HW 1 Report - Linni Cai

Github URL:

https://github.com/linni-cai-lc/CS6650_Distributed_System/tree/main/hw1

Statistics URL:

https://docs.google.com/spreadsheets/d/1VILwSgj6rAozlp8ojxaa1y2lt3YFVT8SVxpUyqiuPo4/edit?usp=sharing

Client Design:

Part 1:

- ClientMultiThreaded
    - main program
    - get parsed arguments from command line arguments, store them locally
    - calculate num_threads, num_posts, num_trigger for phase1, phase2, phase3
    - initialize safe counter for num_success and num_unsuccess
    - initialize sync tool for phase2, phase3, and total completeness
    - create phase1, phase2, phase3
    - timer on
    - start phase1, phase2, phase3
    - timer off
    - print statistics
- ClientSingleThread
    - main program
    - only runs 10000 POST request to get latency
    - run with one thread
- CommandLineParser
    - command-line arguments parsing tool
    - distinguish different flags and store their values locally
- Phase
    - phase properties storage
    - build a runnable phase and count down the completeness
    - increment the counter for success and unsuccess
    - generate random candidates and call POST

Part 2:

- ClientMultiThreaded
    - main program
    - get parsed arguments from command line arguments, store them locally

        - calculate num_threads, num_posts, num_trigger for phase1, phase2, phase3
        - initialize safe counter for num_success and num_unsuccess
        - initialize sync tool for phase2, phase3, and total completeness
        - create phase1, phase2, phase3
        - timer on
        - start phase1, phase2, phase3
        - timer off
        - print statistics for part 1
        - print out all records into a CSV file
        - print statistics for part 2
- ClientMultiThreadedSinglePhase
    - main program
    - phase 2 only client
- CommandLineParser
    - command-line arguments parsing tool
    - distinguish different flags and store their values locally
- Phase
    - phase properties storage
    - build a runnable phase and count down the completeness
    - increment the counter for success and unsuccess
    - generate random candidates and call POST
    - store information a record list
- Calculator
    - calculate statistics for records
- CSVProcessor
    - write the given records into a CSV file with a header

Calculation:

Simple test:

With ClientSignleThread, I calculated latency from the client with one thread and 10000 requests.

- latency = (196341/1000)/10000 = 0.0196

I calculated the predicted throughput for different numbers of threads, latency and mean_response are closer to each other as well.

- predicted throughput (latency) = num_threads / latency
- predicted throughput (mean response) = num_threads / mean_response

Part 1:

The following data and graph are for part 1, with more threads, the runtime is lower and throughput is larger, which means faster.

Part 2:

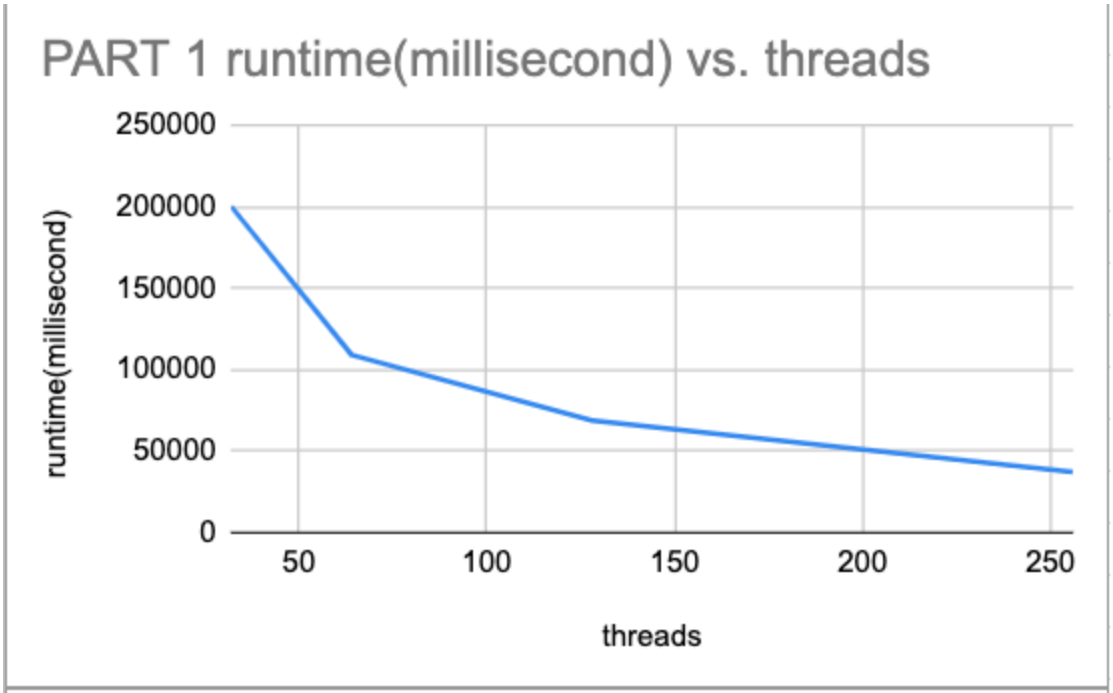
The following data and graph are for part 2, with more threads, the throughput is higher, and the runtime is lower, however, the response time is higher. To apply Little's Law, the throughput prediction is higher than the actual throughput, as we can see, when the number of threads is 32 and 64, phase 2 only case owns closer throughput with the prediction. However, for more threads, the difference becomes larger. It is reasonable that phase2 runs with the all number of threads, phase1 runs 25%, phase3 runs 10%. Since phase 1 and phase 3 don't use all threads, the overall throughput is affected as well. As the number of threads increases, the response time increases due to busy service.

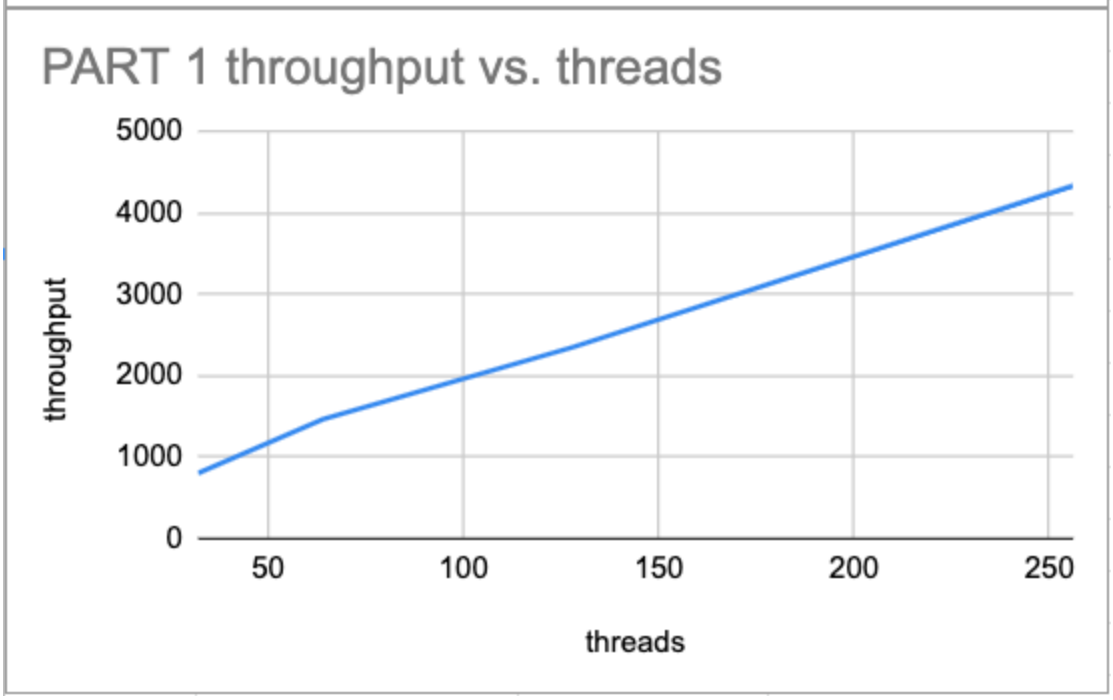
part 0		
threads	runtime(millisecond)	latency
1	196341	19.6341

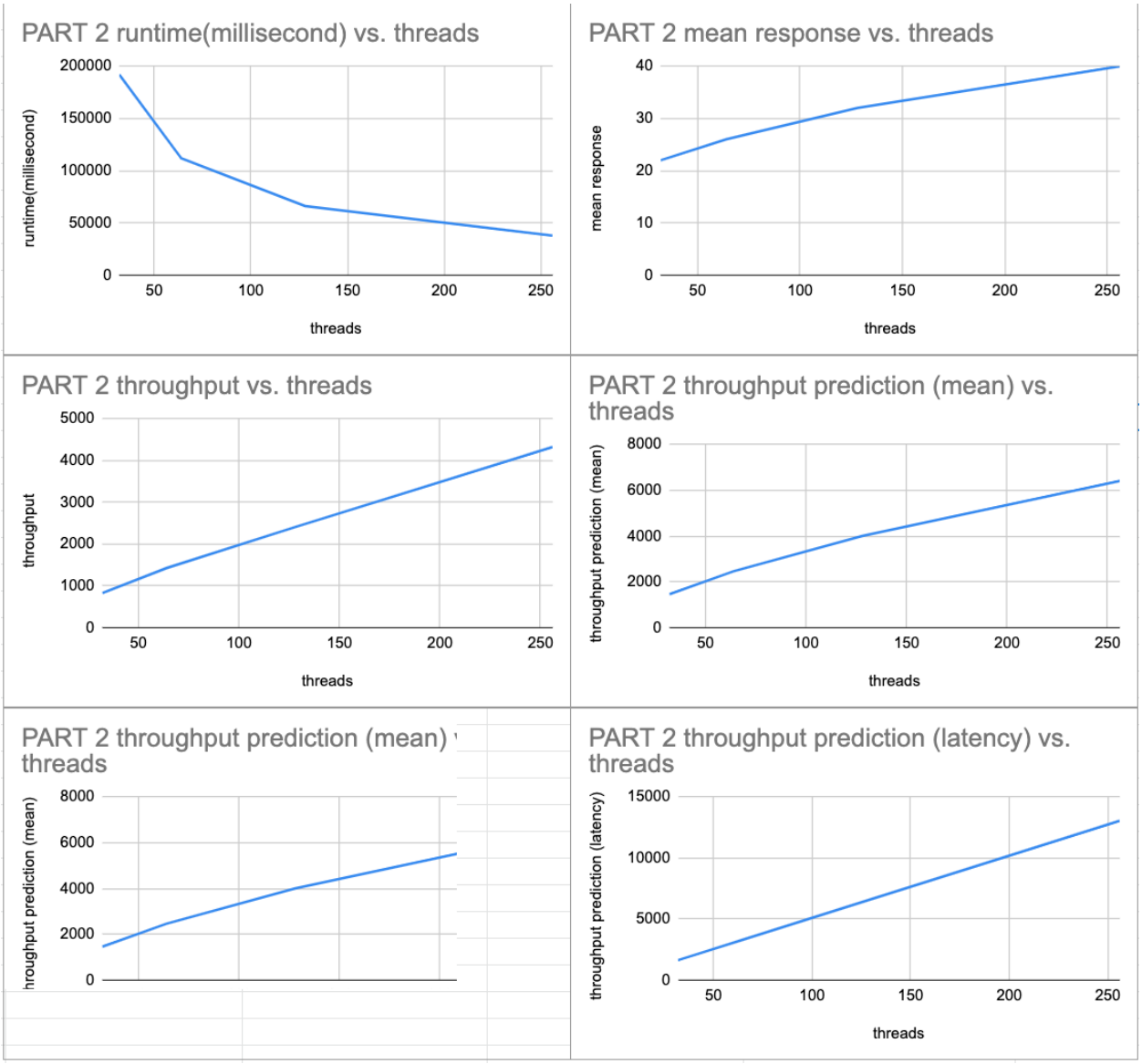
part 1		
threads	runtime(millisecond)	throughput
32	200488	800
64	109228	1466
128	68815	2352
256	37162	4335

part 2								
threads	runtime(millisecond)	mean response	throughput (all phases)	throughput (phase 2 only)	throughput prediction (mean)	throughput prediction (latency)	throughput prediction diff (mean)	throughput prediction diff (latency)
32	192099	22	825	1578	1455	1630	76.31%	97.55%
64	111941	26	1424	3284	2462	3260	72.86%	128.91%
128	66086	32	2405	3822	4000	6519	66.32%	171.07%
256	37736	40	4323	3479	6400	13039	48.05%	201.61%

part 1 & part 2 comparison	
runtime diff	throughput diff
4.18%	3.13%
2.48%	2.86%
3.97%	2.25%
1.54%	0.28%


PART 1 runtime(millisecond) vs. threads
250000
200000
150000
100000
50000
0
runtime(millisecond)
50
100
150
200
250
threads


PART 1 throughput vs. threads
5000
4000
3000
2000
1000
0
throughput
50
100
150
200
250
threads



Simple:

- --num_threads **1** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080
  ------ PHASE 1 ------
  ------ Statistics ------
  number of successful requests sent: 10000
  number of unsuccessful requests: 0
  the total run time for all phases to complete: **196341**
  the total throughput in requests per second: 51

Part 1:

- --num_threads **32** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080

------ Statistics ------
number of successful requests sent: 160016
number of unsuccessful requests: 0
the total run time for all phases to complete: **200488**
the total throughput in requests per second: 800

- --num_threads **64** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080
  ------ Statistics ------
  number of successful requests sent: 159857
  number of unsuccessful requests: 0
  the total run time for all phases to complete: **109228**
  the total throughput in requests per second: 1466

- --num_threads **128** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080
  ------ Statistics ------
  number of successful requests sent: 159977
  number of unsuccessful requests: 0
  the total run time for all phases to complete: **68815**
  the total throughput in requests per second: 2352

- --num_threads **256** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080
  ------ Statistics ------
  number of successful requests sent: 160420
  number of unsuccessful requests: 0
  the total run time for all phases to complete: **37162**
  the total throughput in requests per second: 4335

Part 2:

- --num_threads **32** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080

  ALL PHASES:
  ------ Statistics ------

  ------ PART 1 ------
  number of successful requests sent: 160016
  number of unsuccessful requests: 0
  the total run time for all phases to complete: **192099**
  the total throughput in requests per second: 833

  ------ PART 2 ------
  mean response time (millisecs): **22**
  median response time (millisecs): 16
  throughput: **825**
  p99 (99th percentile) response time: 123
  min response time (millisecs): 10
  max response time (millisecs): 503

PHASE 2 ONLY:
------ Statistics ------

------ PART 1 ------
number of successful requests sent: 120000
number of unsuccessful requests: 0
the total run time for all phases to complete: 75741
the total throughput in requests per second: 1600

------ PART 2 ------
mean response time (millisecs): 20
median response time (millisecs): 16
throughput: **1578**
p99 (99th percentile) response time: 139
min response time (millisecs): 10
max response time (millisecs): 858

- --num_threads **64** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080

ALL PHASES:
------ Statistics ------

------ PART 1 ------
number of successful requests sent: 159857
number of unsuccessful requests: 0
the total run time for all phases to complete: **111941**
the total throughput in requests per second: 1440

------ PART 2 ------
mean response time (millisecs): **26**
median response time (millisecs): 18
throughput: **1424**
p99 (99th percentile) response time: 153
min response time (millisecs): 10
max response time (millisecs): 1097

PHASE 2 ONLY:
------ Statistics ------

------ PART 1 ------
number of successful requests sent: 119808
number of unsuccessful requests: 0
the total run time for all phases to complete: 36783
the total throughput in requests per second: 3328

------ PART 2 ------
mean response time (millisecs): 19
median response time (millisecs): 16

throughput: **3284**
p99 (99th percentile) response time: 78
min response time (millisecs): 10
max response time (millisecs): 1417

- --num_threads **128** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080

  ALL PHASES:
  ------ Statistics ------

  ------ PART 1 ------
  number of successful requests sent: 159977
  number of unsuccessful requests: 0
  the total run time for all phases to complete: **66086**
  the total throughput in requests per second: 2423

  ------ PART 2 ------
  mean response time (millisecs): **32**
  median response time (millisecs): 23
  throughput: **2405**
  p99 (99th percentile) response time: 234
  min response time (millisecs): 10
  max response time (millisecs): 797

  PHASE 2 ONLY:
  ------ Statistics ------

  ------ PART 1 ------
  number of successful requests sent: 119808
  number of unsuccessful requests: 0
  the total run time for all phases to complete: 31192
  the total throughput in requests per second: 3864

  ------ PART 2 ------
  mean response time (millisecs): 32
  median response time (millisecs): 23
  throughput: **3822**
  p99 (99th percentile) response time: 218
  min response time (millisecs): 11
  max response time (millisecs): 2495

- --num_threads **256** --num_skiers 20000 --num_lifts 40 --ip_address 54.200.234.195:8080

  ALL PHASES:
  ------ Statistics ------

------ PART 1 ------
number of successful requests sent: 160420
number of unsuccessful requests: 0
the total run time for all phases to complete: **37736**
the total throughput in requests per second: 4335

------ PART 2 ------
mean response time (millisecs): **40**
median response time (millisecs): 26
throughput: **4323**
p99 (99th percentile) response time: 404
min response time (millisecs): 10
max response time (millisecs): 2191

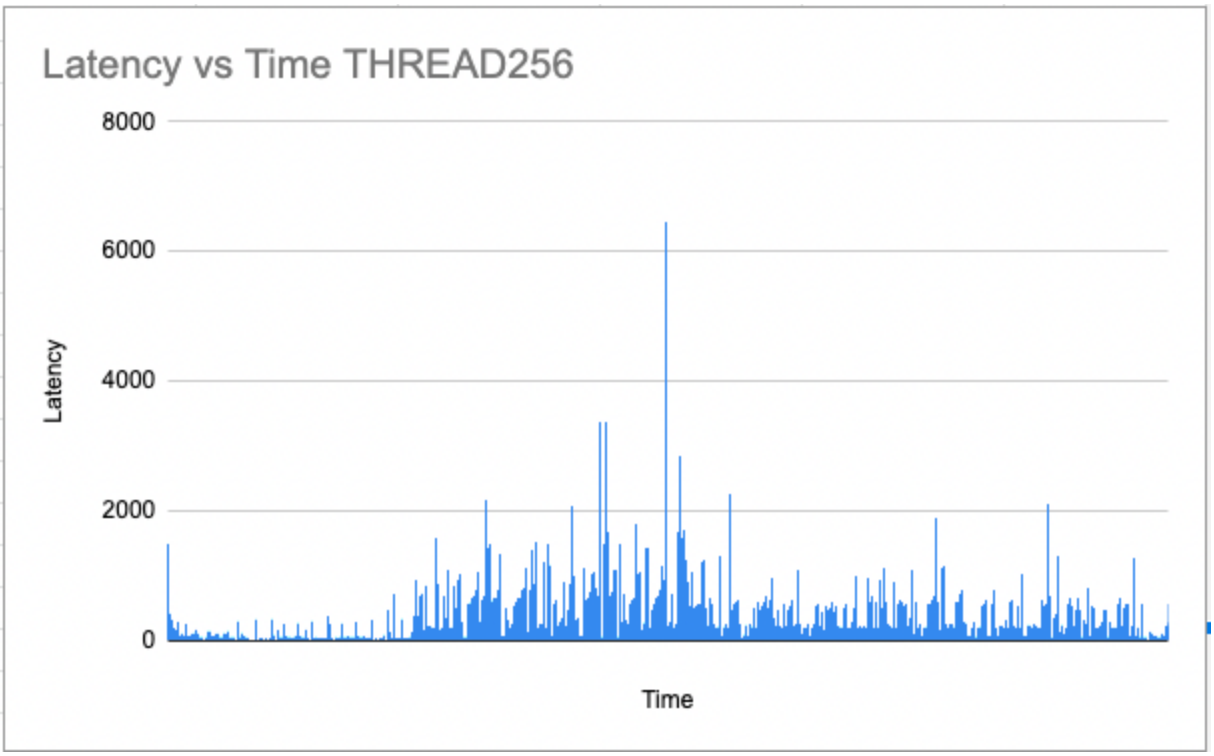
PHASE 2 ONLY:
------ Statistics ------

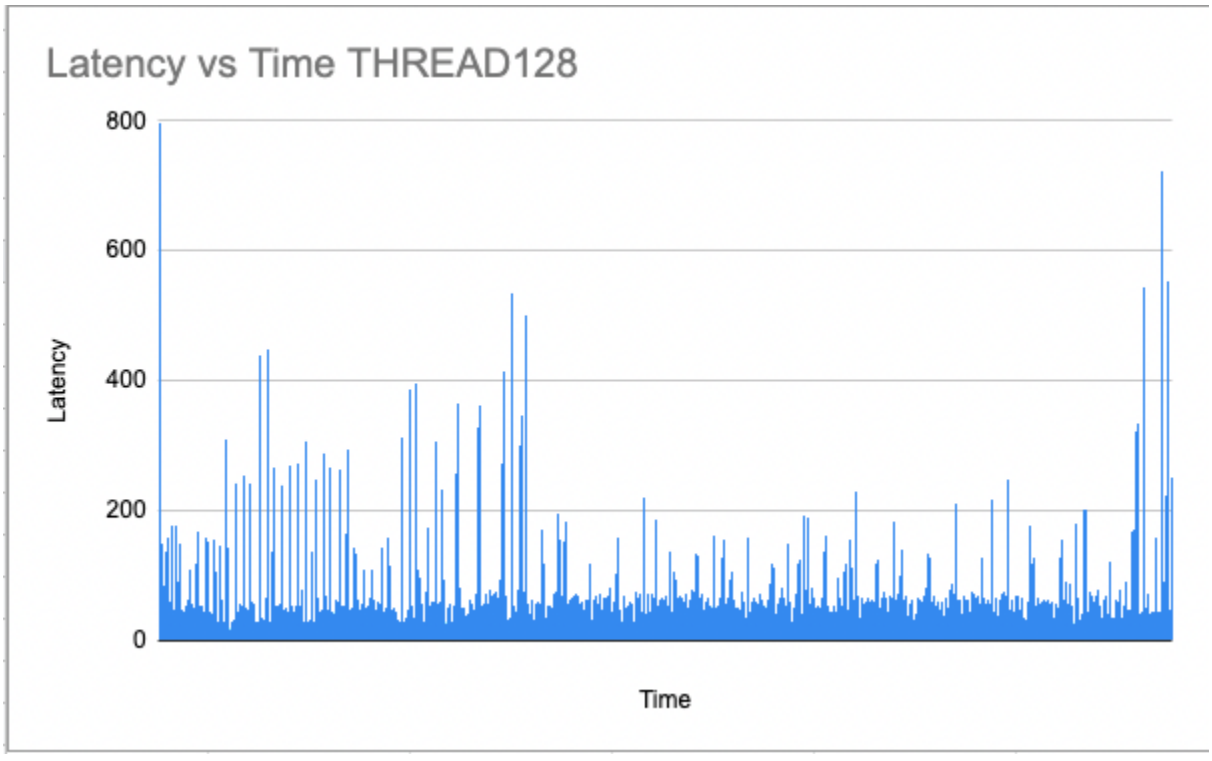
------ PART 1 ------
number of successful requests sent: 119808
number of unsuccessful requests: 0
the total run time for all phases to complete: 34146
the total throughput in requests per second: 3523

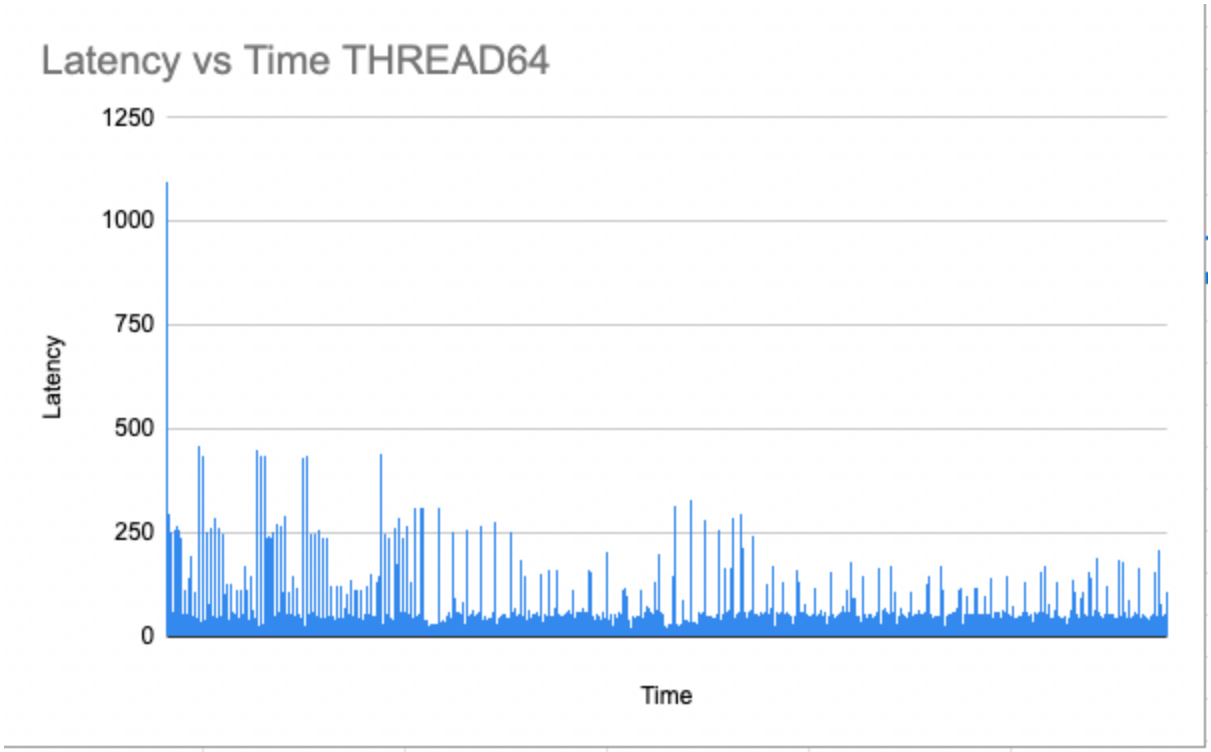
------ PART 2 ------
mean response time (millisecs): 69
median response time (millisecs): 27
throughput: **3479**
p99 (99th percentile) response time: 514
min response time (millisecs): 11
max response time (millisecs): 4255

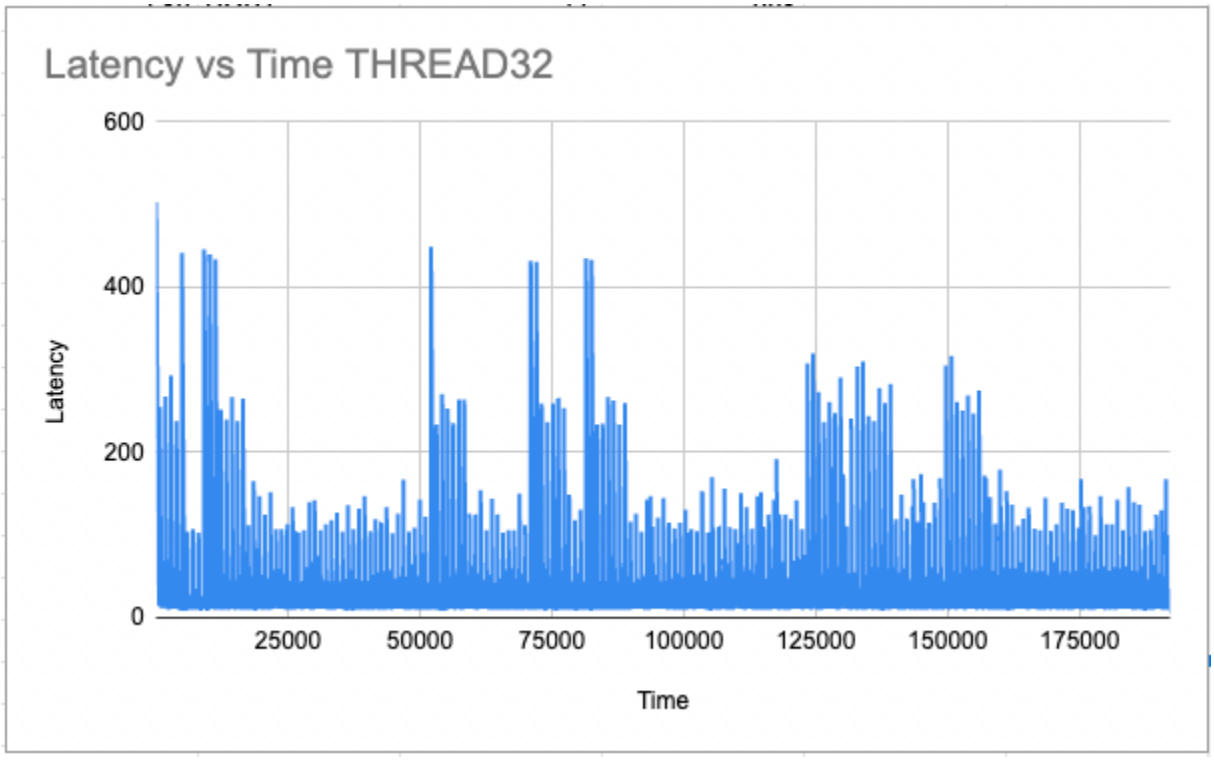
Bonus:

Plot for latency against time for thread numbers from 256, 128, 64, 32


Latency vs Time THREAD256
8000
6000
4000
2000
0
Latency
Time

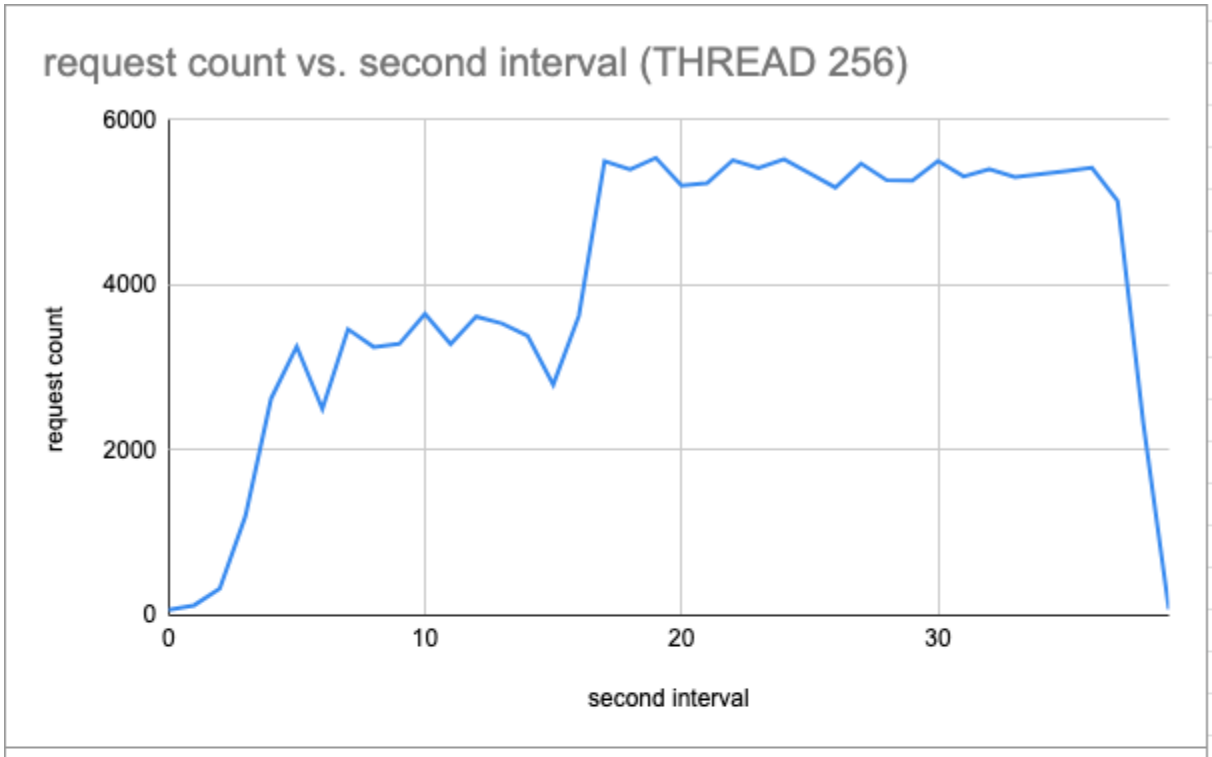

Latency vs Time THREAD128
800
600
400
200
0
Latency
Time

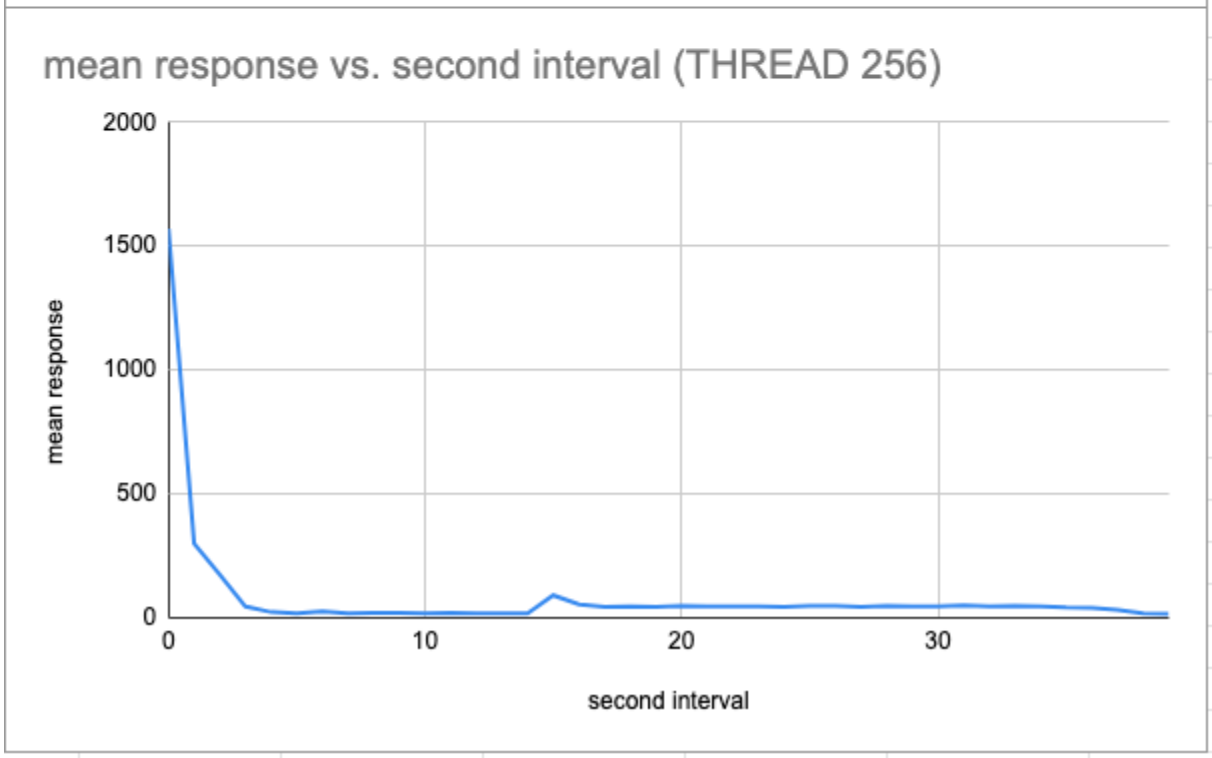

Latency vs Time THREAD64
1250
1000
750
500
250
0
Latency
Time

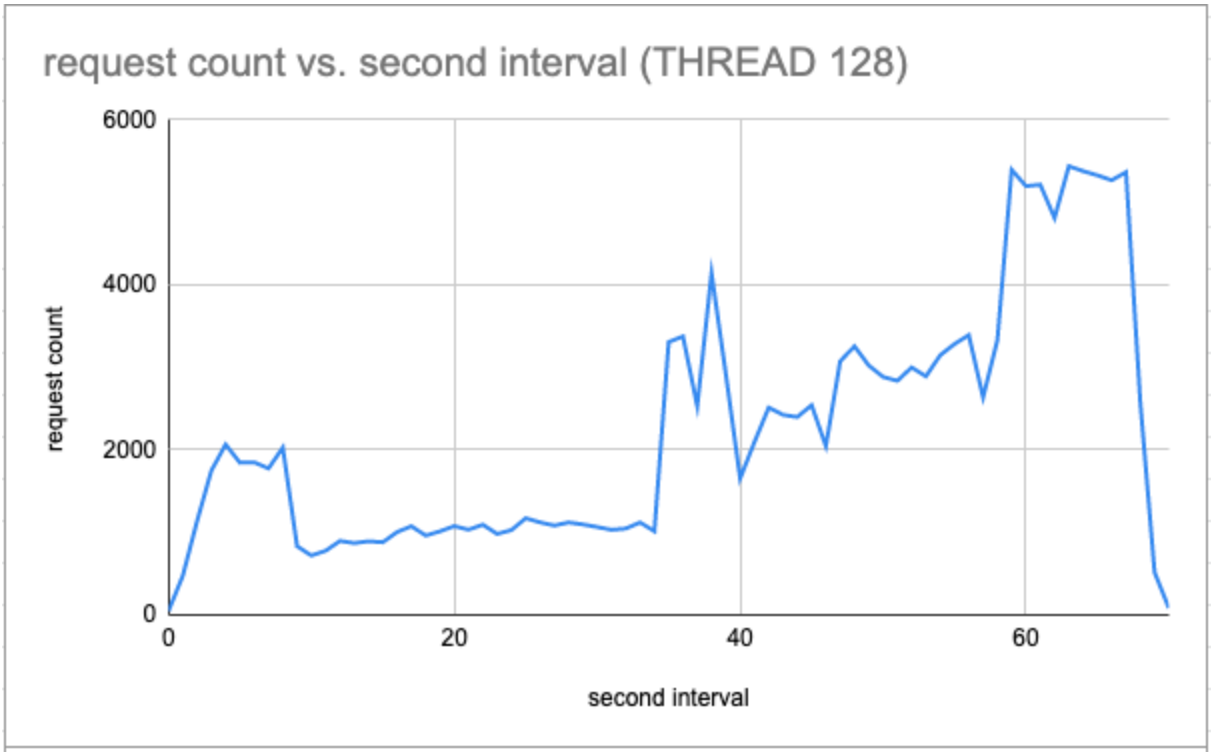

Latency vs Time THREAD32
600
400
200
0
Latency
25000
50000
75000
100000
125000
150000
175000
Time

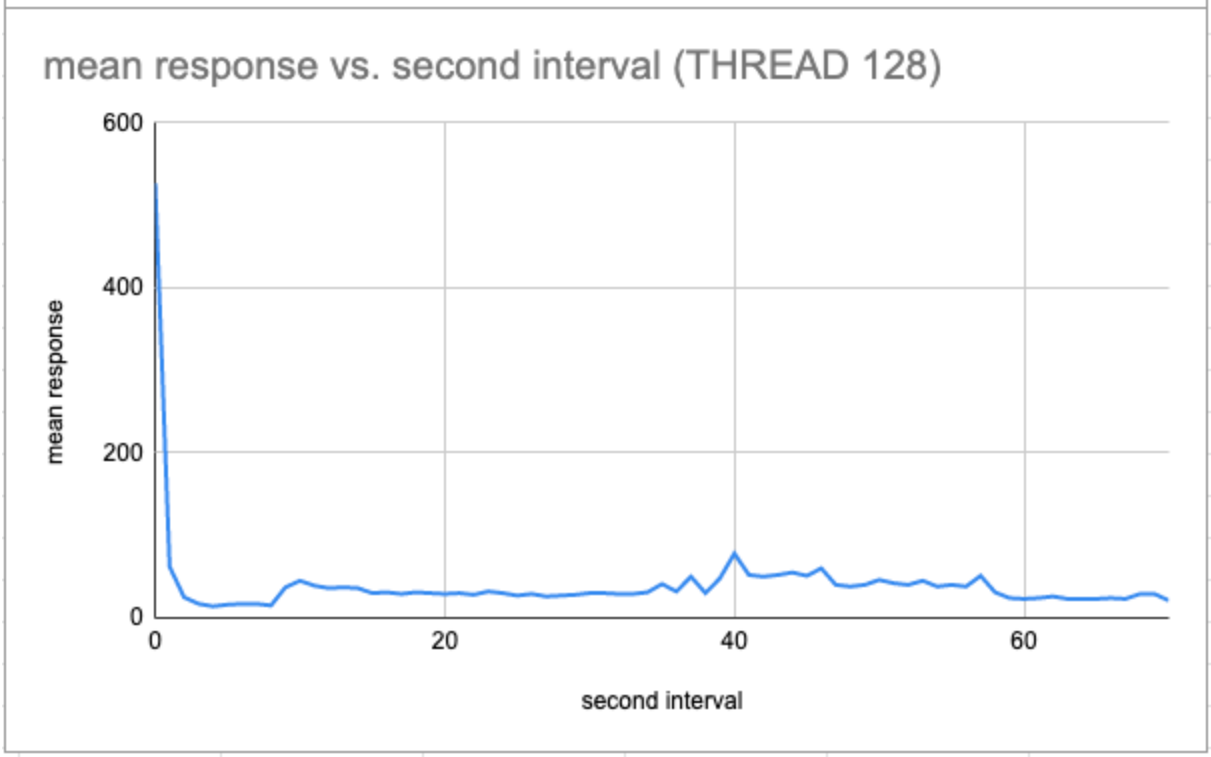
Plots for requests count against the second interval for thread numbers from 256, 128, 64, 32

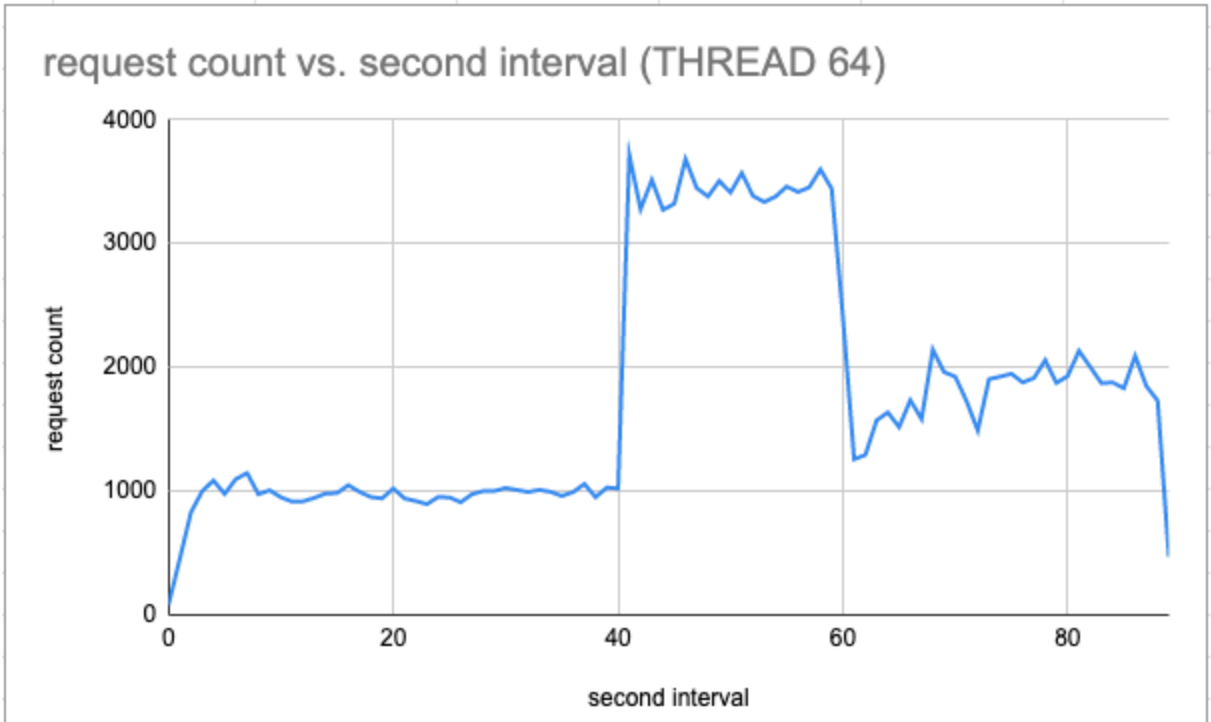
Plots for mean response against the second interval for thread numbers from 256, 128, 64, 32

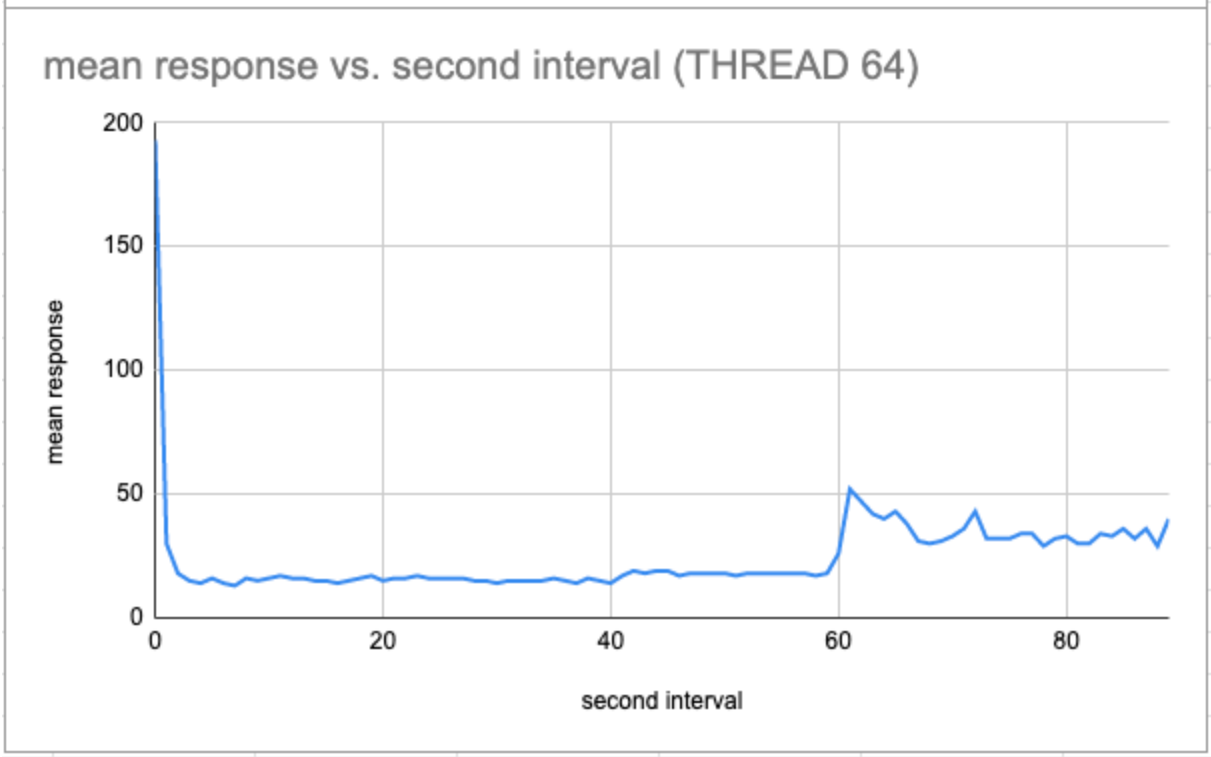


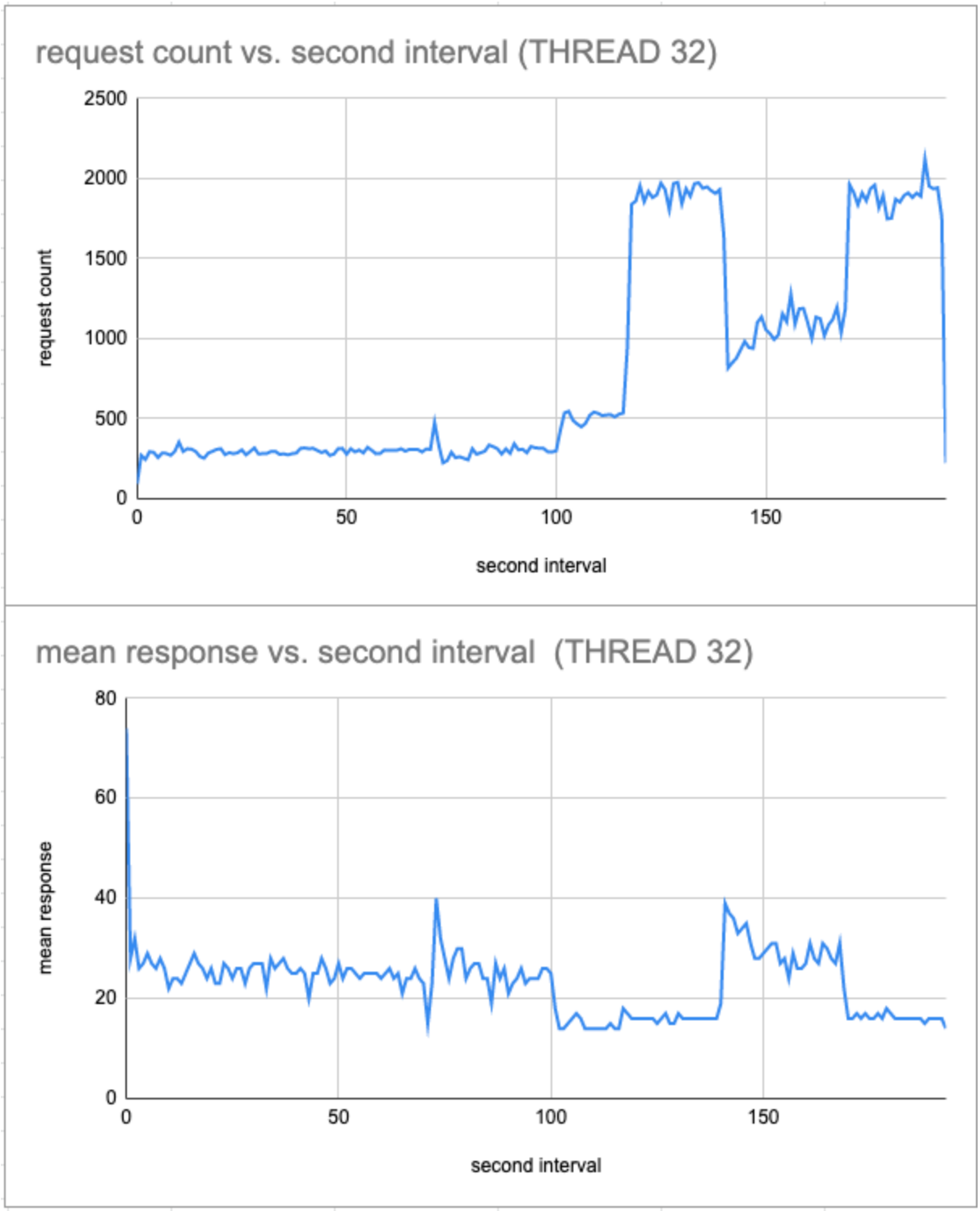



request count vs. second interval (THREAD 128)
6000
4000
2000
0
request count
0
20
40
60
second interval


mean response vs. second interval (THREAD 128)
600
400
200
0
mean response
0
20
40
60
second interval


request count vs. second interval (THREAD 64)
request count
4000
3000
2000
1000
0
0
20
40
60
80
second interval


mean response vs. second interval (THREAD 64)
mean response
200
150
100
50
0
0
20
40
60
80
second interval


request count vs. second interval (THREAD 32)
request count
2500
2000
1500
1000
500
0
0
50
100
150
second interval
mean response vs. second interval (THREAD 32)
mean response
80
60
40
20
0
0
50
100
150
second interval