CSCI/ECEN 5673: Distributed Systems
Spring 2026

Programming Assignment Two

Team: Anirudh Ragam, Ishita Badole

Github: https://github.com/anirudhragam/distributed-systems-programming-assignments

Experiments Setup

The following VMs are created on Google Cloud Platform:

1. `buyer-server-vm` : runs the buyer server and the financial transactions SOAP server
2. `seller-server-vm` : runs the seller server
3. `customer-db-vm` : runs the customer db and customer db gRPC server
4. `product-db-vm` : runs the customer db and customer db gRPC server
5. `test-runner-vm` : we SSH into this VM and run the command to test scenarios

On the `test-runner-vm`, we ran the `performance_tests.py` file to run our experiments which we created for PA1.

In the performance tests, we define the `run_buyer_operations` function to run the 1000 buyer operations and the `run_seller_operations` to run 1000 buyer operations. These functions record a list of response times for each operation and return the list of 1000 response times to the caller.

The `run_scenario` function creates one thread for each seller and buyer (1, 10, or 100) in the experiment. Each thread calls its corresponding run operations function to run 1000 operations. The results of the threads are collected as the threads complete. These results are the list of response times for the 1000 operations. The `run_scenario` function creates a list of all seller response times and a list of all buyer response times.

The `compute_metrics` function takes a list of response times as input and computes the average response time and the throughput. Once all the threads have completed, `run_scenario` function calls `computer_metrics` for seller response times and buyer response times to get the average response time and throughput for one run for all sellers and buyers for a scenario.

The `run_experiments` function runs the experiment for 10 iterations. Finally it computes the overall average of the average response times and throughputs for buyers and sellers for the 10 iterations and reports them as the final results of the experiment.

`run_experiment` is called for 1, 10 and 100 buyers and sellers.

Experiment Results

Scenario 1: Run 1 seller and 1 buyer.

	Sellers		Buyers	
	Average Response Time (ms)	Throughput (ops/sec)	Average Response Time (ms)	Throughput (ops/sec)
Iteration 1	15.44 ms	64.77 ops/sec	15.31 ms	65.32 ops/sec
Iteration 2	15.73 ms	63.57 ops/sec	15.89 ms	62.94 ops/sec
Iteration 3	15.56 ms	64.26 ops/sec	15.97 ms	62.64 ops/sec
Iteration 4	15.43 ms	64.79 ops/sec	16.21 ms	61.68 ops/sec
Iteration 5	15.70 ms	63.70 ops/sec	16.68 ms	59.97 ops/sec
Iteration 6	15.60 ms	64.10 ops/sec	17.14 ms	58.34 ops/sec
Iteration 7	20.00 ms	50.01 ops/sec	22.03 ms	45.40 ops/sec
Iteration 8	16.91 ms	59.13 ops/sec	17.64 ms	56.69 ops/sec
Iteration 9	15.87 ms	63.02 ops/sec	18.22 ms	54.88 ops/sec
Iteration 10	15.59 ms	64.13 ops/sec	18.42 ms	54.30 ops/sec
Average	16.18 ms	62.15 ops/sec	17.35 ms	58.22 ops/sec

Scenario 2: Run 10 concurrent sellers and 10 concurrent buyers.

	Sellers		Buyers	
	Average Response Time (ms)	Throughput (ops/sec)	Average Response Time (ms)	Throughput (ops/sec)
Iteration 1	41.51 ms	24.09 ops/sec	47.77 ms	20.93 ops/sec
Iteration 2	53.38 ms	18.73 ops/sec	72.19 ms	13.85 ops/sec

Iteration 3	67.85 ms	14.74 ops/sec	101.20 ms	9.88 ops/sec
Iteration 4	78.22 ms	12.78 ops/sec	145.61 ms	6.87 ops/sec
Iteration 5	154.28 ms	6.48 ops/sec	229.13 ms	4.36 ops/sec
Iteration 6	196.19 ms	5.10 ops/sec	296.26 ms	3.38 ops/sec
Iteration 7	230.13 ms	4.35 ops/sec	360.51 ms	2.77 ops/sec
Iteration 8	229.60 ms	4.36 ops/sec	385.30 ms	2.60 ops/sec
Iteration 9	242.47 ms	4.12 ops/sec	450.25 ms	2.22 ops/sec
Iteration 10	242.35 ms	4.13 ops/sec	473.00 ms	2.11 ops/sec
Average	153.60 ms	9.89 ops/sec	256.12 ms	6.90 ops/sec

Scenario 3: Run 100 concurrent sellers and 100 concurrent buyers.

	Sellers		Buyers	
	Average Response Time (ms)	Throughput (ops/sec)	Average Response Time (ms)	Throughput (ops/sec)
Iteration 1	440.20 ms	2.27 ops/sec	590.47 ms	1.69 ops/sec
Iteration 2	557.94 ms	1.79 ops/sec	719.82 ms	1.39 ops/sec
Iteration 3	932.68 ms	1.07 ops/sec	1178.99 ms	0.85 ops/sec
Iteration 4	1268.07 ms	0.79 ops/sec	2072.99 ms	0.48 ops/sec
Iteration 5	1394.31 ms	0.72 ops/sec	2228.37 ms	0.45 ops/sec
Iteration 6	1490.02 ms	0.67 ops/sec	2031.78 ms	0.49 ops/sec
Iteration 7	3003.65 ms	0.33 ops/sec	3445.48 ms	0.29 ops/sec
Iteration 8	3635.42 ms	0.28 ops/sec	4230.96 ms	0.24 ops/sec
Iteration 9	3343.45 ms	0.30 ops/sec	4702.04 ms	0.21 ops/sec
Iteration 10	4048.31 ms	0.25 ops/sec	5042.46 ms	0.20 ops/sec
Average	2011.41 ms	0.85 ops/sec	2624.34 ms	0.63 ops/sec

Results Explanation

Comparisons between the three scenarios

Scenario 1 with one current buyer and seller has the lowest average response time, followed by Scenario 2 and Scenario 3 has the highest average response time.

The opposite trend is observed for throughput. Scenario 1 has the highest throughput, followed by Scenario 2, and Scenario 3 has the lowest throughput.

The explanation for these observations is that increasing the number of clients increases the load on the server. As concurrency rises, the server has to handle more concurrent requests, leading to higher simultaneous need for database connections and operations. So the server takes more time to complete per operation and the throughput decreases.

Every operation first checks that the session is valid. When there are more concurrent sellers and buyers, there are more database checks for session validation and that leads to a lot of database overhead. This explains the increase in the average response times for scenarios. One way to optimize this would be to keep the session table in memory for faster lookup.

For each scenario, average response time remains relatively consistent across runs but the throughput value decreases with each iteration. This could be because the size of the database tables increases over time. The performance test is designed in a way where items are continuously added to the database, so the database operations take longer leading to lower throughput. But each request does relatively the same steps, so the response time variance remains low.

Comparison between PA1 and PA2 results

The average latency in PA1 is significantly lower than in PA2, while the average throughput in PA1 is significantly higher. This trend was observed for each of the three scenarios. This is primarily because the PA1 performance tests were run locally on our machine, whereas in PA2, each system component was deployed on a separate GCE instance in us-west1 (Oregon). The resulting inter-component network communication introduces additional latency, which accounts for the higher average latency and lower average throughput observed in PA2.