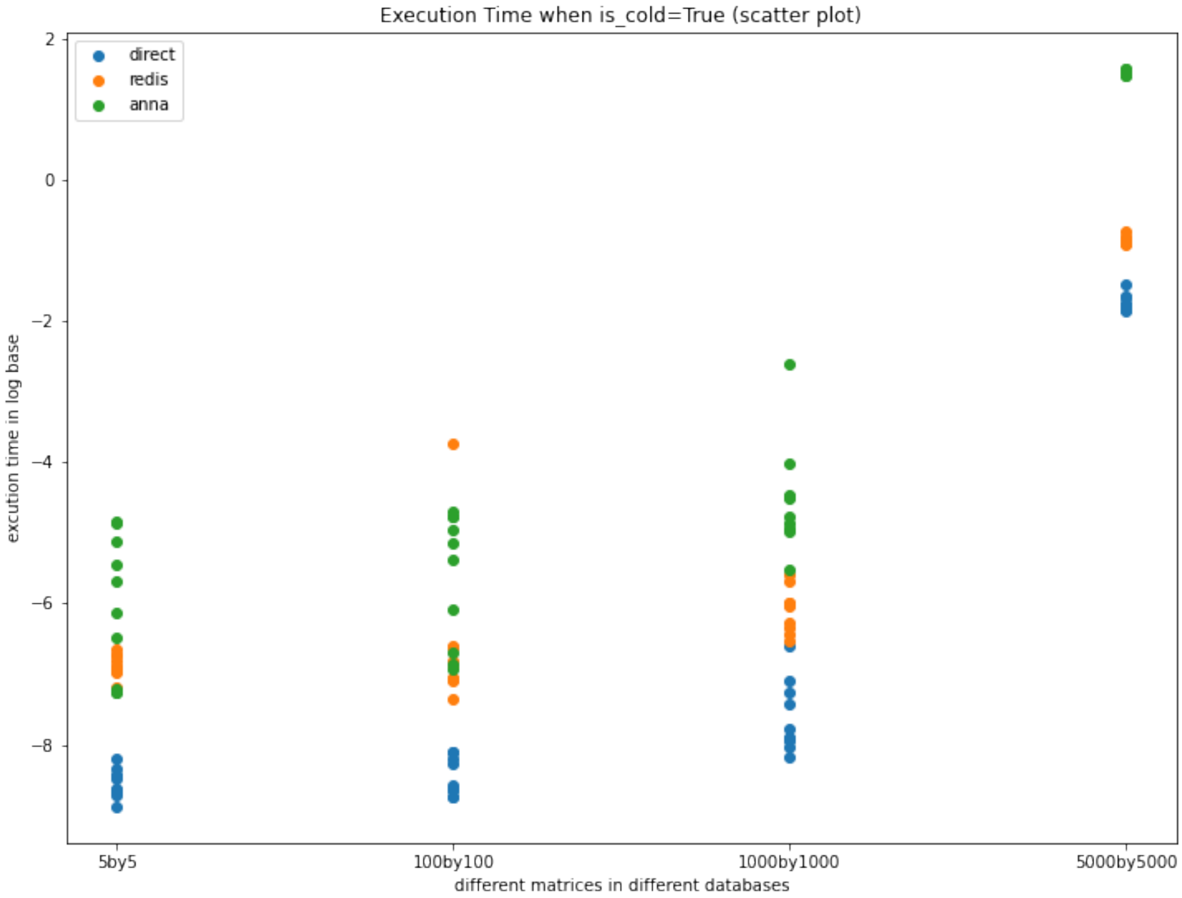
Notice that our plots are drawn with log-based sample points, so every point above the line y=0 has execution time actually way larger than point below that line.

1. Test cold and warm caches

I set `is_cold=True` for cold caches and `is_cold=False` for warm caches. Notice that I implement the warm caches by looping one more time and ignoring the time slot from the first loop.

Below are the plots comparing cold and warm caches for reads and writes of different KV databases.





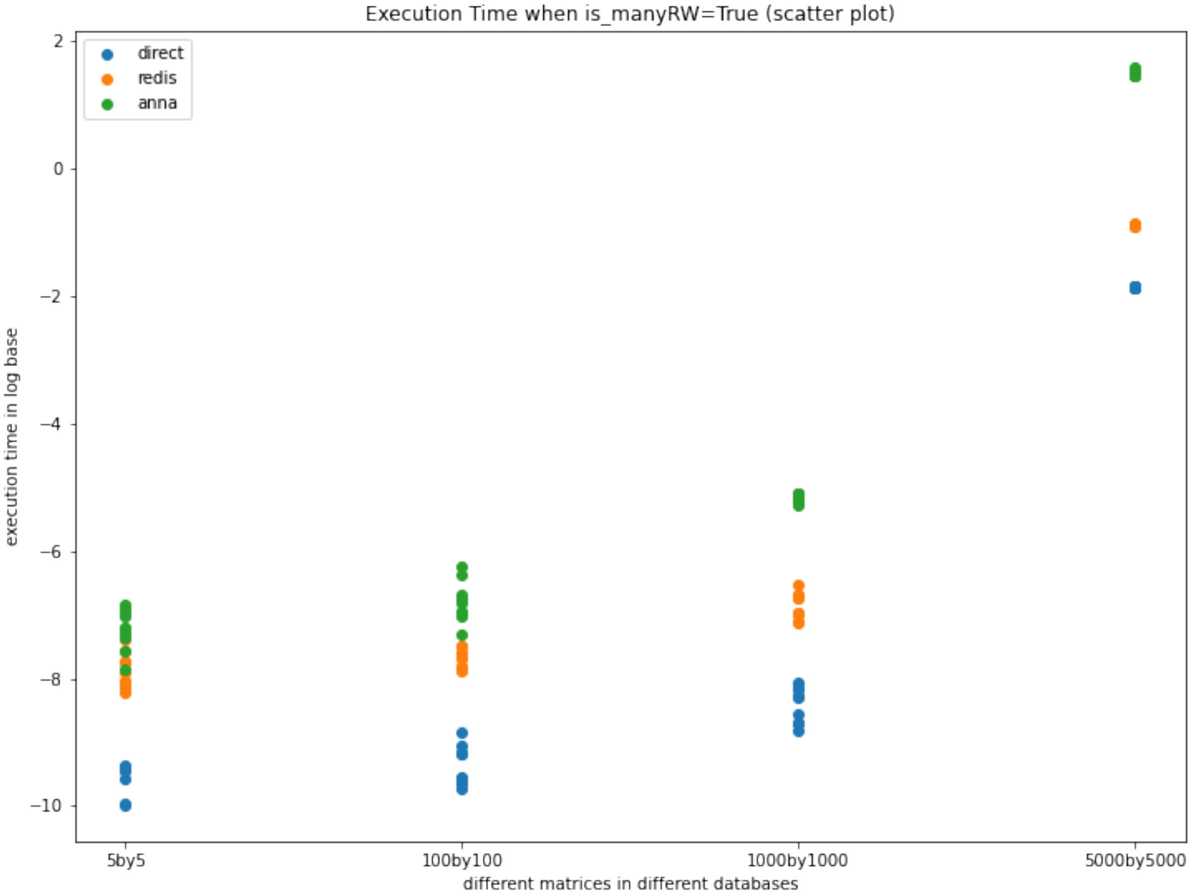
According to the plots for both warm and cold startups, as the size of the matrix increases, Redis behaves much better than Anna, and Redis acts nearly as well as the local dictionary. One thing to notice is that for 5000 by 5000 matrix, Anna performs much worse.

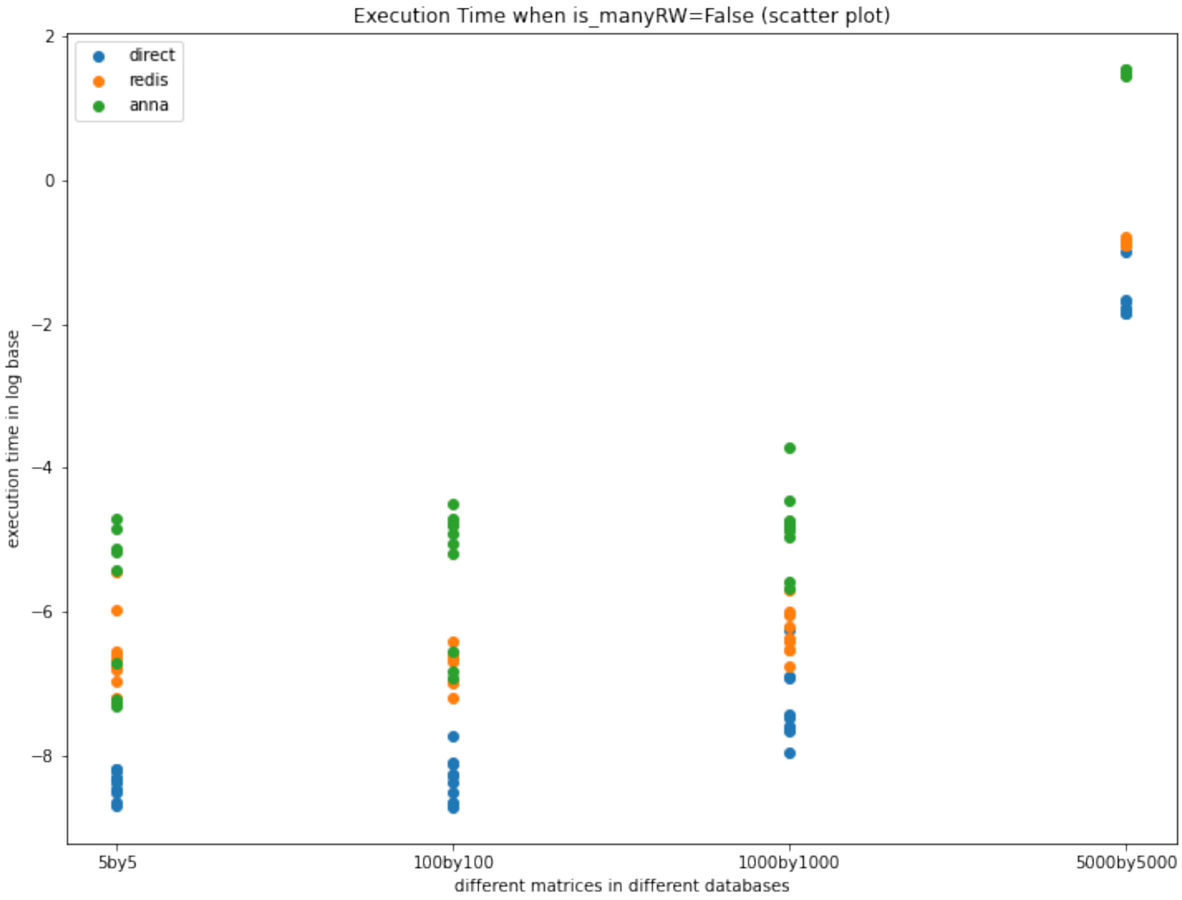
Redis and Anna spend less execution time in warm caches than cold caches. Another thing to notice is that in warm startup for large matrix size (5000 by 5000), Redis performs as well as the local dictionary.

2. Test mutiple reads and writes

I set `is_manyRW=False` for writing one matrix only once in the databases and `is_manyRW=True` for writing one matrix ten times in the databases. When `is_manyRW=True`, I record the mean time of those ten execution times to the list.

Below are the plots.


Execution Time when is_manyRW=True (scatter plot)
direct
redis
anna
2
0
−2
−4
−6
−8
−10
execution time in log base
5by5
100by100
1000by1000
5000by5000
different matrices in different databases

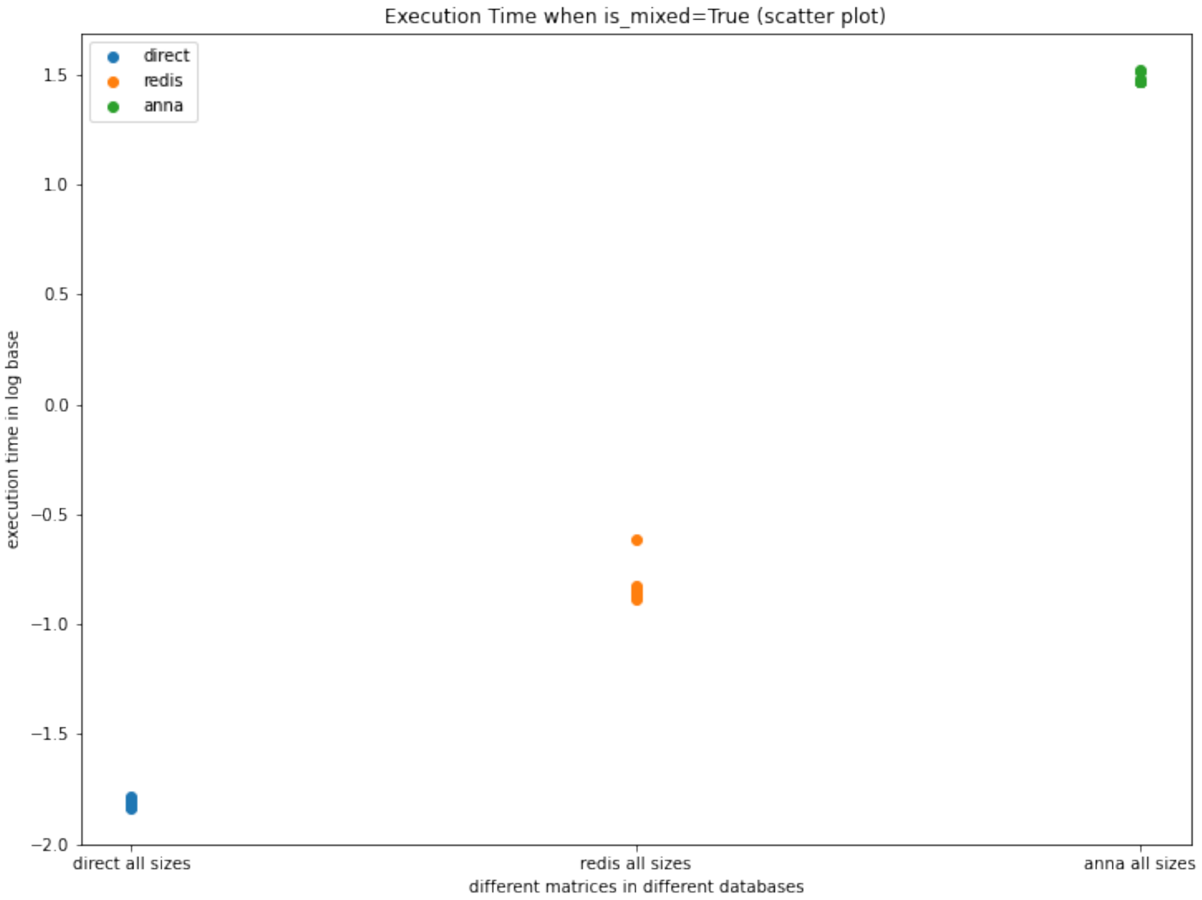


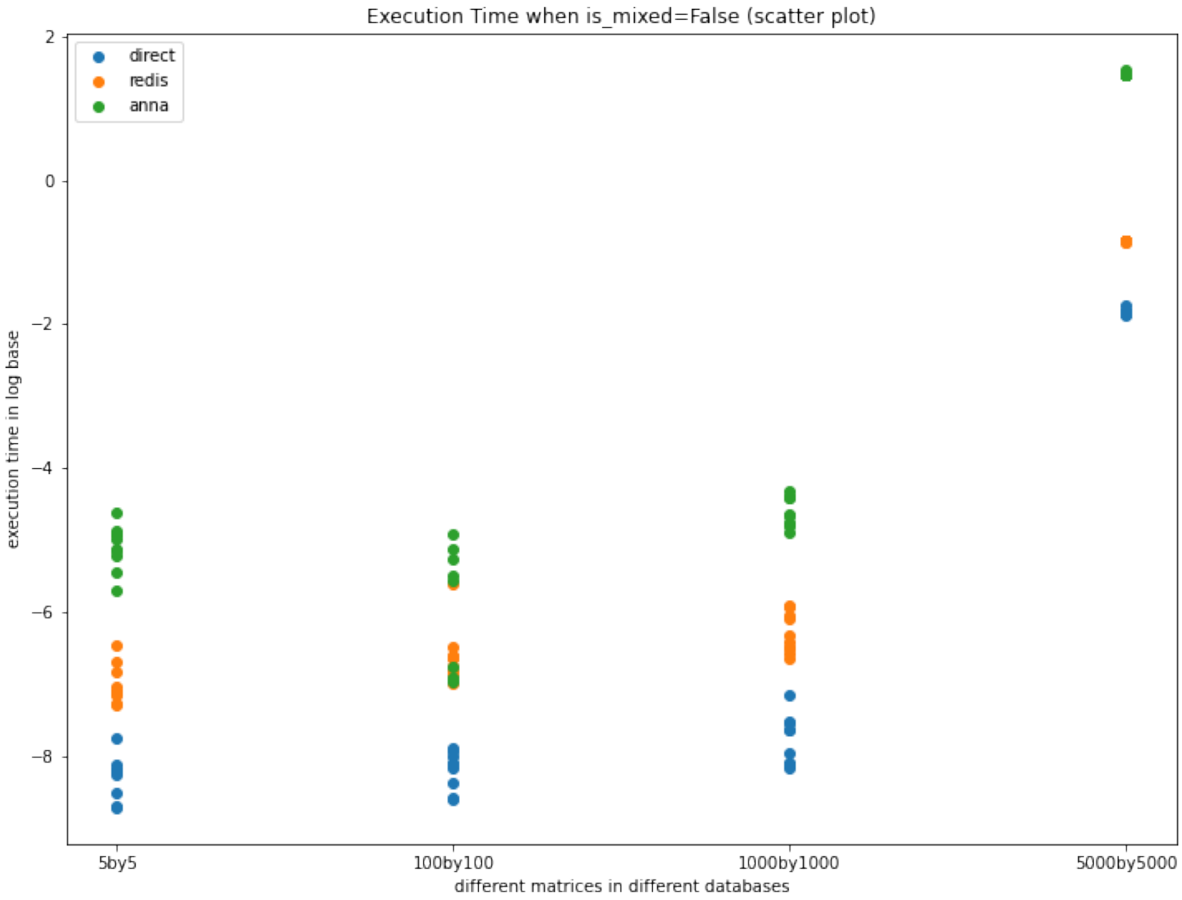
When doing multiple reads and writes at once, the outputs (sample points) are closer to each other for each different databases compared to single read and write. The vertical gaps for each size of matrix (in the plot) are wider in `is_manyRW=True` than `is_manyRW=False`. However, for very large matrix (5000 by 5000), Anna still performs way worse than Redis and local dictionary.

3. Test mixed sizes of matrices

I set `is_mixed=True` for reading and writing all sizes of matrices at once and `is_mixed=False` just as the default setup. I record the execution time of all matrices as a whole when `is_mixed=True`.

Below are the plots.


Execution Time when is_mixed=True (scatter plot)
direct
redis
anna
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
execution time in log base
direct all sizes
redis all sizes
anna all sizes
different matrices in different databases

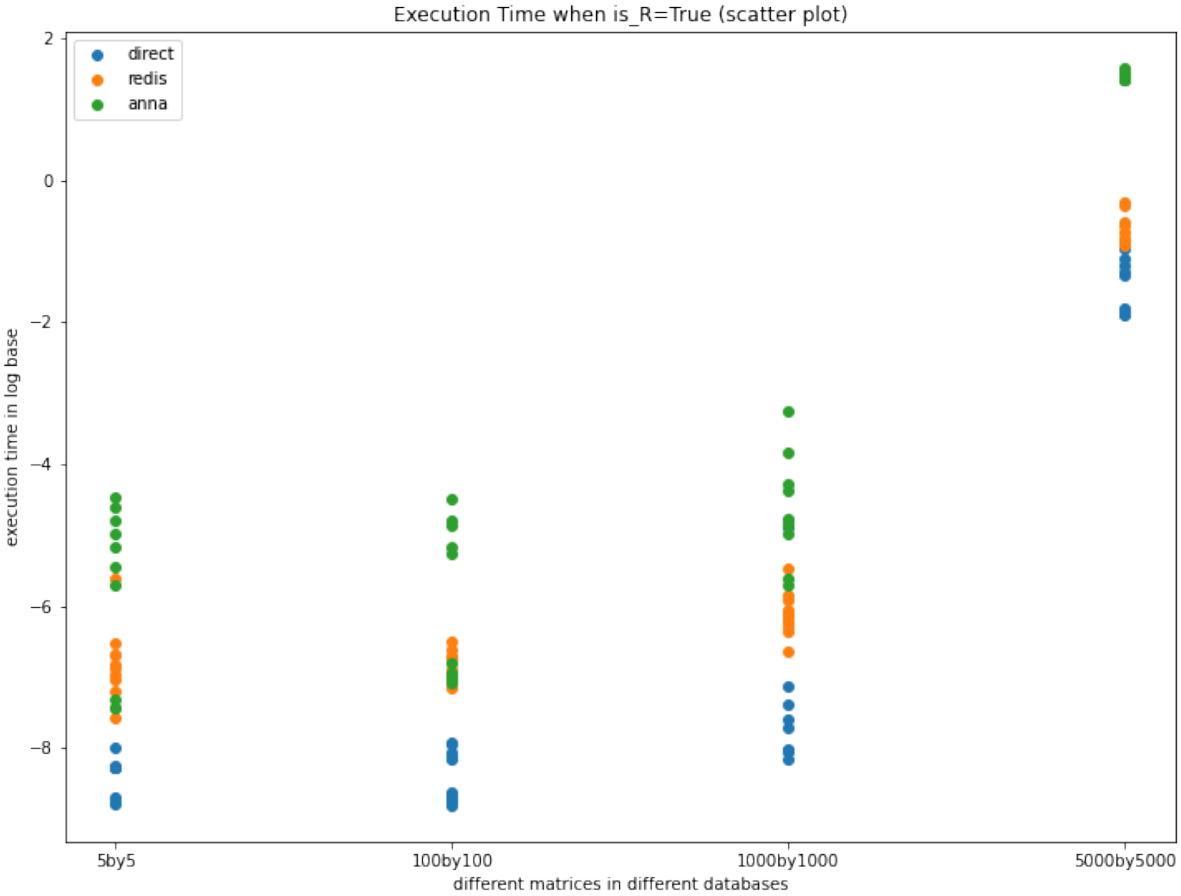


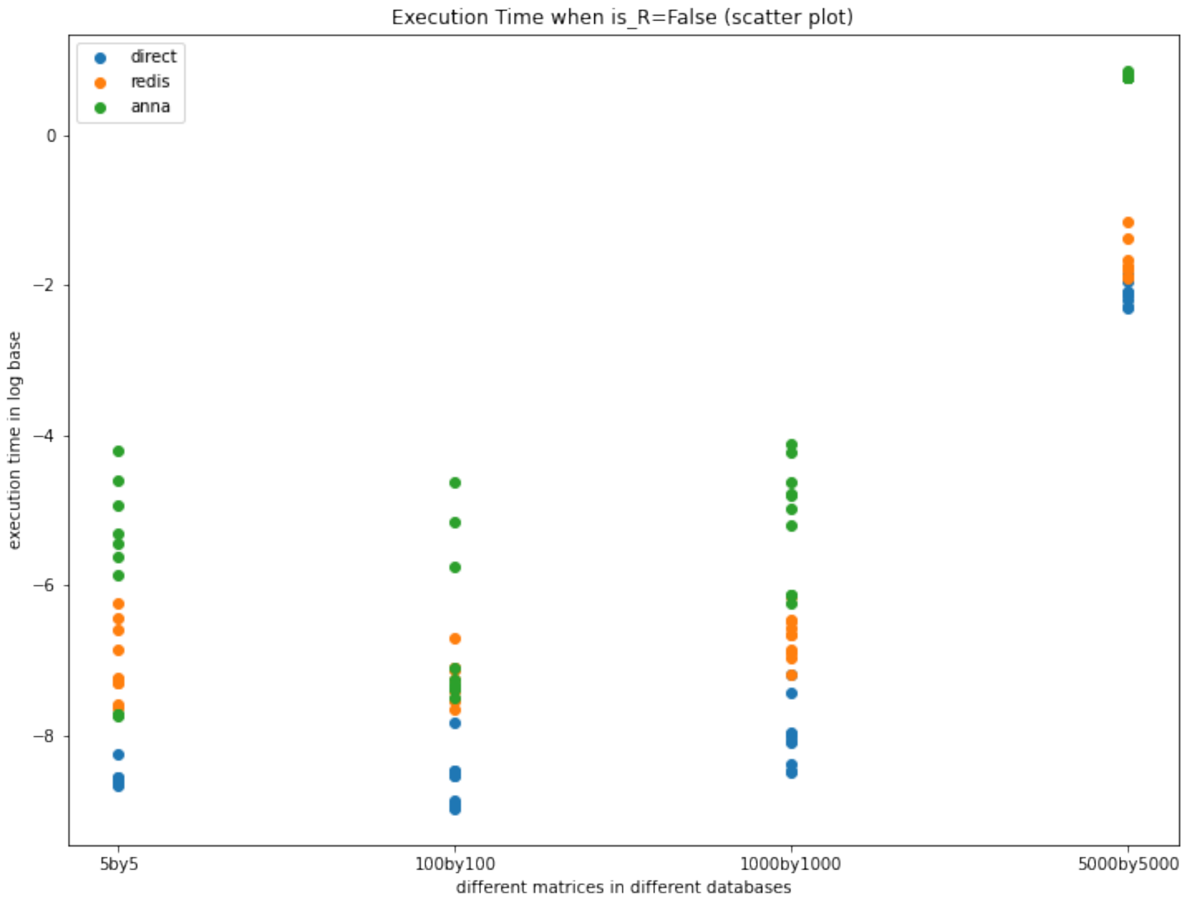
The parameter `is_mixed` gives a real-world simulation when KV databases receive different kinds of requests for puts and gets. According to the plot, Anna doesn't perform really well. Based on the previous plots, the reason is that Anna has low efficiency in reading and writing large-scale data (like 5000 by 5000 matrix). I can conclude this fact from the plot when `is_mixed=False`. In reading and writing matrices other than 5000 by 5000, Anna performs nearly as well as Redis.

4. Test only writes

I set `is_R=True` for timing both reads and writes and `is_R=False` for timing only writes.

Below are the plots.


Execution Time when is_R=True (scatter plot)
direct
redis
anna
2
0
−2
−4
−6
−8
execution time in log base
5by5
100by100
1000by1000
5000by5000
different matrices in different databases

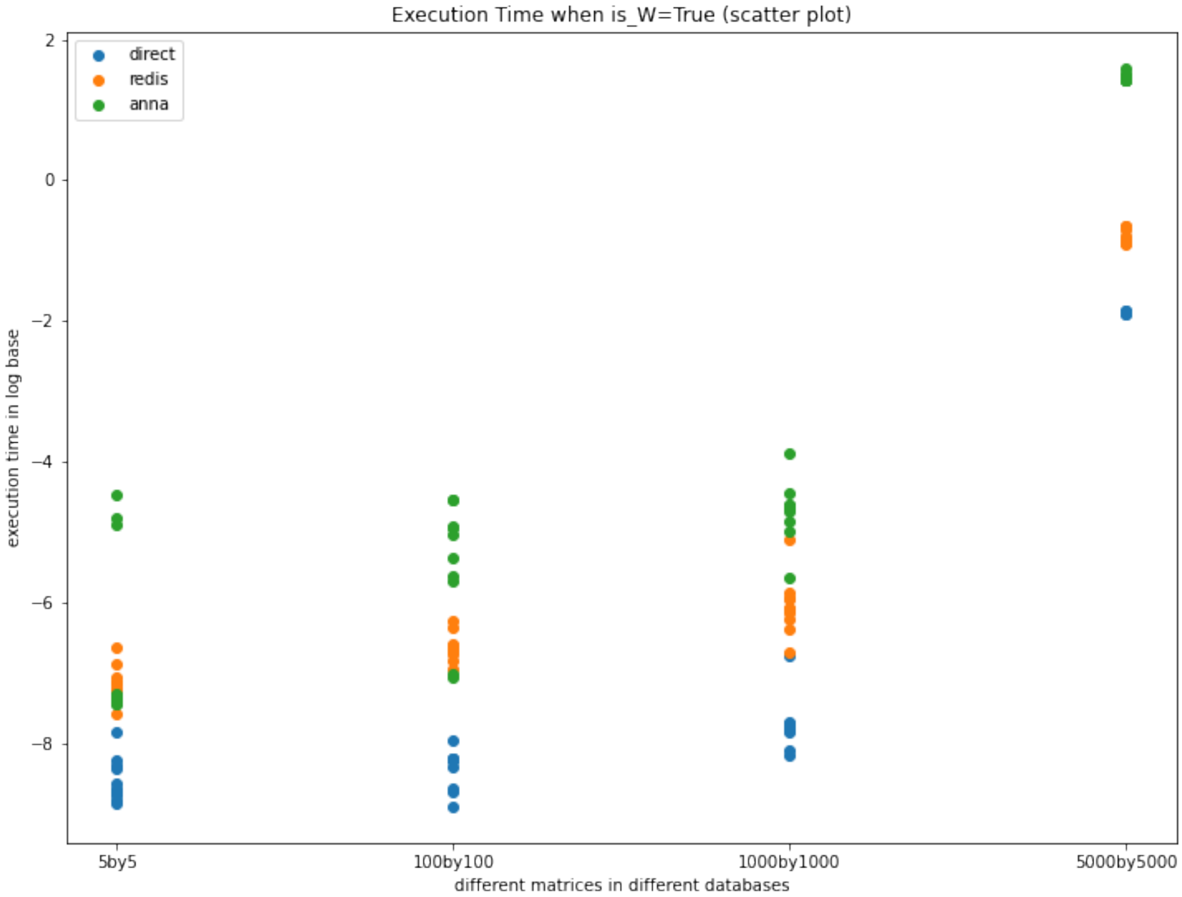


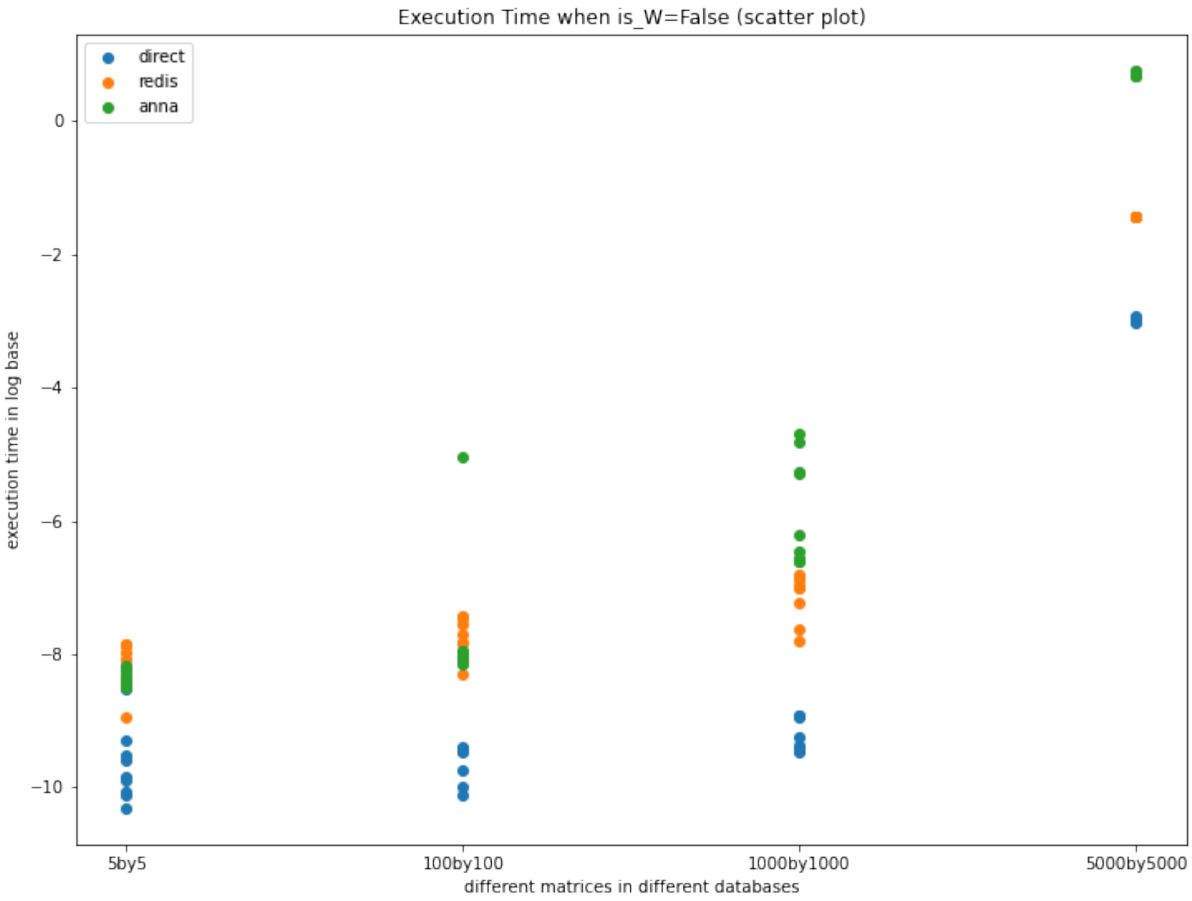
Writes take more time than reads in each databases. And that's the reason why the two plots above are not very different from each other.

5. Test only reads

I set `is_W=True` for timing both reads and writes and `is_W=False` for timing only reads.

Below are the plots.


Execution Time when is_W=True (scatter plot)
direct
redis
anna
2
0
−2
−4
−6
−8
execution time in log base
5by5
100by100
1000by1000
5000by5000
different matrices in different databases



Besides the result concluded in (4), it's interesting to see that with only reads, Anna performs as well as Redis and it is even better than Redis for matrices of sizes 5 by 5 and 100 by 100 (small size matrix).

Not much result can be generated from `is_R` and `is_W`, and these two parameters may give better result when changed together with other parameters.

Future Improvements

In my experiment, I only use 10 sample points for each size of matrices, which gives relatively high variance. In the future, I will try to use much more sample points (100 or even 1000) to reduce that variance.

My metric here is only the execution time. In the future, I may include more metrics besides time. And I will try to change more than one parameter at once and compare the results.

Conclusion

Overall (considering all the parameters I discuss above), Redis behaves nearly as well as the local dictionary, and Anna behaves well except for large-scale data (like 5000 by 5000 matrix). I think the overhead of reads and writes (due to the implementation of Anna) is the main reason why Anna doesn't handle large-scale data well.

One strength about my benchmark is that it can embrace more KV databases to test with simple modifications. And One weakness is that due to the limited size of sample points, outliers may influence the result and give high variance.

The most valuable thing I learn from this experiement is the process of building a good benchmark. And I will continue perfecting my current benchmark to help with my future experiments on KV databases for our system.