CS342 OPERATING SYSTEMS

PROJECT #3

MF SERVER LIBRARY

EXPERIMENTS REPORT

20.04.2024

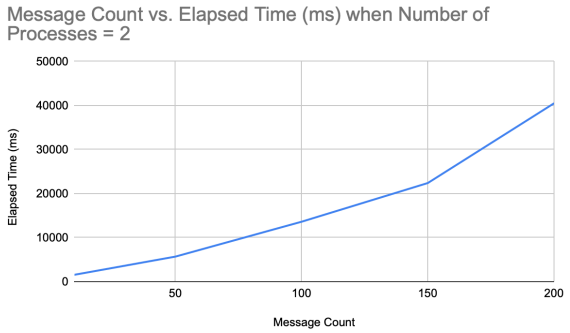
Yasemin Akın Section 002 22101782

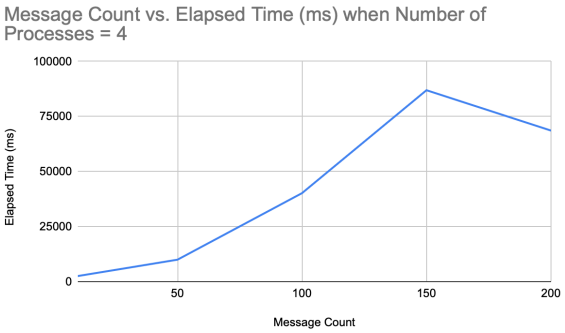
Göktuğ Serdar Yıldırım Section 002 22103111

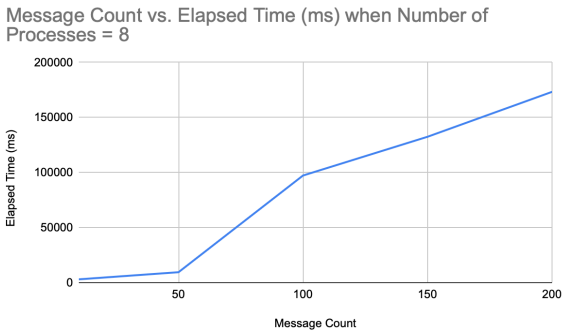
Experiment 1: Message Count and Number of Processes vs. Termination Time

The experiment was designed to evaluate how the number of processes and the volume of messages affect the performance of a messaging system. The results show a clear trend: as the number of messages per process increases, the execution time grows across the board, hinting at a linear relationship between workload and processing time. However, when more processes are added, the increase in execution time is less than proportional, indicating that the system benefits from concurrent processing but also suffers from synchronization overhead and resource contention. Particularly for higher message counts, the system's performance gain from adding more processes diminishes, suggesting there is an optimal balance between concurrency and workload that needs to be struck for efficient system operation. This balance point is key to optimizing the messaging system's throughput and ensuring that it scales effectively with the demands placed upon it.

Message Count \ Number of Processes (max_msgs_in_queue 100)	**2**	**4**	**8**
10	1548	2529	2973
50	5674	9954	9435
100	13596	40172	97262
150	22381	86920	132453
200	40479	68596	173254



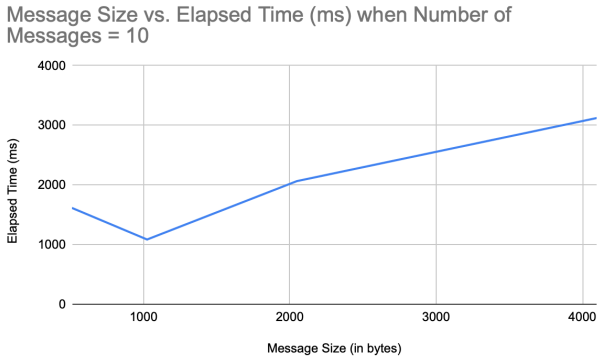


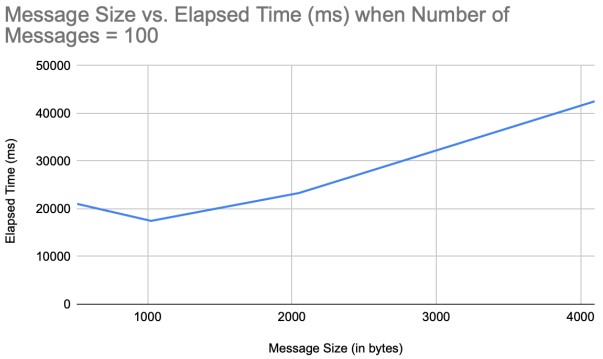


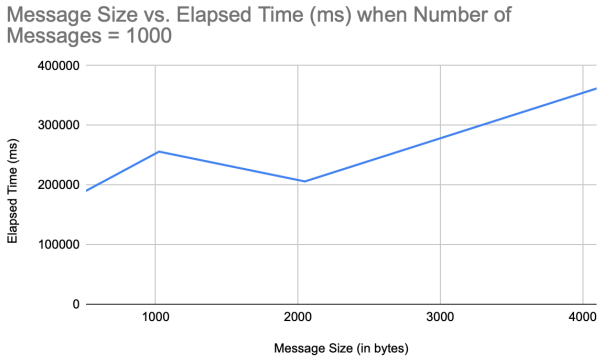
Experiment 2: Number of Messages and Message Sizes vs. Termination Time (ms) (with 1 Producer, 1 Consumer, 1 Message Queue)

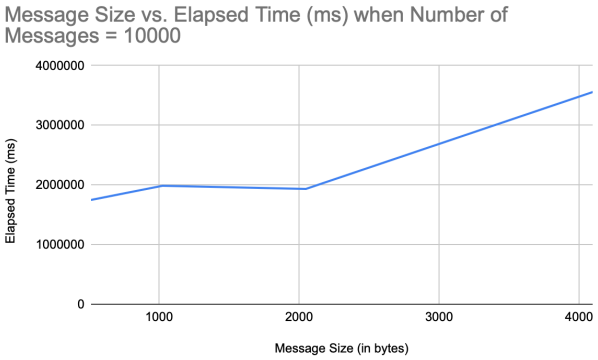
In the second experiment, the issue of the relationship between message size and the volume of communication on the performance of the system is studied, in the setup that includes one producer and one consumer working with one queue. As the size and number of messages go up, the termination time grows exponentially, implying that message size is the chief predictor of system slowness. The plunge on delay time is, however, more drastic and sharper when message size is quadrupled from 2048 bytes to 4096 bytes, as compared to when message count is scaled by the same factor. This hints that the system is better at processing more small messages in a timely manner than it is with fewer large messages. This can be attributed to the higher overheads of processing large messages such as the increase in memory access and possible bottlenecks on data transfer. These insights can be extremely helpful for optimizing the system as a whole, potentially leading to improvements in buffering, memory management or message sizes for specific workloads.

Message Size \ Number of Messages	**10**	**100**	**1000**	**10000**
512	1613	21030	190218	1747856
1024	1085	17467	255842	1985815
2048	2064	23298	205866	1932938
4096	3121	42516	361681	3556751









Experiment 3: Throughput of the Server
(Message Size vs. Maximum Messages Sent in Time Interval)

The throughput table represents how many messages were sent over one millisecond (1 msec) for the different message sizes. Test message sizes are 512, 1024, 2048, and 4096 bytes, and the number of messages sent at each size is 10, 100, and 1000.

Message Size \ Number of Messages	**10**	**100**	**1000**
512	203 micros	36msg in 1millisec	44msg in 1 millisec
1024	223 micros	28msg in 1 millisec	31msg in 1 millisec
2048	233 micros	34msg in 1millisec	39msg in 1 millisec
4096	317 micros	29msg in 1 millisec	34 msg in 1 millisec

So, for the minimum message size, which was 512 bytes, the server took 203 microseconds to send the 10 messages. As the number of messages increased to 100 and 1000, the server could send 36 and 44 messages within one millisecond, respectively. When the size of the message is doubled from 512 bytes to 1024 bytes, the time taken to send ten messages increases from 203 to 223 microseconds. With higher loads (100 and 1000 messages), the server can send more messages within a one-millisecond duration of 28 and 31, respectively, compared to the 512-byte size.

Further increase of message size to 2048 bytes shows growth in time for sending ten messages to 233 microseconds. The server throughput for 100 messages remains the same, with 34 messages within one millisecond. One thousand messages in, and the throughput slightly drops to 39 in the millisecond window. Lastly, with the most significant message size tested at 4096 bytes, a proportionally higher time of 317 microseconds is recorded. It suggests that the processing time for message sizes has increased.

However, when scaled to sending loads of 100 and 1000 messages, the throughput hits 29 and 34 messages within a millisecond window, showing a decrease in throughput with an increase in message size. These results indicate that the server can send smaller messages faster than more significant messages, as one would expect, because less data is being sent. However, in 100 and 1000 messages, the throughput does not reduce linearly as the size of the message increases.

Suggestive possible optimizations or buffering effects that allow the server to handle larger loads more effectively up to a certain point. It was observed that there is a message size threshold above which throughput starts leveling off or even dropping, showing the message size that lets users understand the performance characteristics and possible bottlenecks in data handling capacity.