Scalability: Exchange Matching Engineering

Robust Server Software

Homework 3 Report

1. Introduction

We accept requests from another server and spawn threads via cached thread pool to handle them without blocking. We use maven to manage our project and use MyBatis as persistence framework and MySQL to store the info of accounts, positions and orders. With the ACID properties of transactions and optimistic locking, we can safely take care of incoming requests. With NIO Selector and index of orders created in MySQL, we render some improvement on the system' s reacting performance.

2. Implementation

We use a different server running client program using multiple threads to test our scalability in their socket channels. To make sure the results are comparable (the remaining orders in the database will have a huge impact on the result), we initialize the database during each test.

To test our scalability, we focus on two aspects: throughput and latency. We create the shell file to run the jar file with taskset, spawn 1000, 5000, 10000 threads from thread pool for each test and implement request of account creation for each thread. The server being tested will run the server program with 1, 2, 3, 4 cores respectively. The result for the latency of receiving responses at the client side shows as following. The unit of the abscissa is the number of threads, and the unit of the ordinate is milliseconds.

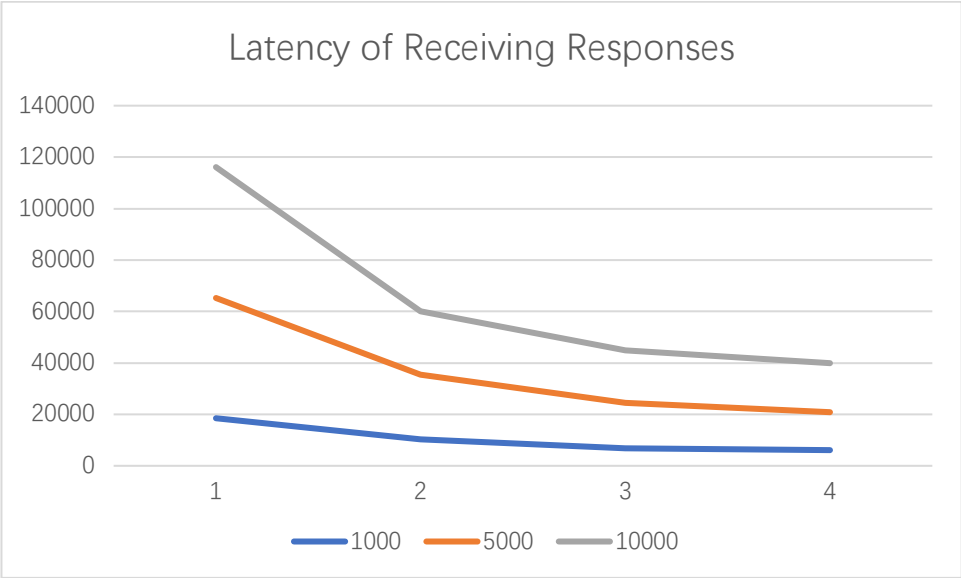

Figure 1 Latency of Receiving Response

3. Analysis

We can see, from the graph, the different colors of lines signify the different threads used to send request concurrently and each thread contains the query of 10k orders. The unit of the latency is milliseconds, we can see that the latency of receiving responses drop down as the number of cores increases, yet with the decrease of gradience. Obviously, the improve of performance can be credited to the parallel programming performed by multiple cores. However, when it comes to 4 cores, the performance of the system decreases a little. Our assumption is that the IO operations performed by MySQL require lots of CPU resources, so for the 4-core VCM, assigning the whole 4 cores to the system will indeed hinder the performance of MySQL and thus, the whole latency.