OS Assignment 1 Group 02

Introduction:

Threads are a way for a program to operate more efficiently, by executing multiple tasks simultaneously. With threads a program can improve its overall performance and responsiveness. The goal of the assignment is to learn about threads, and to understand how they function. The focus of this report will be on the performance of a server handling multiple clients, when implemented with both single- and multithreaded servers.

Task:

The task is to make a multi and single threaded server that can handle multiple clients simultaneously. The clients will send two numbers and an operator to the server. The server will perform an arithmetic operation based on the input and return the result back to the client. The server needs to support addition(A), subtraction(S), multiplication(M), division(D) and modulus operation(MOD). For example, the client sends 20 20 A, and the server returns 40. The performance is observed by calculating the time taken and comparing the difference.

System Design and Implementation:

Given the task there are two server implementations. One following the single-threaded model and the other employing a multi-threaded approach. Both servers operate on the same communication port, their internal mechanisms for managing client interactions showcase the contrast between sequential and parallel processing.

The single-threaded server utilizes a straightforward approach where it listens for incoming client connections on a predefined port. Upon accepting the connection it processes client requests synchronously within a single thread of execution. The server reads and incoming request,performs the calculations and returns the result to the client. This cycle repeats for each client one at a time. With each new connection requiring the completion of the previous before proceeding.

Since this model each client must wait for all preceding client requests to be processed it can lead to significant delays when dealing with multiple simultaneous connections.

In contrast the multi threaded server is designed to handle concurrent client requests. It listens on the same port and upon accepting a new client it delegates the request handling to a separate thread by instantiating a clienthandler. This allows the server to return to listening for new connections immediately while the individual clienthandler instances process each client request in parallel. Each clienthandler thread performs the same sequence of actions as the single threader server,but these actions occur concurrently for multiple clients. Which can significantly reduce the wait time for each connection under heavy client load.

Client behavior is simulated using the clientTask class which generates random calculation requests and measures the round trip time for each request to be processed by the server. ClientSimulator manages a pool of such tasks and executes them in parallel to simulate concurrent client activity.

Both server approaches use the same performOperation method which supports the required arithmetic operations in this assignment. This method keeps the calculation process the same for both servers,making sure we can compare them fairly.

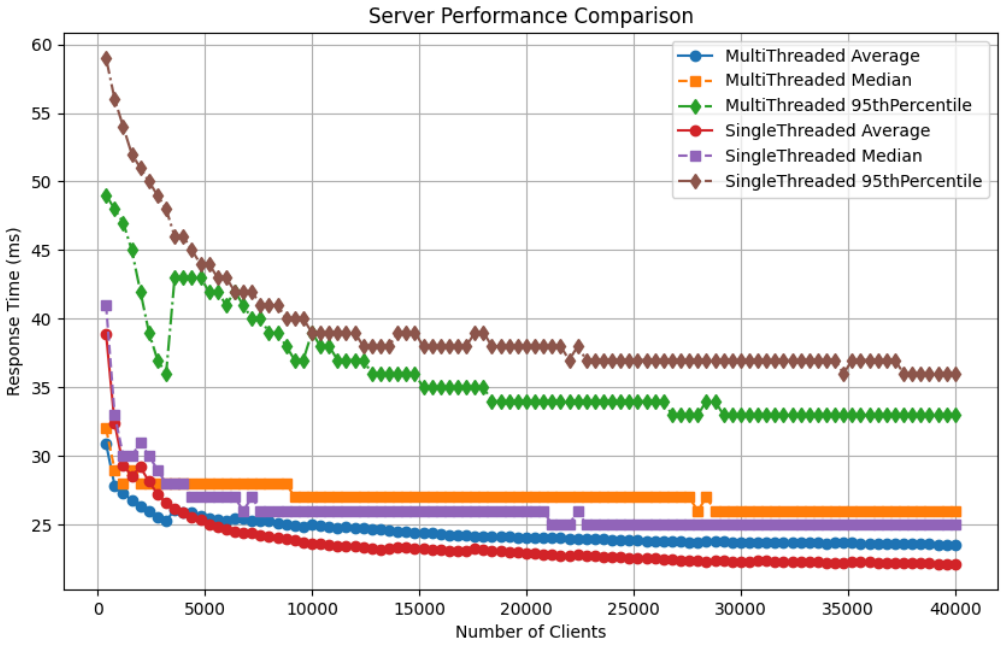
Testing:

The point of the testing was to evaluate and compare the time performance of single-threaded and multi-threaded servers under the same specific load. For this test we used 400 simultaneous client connections each performing calculation requests with the test repeated over 100 rounds to ensure statistical significance and reliability of the results. The test was set up on the same computer to eliminate any bias.

Data collection:

For each client request,we measured the time from sending the request to receiving the servers response. And the response times were logged and key performance metrics such as average response time,median response time,and the 95th

percentile were computed for each round. The performance metrics were then aggregated across all 100 rounds to establish a robust dataset for each server type. Execution of the test can be viewed in the video.



Observation:

The graph presents the performance outcomes from a series of test involving 400 clients over 100 rounds. It shows the response times for both single threaded and multi threaded servers as the client load persists. Performance is measured using three key metrics. Average response time,median response time and 95th percentile response time.

The results gathered are unexpected of what we thought the outcome should look like. The single threaded server recorded a better response time for median and average metric. The only metric the multi threaded seems to be superior is the 95th percentile metric.

Conclusion:

The findings from our performance test yielded some unexpected results. Despite conventional expectations that the multi threaded server would outperform the single threaded server under high client load our data showed that the single threaded server had a better median and average response time.

However the multi threaded server did have superior performance in managing the upper extremes of response times as we can see in the graph.
Our own theory is that the calculations and generally the load is too light for a modern cpu to show the real performance for single threaded vs multi threaded.