Scalability Results

Zhou An(za53), Hunter Shen (zs149)

1. Introduction:

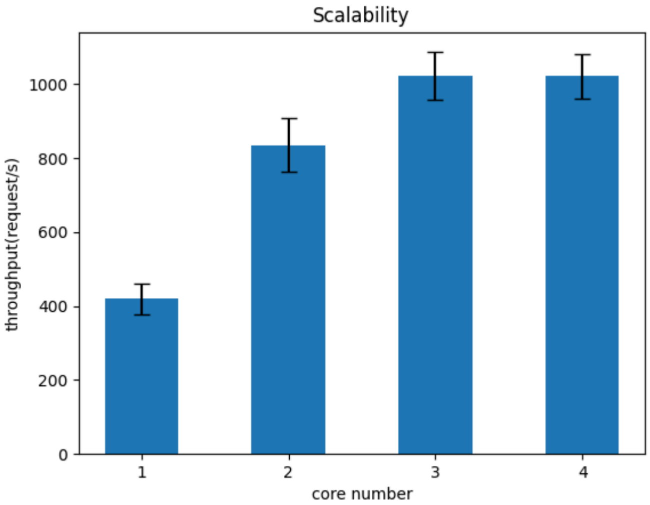
In this report, we present the scalability results of an exchange matching engine that we have developed. Our matching engine is a piece of software that matches buy and sell orders for a stock/commodities market. To achieve high throughput, we varied the number of CPU cores used by our server from 1 to 4. We also used error bars to show the variance in response time across multiple runs for each CPU core count.

2. Methodology:

We conducted scalability tests on our exchange matching engine to measure its performance as the number of CPU cores increased. To do this, we **used a fixed number of client requests and measured the throughput(requests/s) of the server**. Specifically, our server has 4 processes that keep listening to upcoming client requests. Similarly, the client side also has 4 processes, each generating 1000 clients with 1 request, and we specify the core number the server could use by:

```
taskset -c 0,1,2,3 python3 server.py
```

3. Results:



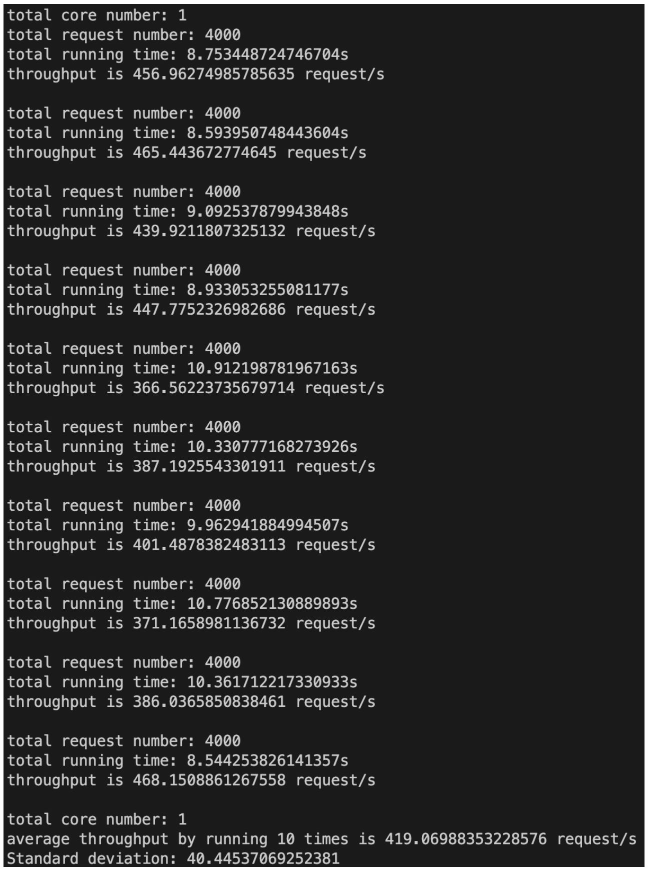
We runned the program 10 times for each number of cores to get average throughput, our results showed that the throughput of our exchange matching engine increased as the number of CPU cores increased. Specifically, with 1 CPU core, the average throughput was 419.07 requests/s with a deviation of 40.45 requests/s. With 2 CPU cores, the average throughput increased to 835.61 requests/s with a deviation of 71.90 requests/s. With 3 CPU cores, the average throughput further increased to 1022.30 requests/s with a deviation of 64.50 requests/s. Finally, with 4 CPU cores, the average throughput was 1021.55 requests/s with a deviation of 59.24 requests/s.

The results showed that **the largest increase in throughput occurred when the number of CPU cores increased from 1 to 2**. The throughput increased from an average of 419.07 requests/s with 1 CPU core to 835.61 requests/s with 2 CPU cores, indicating an increase of approximately 99%. One possible explanation for this significant increase is that the exchange matching engine's workload is highly parallelizable. By using multiple CPU cores, the engine can process multiple requests simultaneously, leading to a substantial increase in throughput. However, when the number of CPU cores is increased beyond 2, the benefits of parallelization may start to diminish due to factors such as synchronization overhead or contention for shared resources.

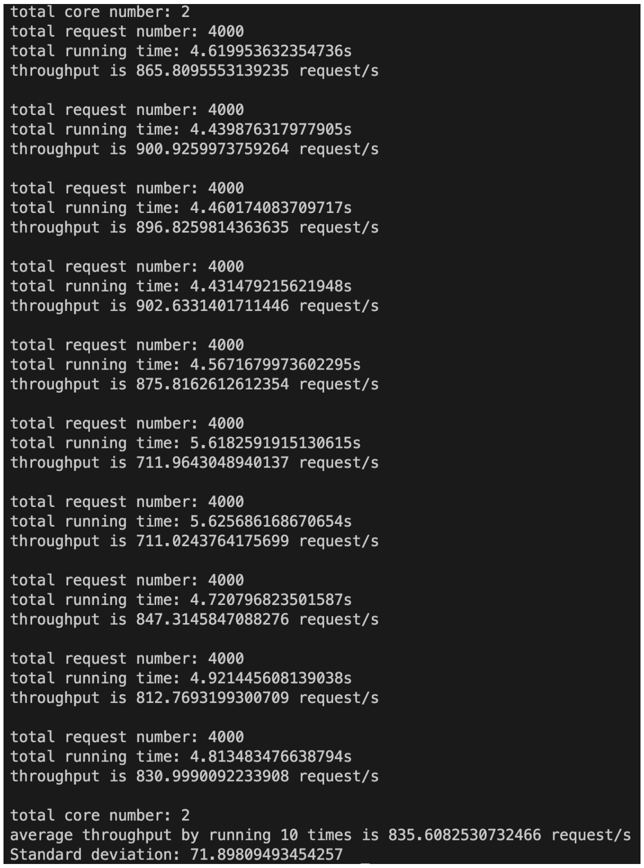
The results also indicate that **adding more CPU cores beyond 3 did not significantly improve the throughput** of the exchange matching engine. This suggests that there may be diminishing returns for each additional core beyond a certain point. Alternatively, the exchange matching engine may have other bottlenecks that limit its throughput, such as network bandwidth, disk I/O, or memory bandwidth. These bottlenecks may limit the effectiveness of adding more CPU cores beyond a certain point.

Appendix:

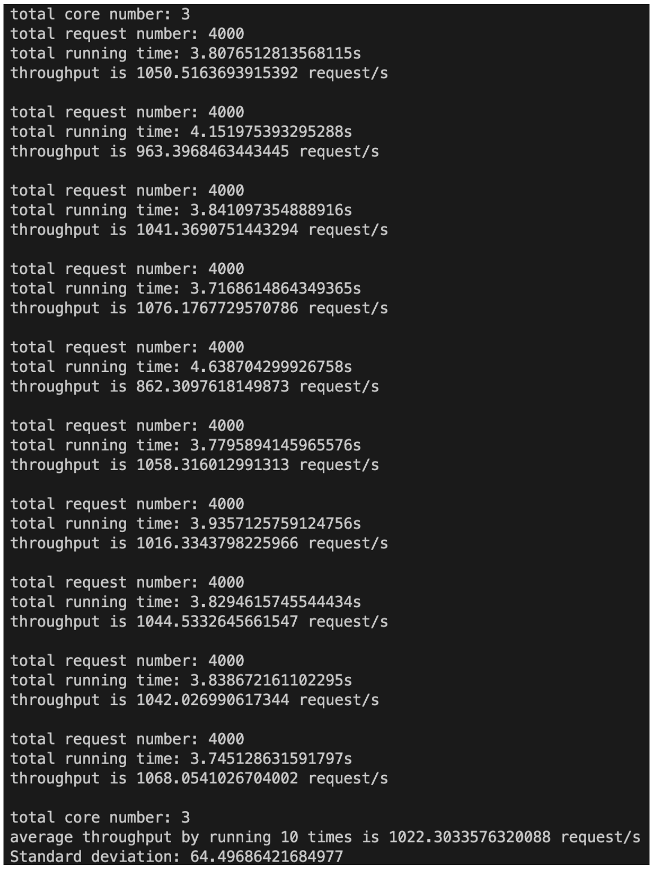
Full details of the results of the server running on different numbers of cores：



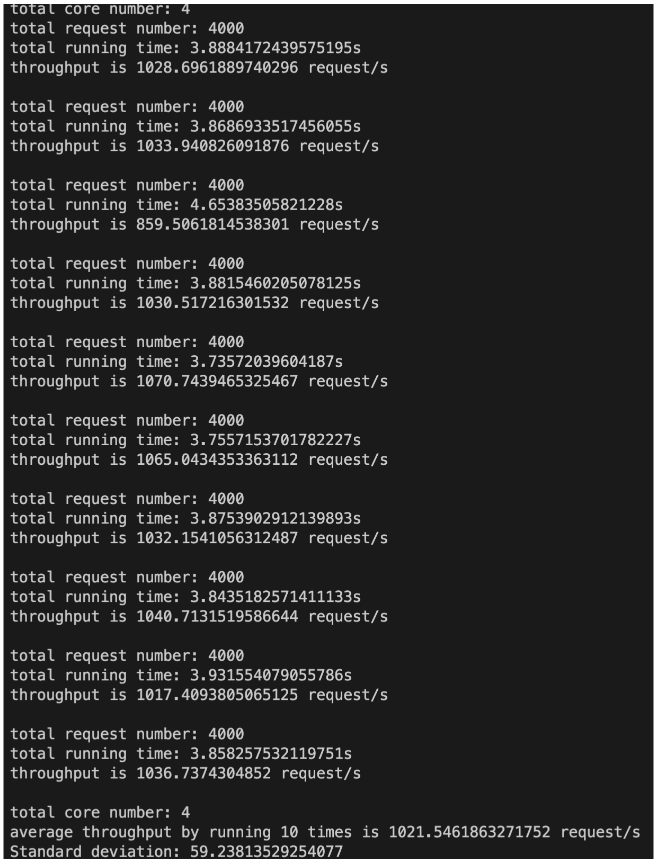
```
total core number: 1
total request number: 4000
total running time: 8.753448724746704s
throughput is 456.96274985785635 request/s

total request number: 4000
total running time: 8.593950748443604s
throughput is 465.443672774645 request/s

total request number: 4000
total running time: 9.092537879943848s
throughput is 439.9211807325132 request/s

total request number: 4000
total running time: 8.933053255081177s
throughput is 447.7752326982686 request/s

total request number: 4000
total running time: 10.912198781967163s
throughput is 366.56223735679714 request/s

total request number: 4000
total running time: 10.330777168273926s
throughput is 387.1925543301911 request/s

total request number: 4000
total running time: 9.962941884994507s
throughput is 401.4878382483113 request/s

total request number: 4000
total running time: 10.776852130889893s
throughput is 371.1658981136732 request/s

total request number: 4000
total running time: 10.361712217330933s
throughput is 386.0365850838461 request/s

total request number: 4000
total running time: 8.544253826141357s
throughput is 468.1508861267558 request/s

total core number: 1
average throughput by running 10 times is 419.06988353228576 request/s
Standard deviation: 40.44537069252381
```



```
total core number: 2
total request number: 4000
total running time: 4.619953632354736s
throughput is 865.8095553139235 request/s

total request number: 4000
total running time: 4.439876317977905s
throughput is 900.9259973759264 request/s

total request number: 4000
total running time: 4.460174083709717s
throughput is 896.8259814363635 request/s

total request number: 4000
total running time: 4.431479215621948s
throughput is 902.6331401711446 request/s

total request number: 4000
total running time: 4.5671679973602295s
throughput is 875.8162612612354 request/s

total request number: 4000
total running time: 5.6182591915130615s
throughput is 711.9643048940137 request/s

total request number: 4000
total running time: 5.625686168670654s
throughput is 711.0243764175699 request/s

total request number: 4000
total running time: 4.720796823501587s
throughput is 847.3145847088276 request/s

total request number: 4000
total running time: 4.921445608139038s
throughput is 812.7693199300709 request/s

total request number: 4000
total running time: 4.813483476638794s
throughput is 830.9990092233908 request/s

total core number: 2
average throughput by running 10 times is 835.6082530732466 request/s
Standard deviation: 71.89809493454257
```



```
total core number: 3
total request number: 4000
total running time: 3.8076512813568115s
throughput is 1050.5163693915392 request/s

total request number: 4000
total running time: 4.151975393295288s
throughput is 963.3968463443445 request/s

total request number: 4000
total running time: 3.841097354888916s
throughput is 1041.3690751443294 request/s

total request number: 4000
total running time: 3.7168614864349365s
throughput is 1076.1767729570786 request/s

total request number: 4000
total running time: 4.638704299926758s
throughput is 862.3097618149873 request/s

total request number: 4000
total running time: 3.7795894145965576s
throughput is 1058.316012991313 request/s

total request number: 4000
total running time: 3.9357125759124756s
throughput is 1016.3343798225966 request/s

total request number: 4000
total running time: 3.8294615745544434s
throughput is 1044.5332645661547 request/s

total request number: 4000
total running time: 3.838672161102295s
throughput is 1042.026990617344 request/s

total request number: 4000
total running time: 3.745128631591797s
throughput is 1068.0541026704002 request/s

total core number: 3
average throughput by running 10 times is 1022.3033576320088 request/s
Standard deviation: 64.49686421684977
```



```
total core number: 4
total request number: 4000
total running time: 3.8884172439575195s
throughput is 1028.6961889740296 request/s

total request number: 4000
total running time: 3.8686933517456055s
throughput is 1033.940826091876 request/s

total request number: 4000
total running time: 4.65383505821228s
throughput is 859.5061814538301 request/s

total request number: 4000
total running time: 3.8815460205078125s
throughput is 1030.517216301532 request/s

total request number: 4000
total running time: 3.73572039604187s
throughput is 1070.7439465325467 request/s

total request number: 4000
total running time: 3.7557153701782227s
throughput is 1065.0434353363112 request/s

total request number: 4000
total running time: 3.8753902912139893s
throughput is 1032.1541056312487 request/s

total request number: 4000
total running time: 3.8435182571411133s
throughput is 1040.7131519586644 request/s

total request number: 4000
total running time: 3.931554079055786s
throughput is 1017.4093805065125 request/s

total request number: 4000
total running time: 3.858257532119751s
throughput is 1036.7374304852 request/s

total core number: 4
average throughput by running 10 times is 1021.5461863271752 request/s
Standard deviation: 59.23813529254077
```