Lab6 Summary

Group: TorFAndBrynjar

19.04.2021

Implementation:

Batching was implemented by adding a list that will contain client requests and a batching delay variable setting the time limit for collecting requests before sending to the proposer. The list and time variable are added to the main application. When the server receives a request from the client that is a bank transaction it is added to the list of requests. A timer is added and a channel that receives every time interval. If the list is not empty at the end of the time interval, then the leader will send the list of requests to its proposer.

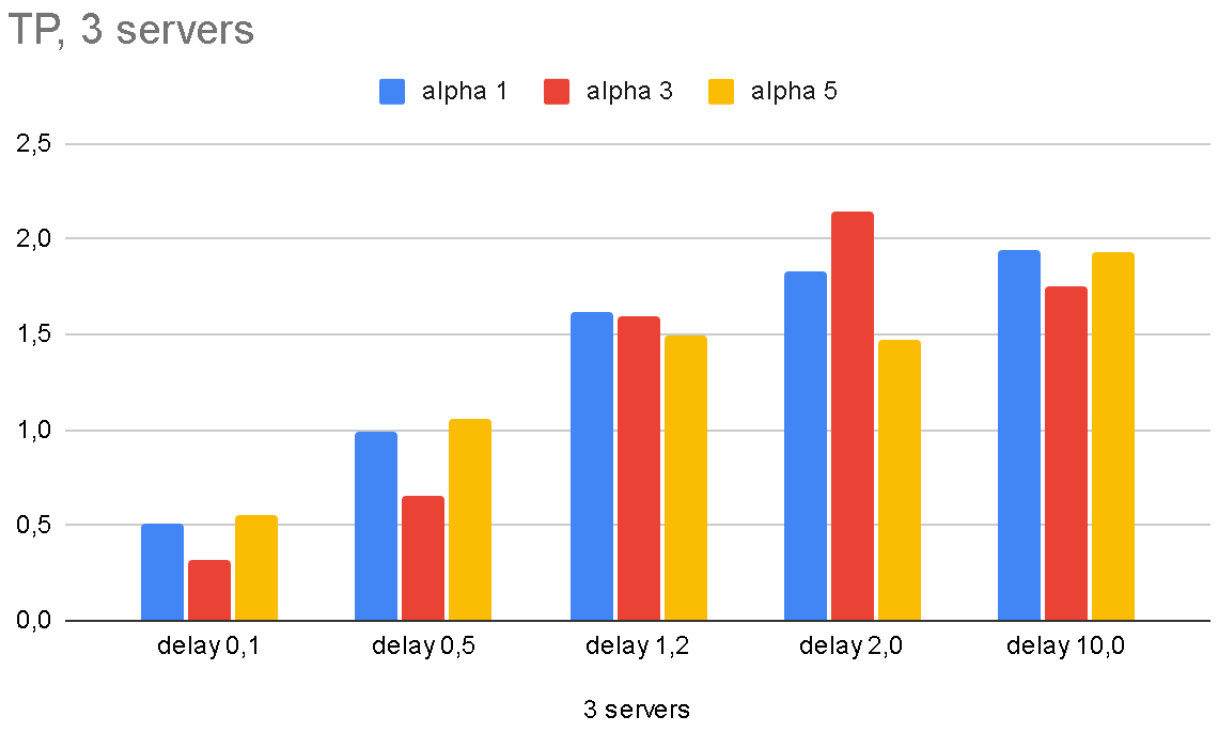
For pipelining the alpha variable was extracted from the SendAccept function to the proposer struct. A for loop is introduced in the sendAccept function. The loop goes from 1 to alpha. As long as proposer next slot is not smaller or equal to alpha it will process one more batch of requests from the request in que.

Test conditions:

We tested with 3 and 5 servers, and two clients. Client0 was a load client which primary goal was create a load in the system. We did that by creating a function that sent messages out evenly in a steady rhythm. Client1 was our testclient. And for each message client1 sent out we started a timer. Then when the response came back we stopped the timer for that message. This allowed us to calculate average RoundTripTime. This process was started with a starttest message. When the server received this message it started a timer and counted all requests that went through paxos during the test time. This gave us the possibility of calculating throughput. We used the numbers coming from Client1 and the lead server in our datacollection. We ran 30 tests with varying time and alpha. 15 tests with 3 servers and 15 tests with 5 servers.

3 Servers Throughput:

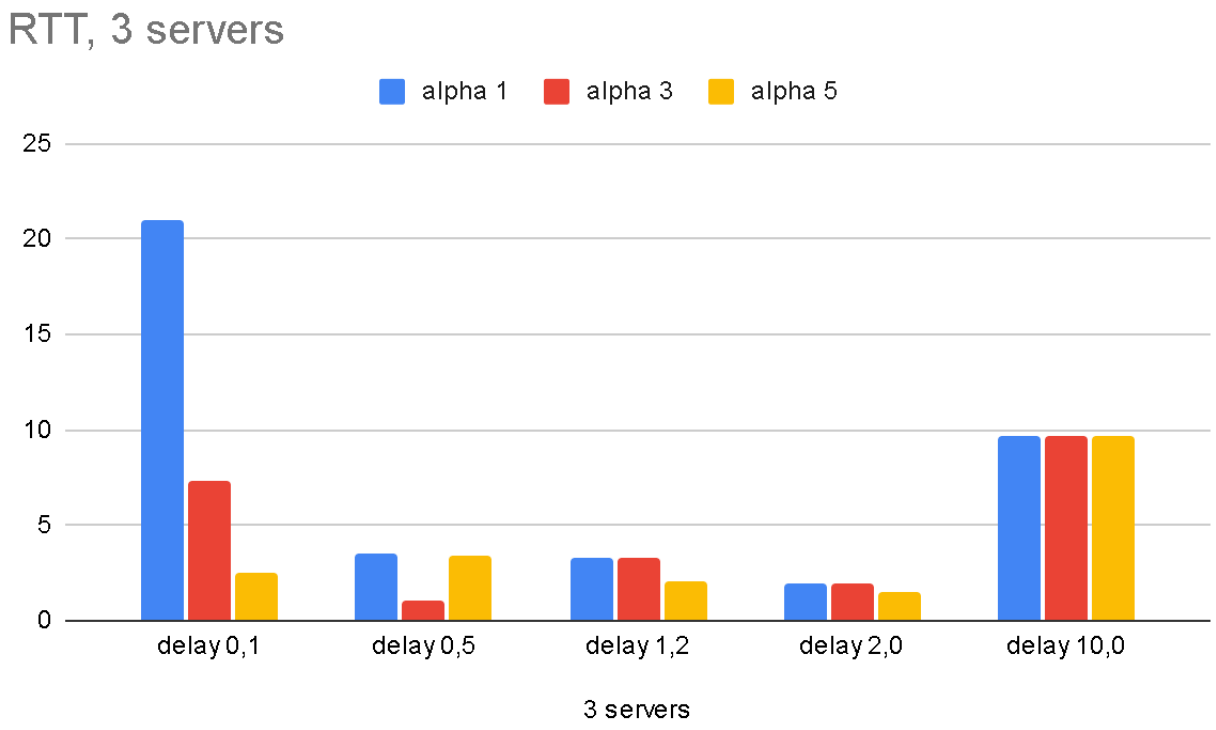
			Throughput	requests per second			
	Alpha						Mean alphas
Batch time		0,1	0,5	1,2	2	10	
	1	0,51	0,99	1,62	1,83	1,94	1,378
	3	0,32	0,66	1,6	2,14	1,75	1,294
	5	0,55	1,06	1,5	1,47	1,93	1,302
	Mean time	0,46	0,9033333333	1,573333333	1,813333333	1,873333333	
	Min:	0,32					
	Max:	2,14					
	Median:	1,5					
	standard deviation:	0,595097431					
	90.Percentile	1,936					
	Mean Total	1,324666667					



As can be seen in the table above and the graph the average throughput was 1,3 requests per second. While the max was 2,14. This came when we had an alpha of 3 and 2 seconds batch time.

3 Servers RTT:

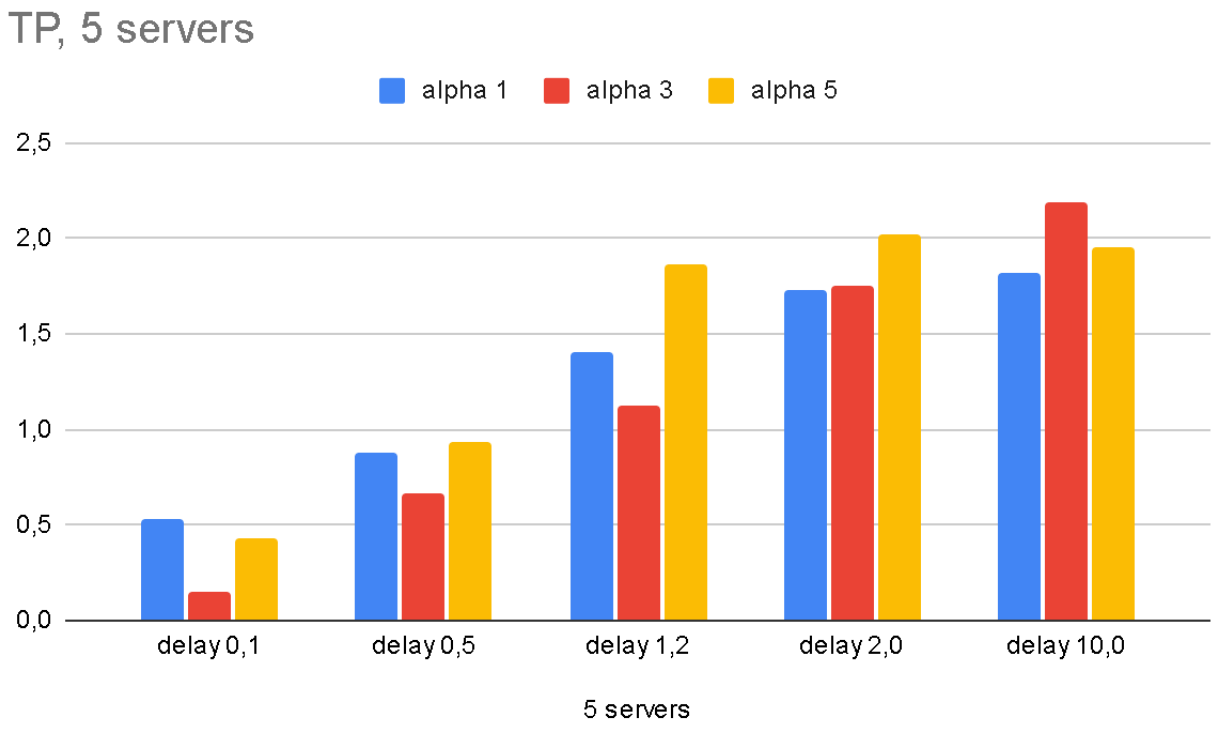
		RTT				
	0.1	0,5	1,2	2.0	10	Mean alphas
1	21	3,55	3,26	1,9	9,7	6.1
3	7,3	1,04	3,33	2	9,7	3.1
5	2,55	3,45	2,12	1,47	9,7	2.1
Mean time:	10,28333333	2,68	2,903333333	1,79	9,7	
Min:	1,04					
Max:	21					
Median:	3,33					
standard deviatio	5,315735222					
90.Percentile	9,7					
Mean Total	5,471333333					



Unsurprisingly the RTT and the batch time had a correlation when the batch time was over 2 seconds. When the batch time was under 2 seconds we noticed that the paxos algorithm struggled to keep up and this the RTT went up. Here we got our worst test with an average of 21 seconds RTT for alpha 1, batch time 0,1 seconds.

5 Servers Throughput:

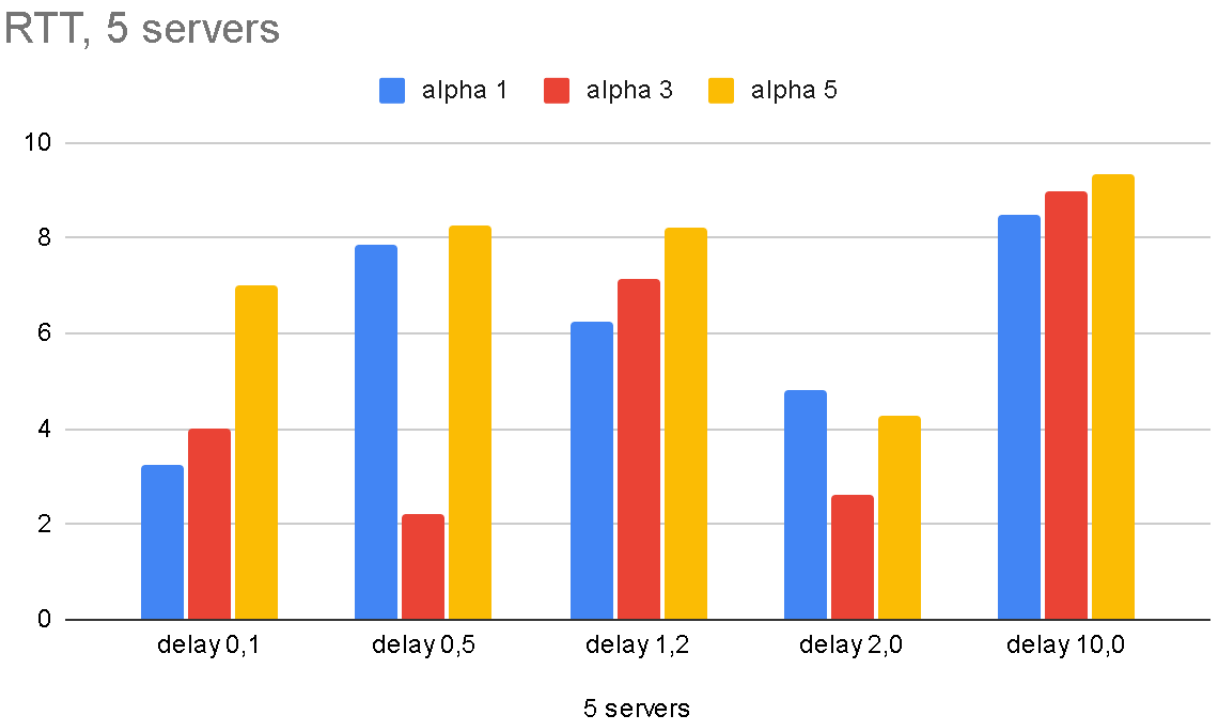
			Throughput	requests per second			
	Alpha						Mean alphas
Batch time		0.1	0.5	1.2	2.0	10	
	1	0,53	0,88	1,41	1,73	1.82	1,1375
	3	0,15	0,67	1,13	1,75	2,19	1,178
	5	0,43	0,93	1,86	2,02	1,95	1,438
	Mean time	0,37	0,8266666667	1,466666667	1,833333333	2,07	
	Min:	0,15					
	Max:	2,19					
	Median:	1,27					
	standard deviation:	0,669919217					
	90.Percentile	1,999					
	Mean Total	1,259285714					



For 5 servers it was the same story as for three. With the best results with a batch time of 2 or higher.

5 Servers RTT:

Batch time	Alpha		RTT				Mean alphas
		0.1	0.5	1.2	2.0	10	
	1	3,26	7,85	6,25	4,83	8,5	6,138
	3	4	2,2	7,125	2,64	9	4,993
	5	7	8,28	8,22	4,26	9,33	7,418
	Mean time	4,753333333	6,11	7,198333333	3,91	8,943333333	
	Min:	2,2					
	Max:	9,33					
	Median:	7					
	standard deviatio	2,439960333					
	90.Percentile	8,8					
	Mean Total	6,183					



What seems clear after these tests is that a batch time between 2 and ten seconds should be the best choice, and we should prevent going lower than 2. We should also prevent going higher than 10 as that would increase the RTT as well resulting in a system that would be just to slow to use.

Summary:

As we can see throughput increases when we increase the delay and alpha is at 3. The problem with increasing the delay up to 10 seconds is that the throughput doesn't increase by a large amount while the RTT follows the linearly after 2 seconds thus the system will take longer to respond. We found with these tests that the optimal solution was alpha 3 and batch delay 2 seconds.