CS42 OPERATING SYSTEMS

PROJECT #1

COMMAND SERVER

EXPERIMENTS REPORT

28.02.2024

Yasemin Akın Section 002 22101782

Göktuğ Serdar Yıldırım Section 002 22103111

Introduction

Our study examines the server program's performance under different conditions, including varying numbers of clients and window sizes (WSIZE). We target the timing of the task completion in those instances. The possibility of added clients might cause more time needed, proportional to the efficiency of the server handling more than one request at a time. Besides, the implementation of the window size has a trade-off between throughput and robustness; bigger sizes may increase throughput but risk congestion, and smaller sizes may offer robustness but limit throughput. Our measurements will be taken using an accurate timing method and shown in tables and graphs for better understanding. Statistical tools will be used to determine if these trends are statistically significant. This study will give an image of the program's scalability and best design in different network conditions and tasks.

How We Have Conducted the Experiments?

We have used the script below to run the same command file concurrently within multiple clients. The script allowed us to input the number of clients to be open simultaneously, the message queue name, the command file's name to run the opened clients in batch mode, and the WSIZE variable. It enabled us to open as many terminals as the number of clients entered and run this number of clients in different terminals. In this way, we were able to run the same installed programs on different clients at the same time. First, we kept the number of clients constant by using it as an independent variable. We tried to observe the effect of the WSIZE variable on execution time by increasing it step by step, using WSIZE as an independent variable. Then, we increased the number of clients and tried the same test on other client numbers. Thus, by replacing the number of clients as an independent variable, we observed its effect on execution time when WSIZE is constant. The table below shows the experimental results of the impact of WSIZE values and the number of clients on execution time (microseconds).

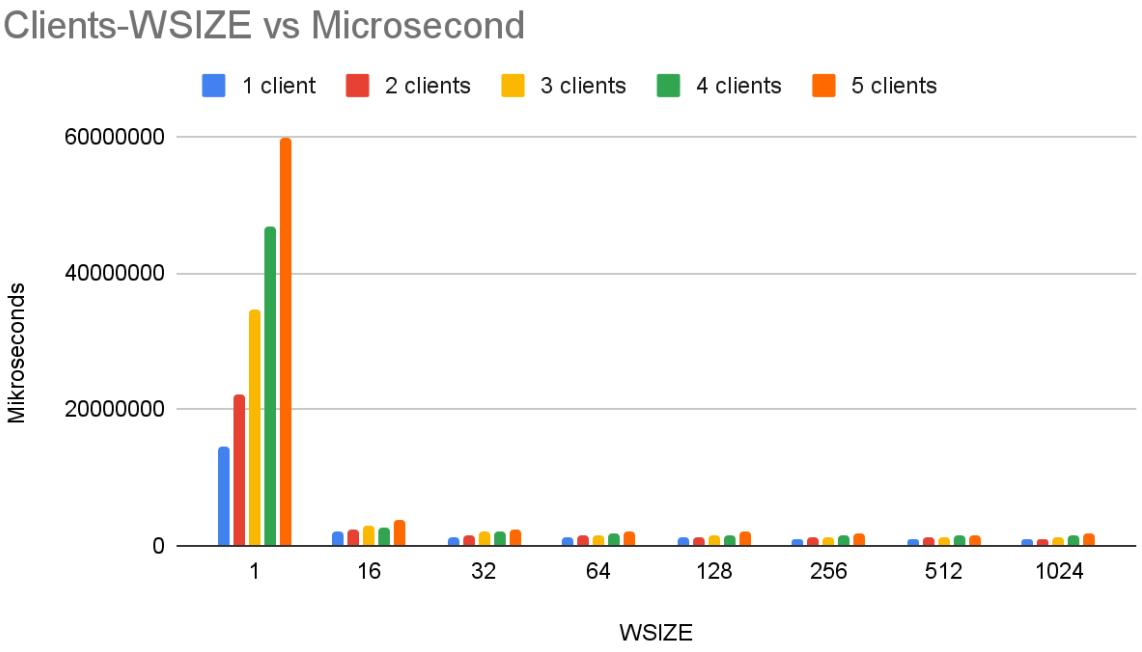
$ run_clients.sh

```
#!/bin/bash

# Check if the correct number of arguments is provided
if [ "$#" -ne 5 ]; then
  echo "Usage: $0 <number_of_clients> <executable_name> <MQNAME> <batch_mode_file_name> <wsize>"
  exit 1
fi

num_clients=$1
executable_name=$2
mqname=$3
batch_mode_file_name=$4
wsize=$5

# Loop to start multiple clients in separate terminals
for ((i = 1; i <= num_clients; i++)); do
  echo "Starting client $i in a new terminal..."
  gnome-terminal -- bash -c "./$executable_name $mqname -b $batch_mode_file_name -s $wsize; exec bash"
done

echo "All $num_clients clients have been started."
```

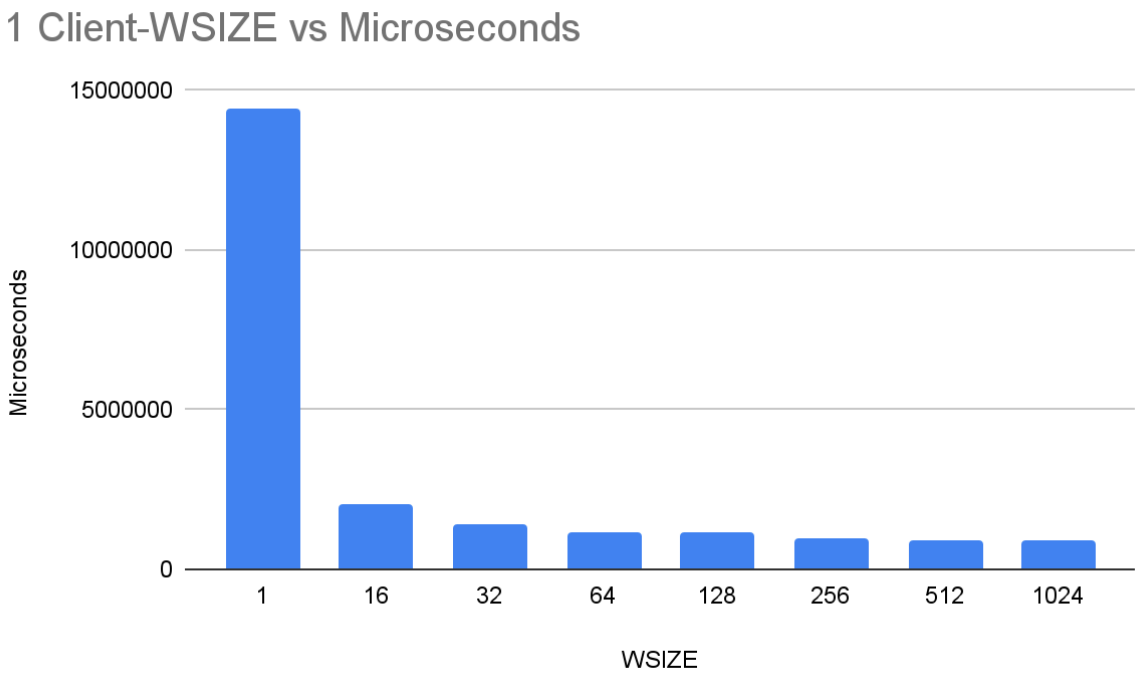
Example command to run the script:

```
yase@yase:~/Desktop/Project1$ ./run_clients.sh  cli "/yase" "com3.txt" 1024
```

Number of Clients → / WSIZE ↓	1	2	3	4	5
1	14447290	22291561	34605629	46874569	59740983
16	2036783	2372818	2991173	2747888	3832034
32	1388768	1633956	2009107	2020333	2421436
64	1153829	1506008	1460571	1699644	2099414
128	1170676	1147319	1528634	1635244	1992333
256	977482	1171584	1160797	1489487	1695585
512	902481	1205266	1315375	1507807	1451048
1024	909010	1075810	1279096	1436613	1694741

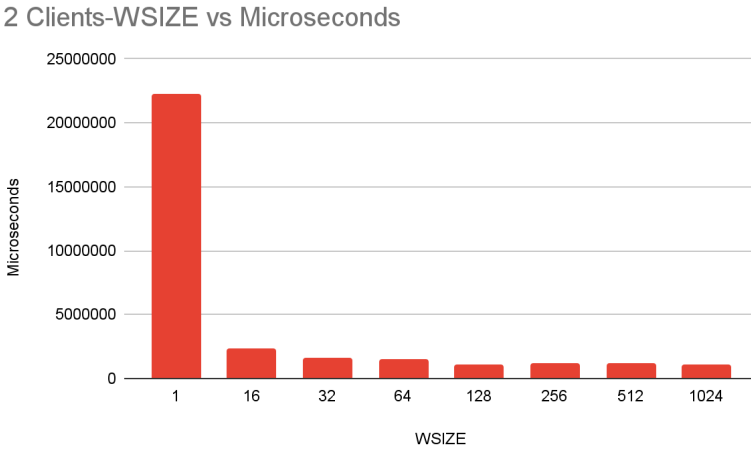


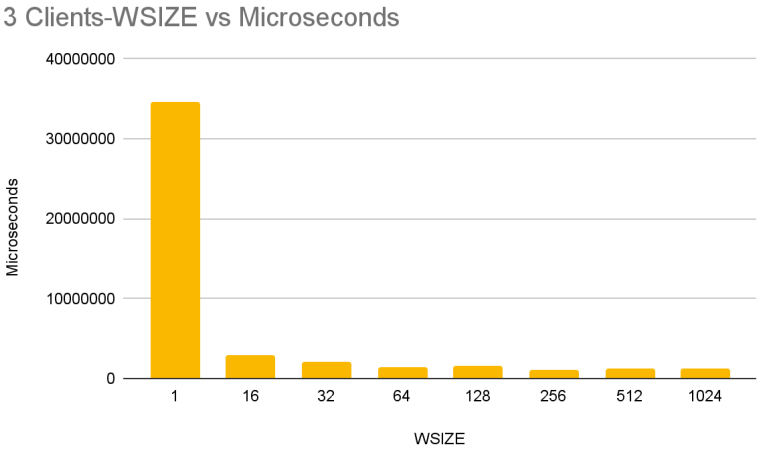
Benchmark 1: Wsize

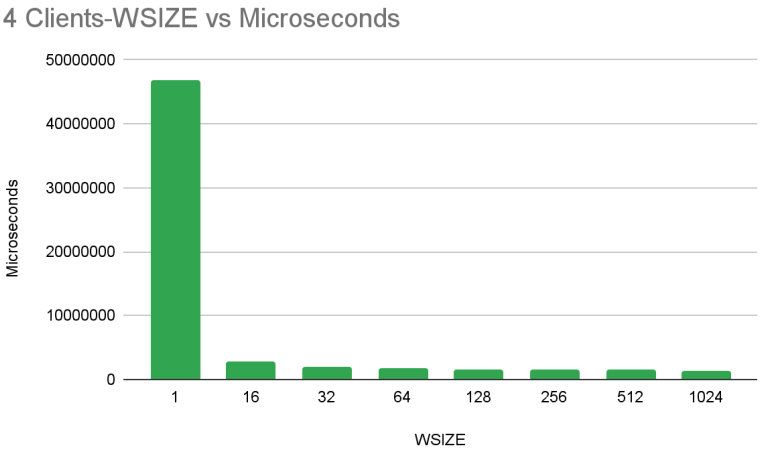


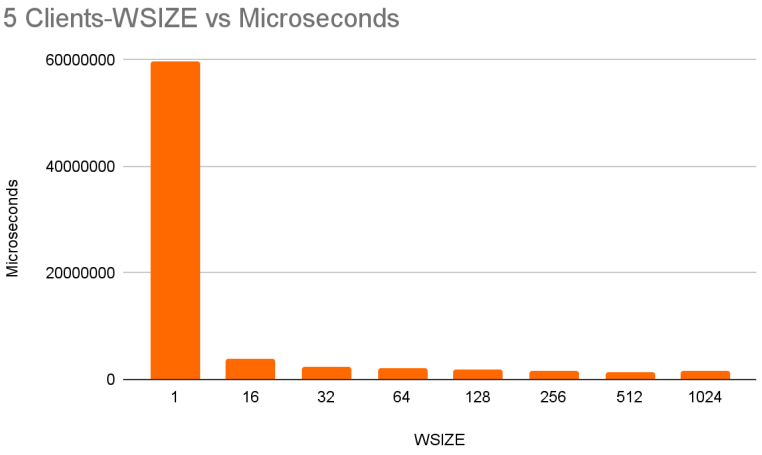
The above graph, labeled "1 Client-WSIZE vs Microseconds", shows how long it takes to process a request, measured in microseconds, for a specific size of the window (WSIZE) when there is only one client. There is a significant amount of time difference when WSIZE grows from 1 to 16, which means that performance will be remarkably improved with a tiny increase in WSIZE, which is sufficient to solve the problem. Nevertheless, the enhancement is negligible after WSIZE 16; the rest of the data set, including WSIZE 32 to 1024, shows that the time remains unchanged. That tendency means a single user does not improve considerably when the window size expands beyond a specific limit. For example, the distinctive overhead cost might be spread among larger message blocks, and the client-server system could become the limiting factor instead of gains from larger message blocks.

We repeated this experiment by increasing the number of clients and observed the same effect. You can visually examine the results of the experiments with increased client numbers with the help of the graphs below.

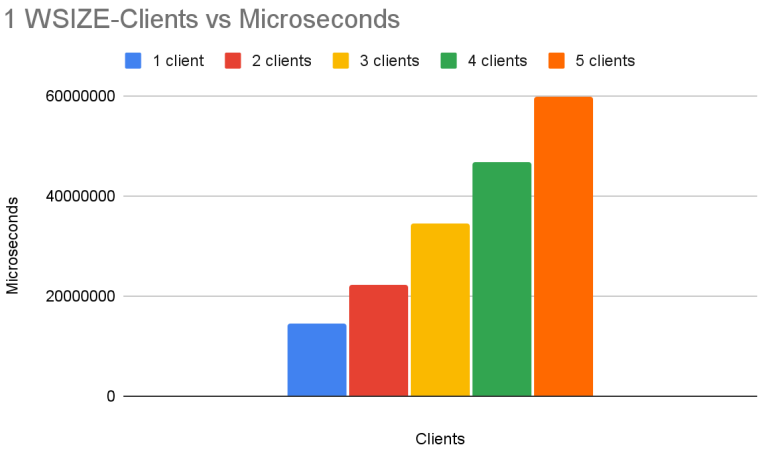

2 Clients-WSIZE vs Microseconds
25000000
20000000
15000000
10000000
5000000
0
Microseconds
1
16
32
64
128
256
512
1024
WSIZE

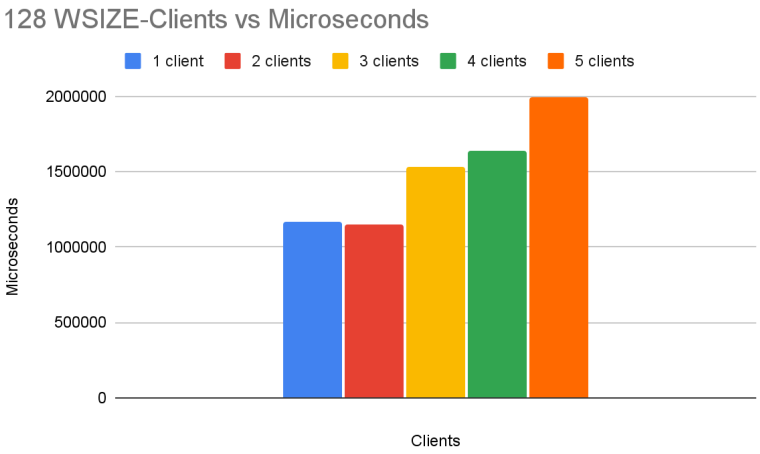

3 Clients-WSIZE vs Microseconds
40000000
30000000
20000000
10000000
0
Microseconds
1
16
32
64
128
256
512
1024
WSIZE

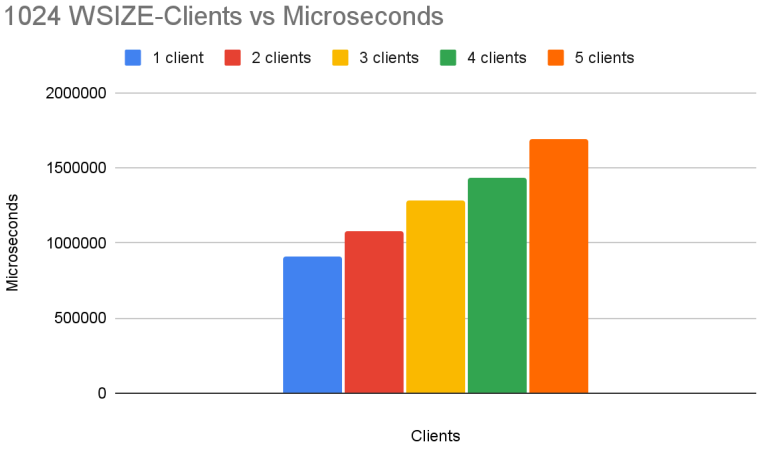

4 Clients-WSIZE vs Microseconds
50000000
40000000
30000000
20000000
10000000
0
Microseconds
1
16
32
64
128
256
512
1024
WSIZE


5 Clients-WSIZE vs Microseconds
60000000
40000000
20000000
0
Microseconds
1
16
32
64
128
256
512
1024
WSIZE

Benchmark 2: Client Number







The above graphs show the correlation between the number of clients connected to a server and the time to handle the requests in microseconds. While the number of clients has grown from just 1 to 5, there is also a corresponding increase in the time taken to handle the tasks in the given period. It thus indicates that the server faces a problem of response time degradation as more members are handled simultaneously. The time consumption tended to rise higher, going towards 4 to 5 clients, forming a possible threshold where the server faced great difficulty in managing concurrent requests. It may be due to increased users, higher Throttle, over saturation of resources, network overloading, or bottlenecks.

Conclusion

In dealing with the network transmission efficiency issue, the window size (WSIZE) versus the time for transmission is very complicated because many factors will play a role that can only be balanced depending on the situation. For instance, a bigger WSIZE is suitable for sending more stuff before a response is required, making the data sending more efficient in case of a stable data connection. The capacity to do this comes from the fact that these processes do not require complex overhead duties, and therefore, a larger WSIZE tends to reduce transmission time.

In contrast, a bigger WSIZE may lead to more packet loss if the network is busy, making retransmissions necessary. On the other hand, this could mirror the time efficiency gains of the higher WSIZE, resulting in a positive correlation where a larger WSIZE increases transmission time.

These opposite effects suggest a specific point when network inefficiencies can balance the benefits and costs of reaching a larger WSIZE. As a result, the interaction between them might influence the scenario where transmission time under differing WSIZEs is less or even non-bilinear.

An alternative benchmark focuses on the server's performance in providing services concurrently and the network's capacity to handle multiple requests. Usually, the rising volume of clients finally results in resource allocation for establishing each new connection, such as CPU time or memory and network bandwidth. This can cause latency growth because each request will be processed longer, as the server would be overloaded with the extra load. Additionally, the network might be a bottleneck if the transferred data fills every available bandwidth. The overfilled bandwidth would lead to congestion and loss of packets.