Discrete Event Simulation of a Web Application

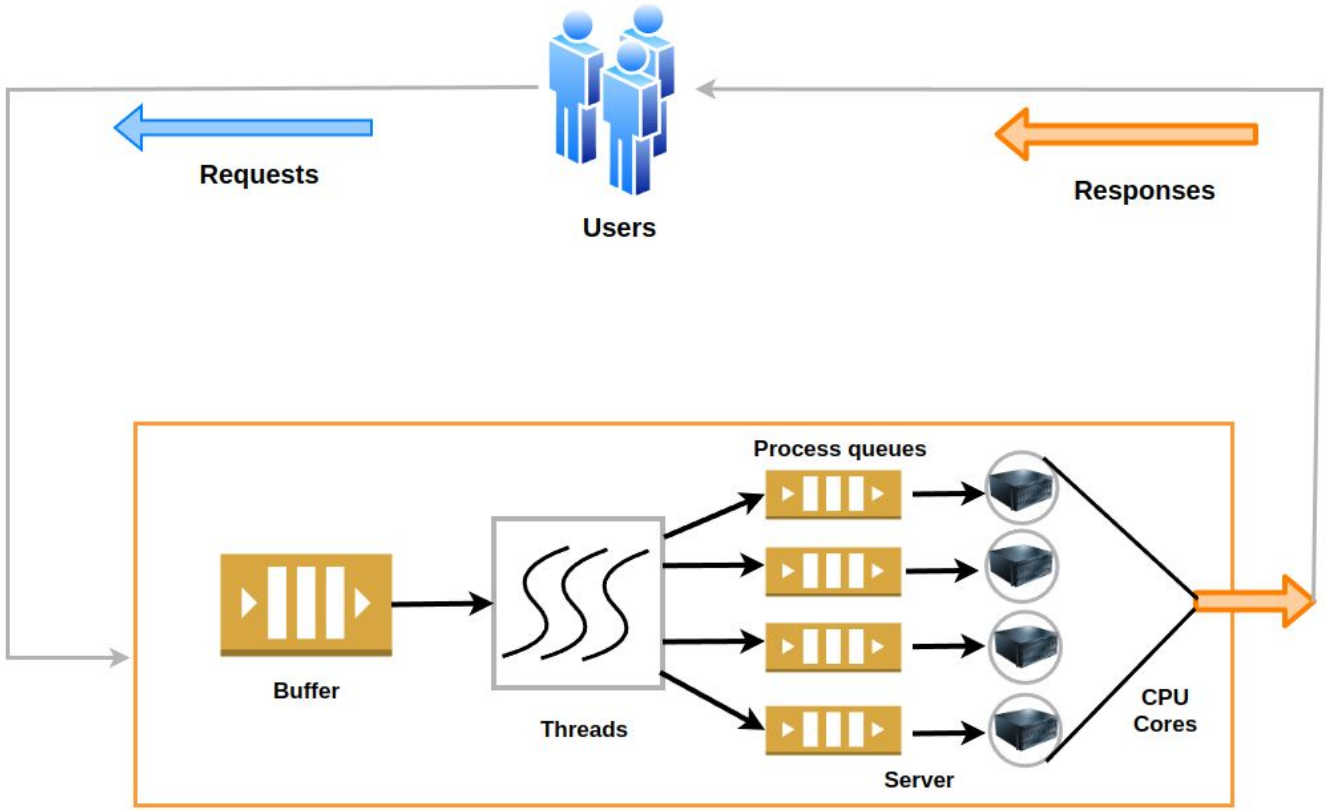
CS681-Performance Analysis of Computer Systems and Networks

Naradasu Hemanth Kumar - 22M0777

Web server system characteristics

- Multi-server machine .
- Multi-threaded web server - once a thread is assigned to a core it remains in the same core until the request it handles completes processing.
- Thread per task model - Each incoming request is assigned a free thread.
- Round Robin Scheduling with context switch overhead.
- Request timeouts and retry after timeout.
- Timeout time has a minimum value and then a variable value chosen from a distribution.
- Various probability distributions for execution time, think time and time out.
    - Uniform probability distribution
    - Constant values
    - Exponential probability distribution

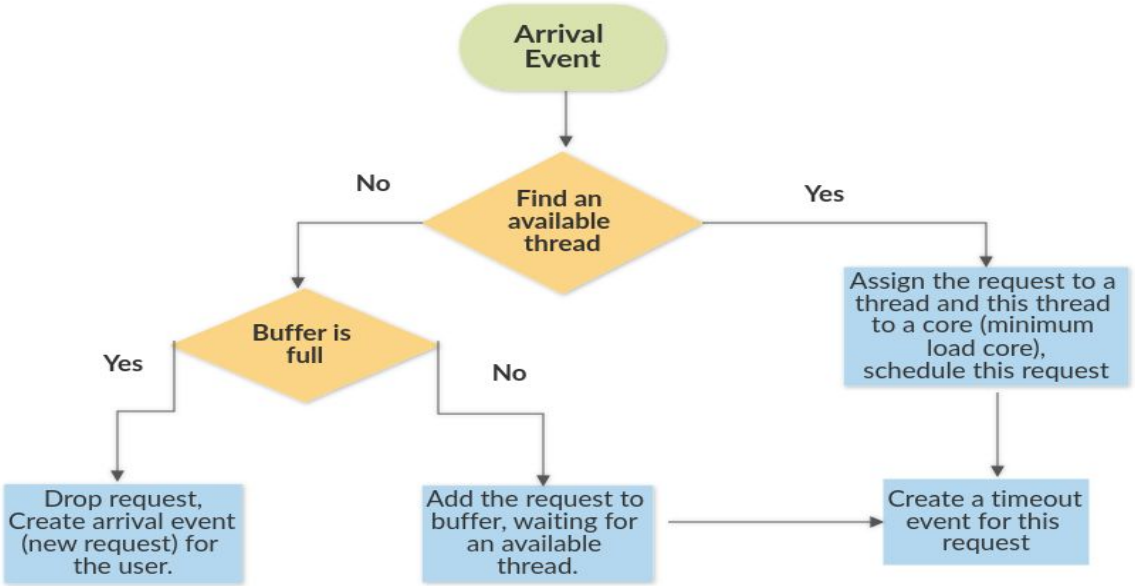
Web application Model



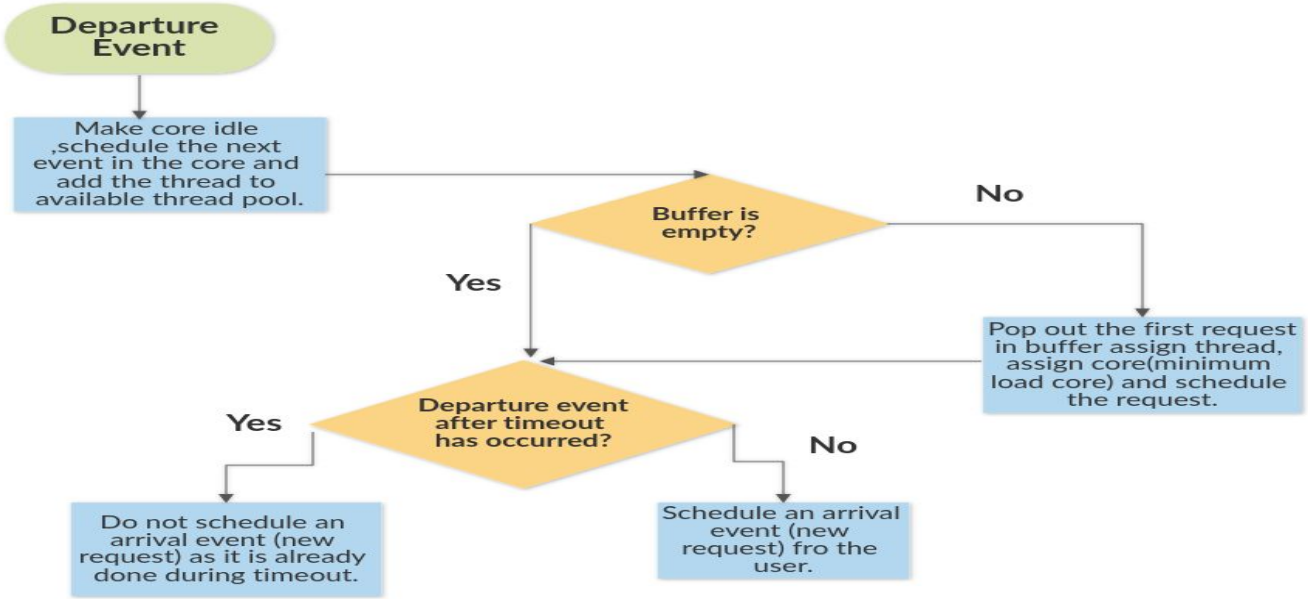
Classes

- **Thread** - Thread is an entity that handles each request.
- **Threadpool -** Threadpool represents all the threads present in the server. Threadpool has functions such as getFreeThread, addThread.
- **Buffer -** Buffer represents the queue, which can be used for request queue or even runqueue of core.
- **Core -** Core class has a buffer data member which has all the requests that are assigned to the core.
- **Server -** This class represents the server system of web application. It has various functions for calculating the server side metrics.
- **Event -** Event class represents various event types. The types of events are arrival event, departure event, context switch event, time slice expire event and time out event.
- **Eventlist -** It a wrap around class for a min heap for getting the next event.
- **Various Probability distributions -** Various probability distribution classes are constant distribution, uniform probability distribution and exponential probability distribution.

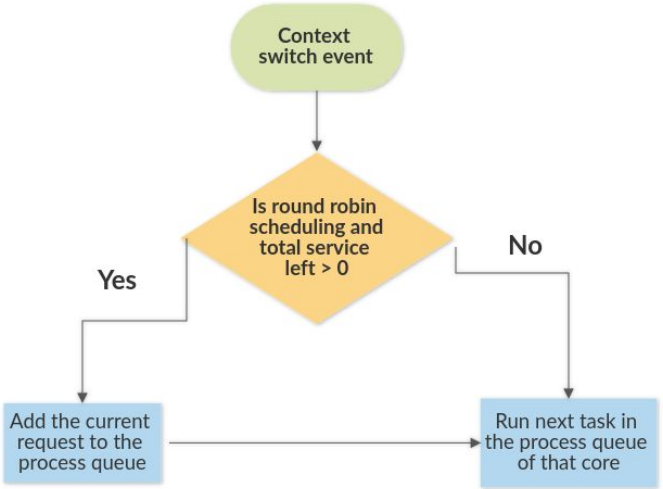
Arrival Event Handler Flow chart

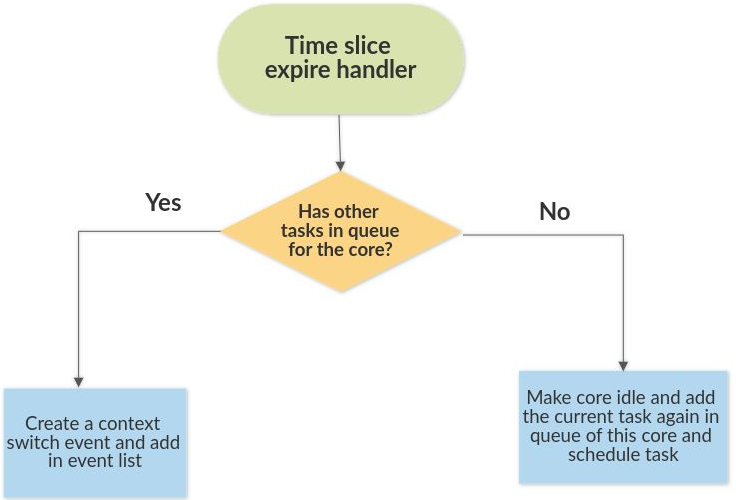


Departure event handler flow chart

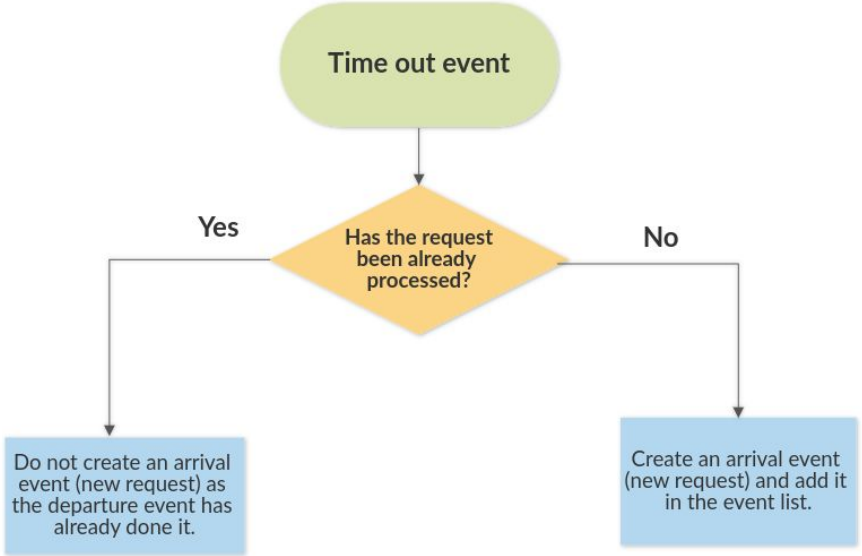


Context Switch event handler and Timesplice expire event handler

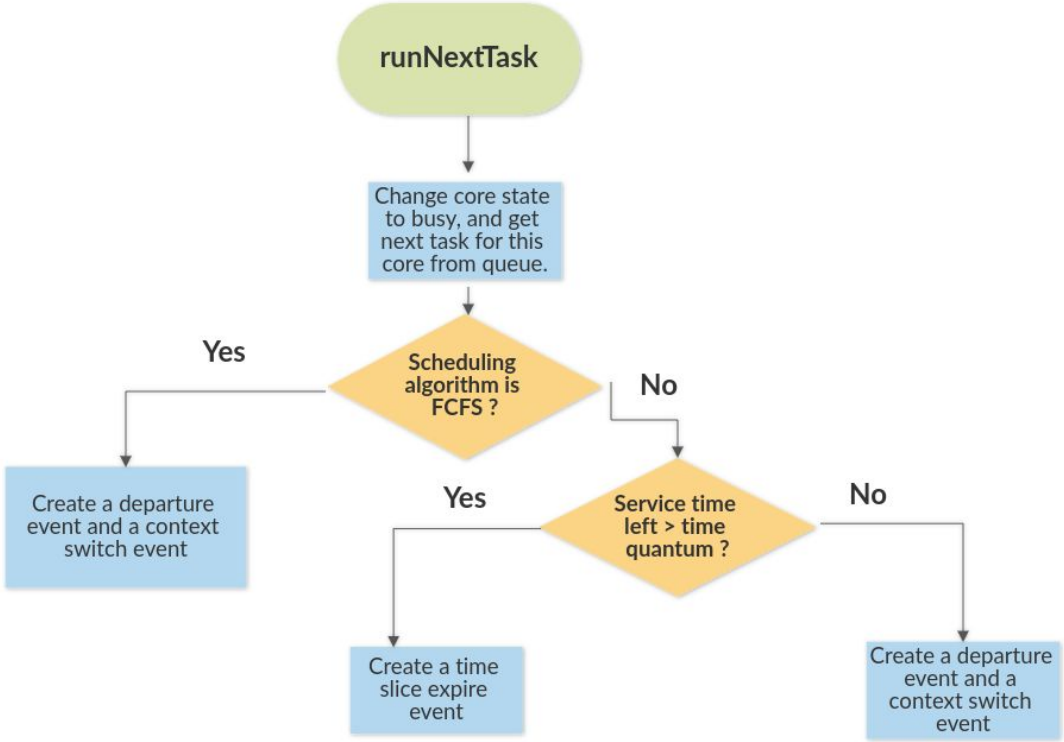




Timeout event handler flow chart



runNextTask flowchart



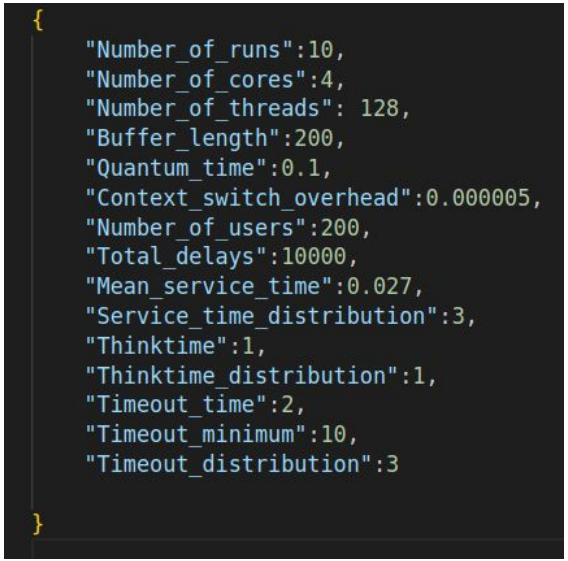
Performance Metrics

Configuration of the web server:

- All times are in seconds.
- Distribution types:
  - 1 - Constant
  - 2 - Uniform distribution
  - 3 - Exponential distribution
- Quantum time of the machine is obtained by using

  `/proc/sys/kernel/sched_rr_timeslice_ms`

  command



Calculations

- τ = 0.027 seconds (Mean service time - exponential distribution)
- c = 4 (number of cores)
- Maximum System throughput :

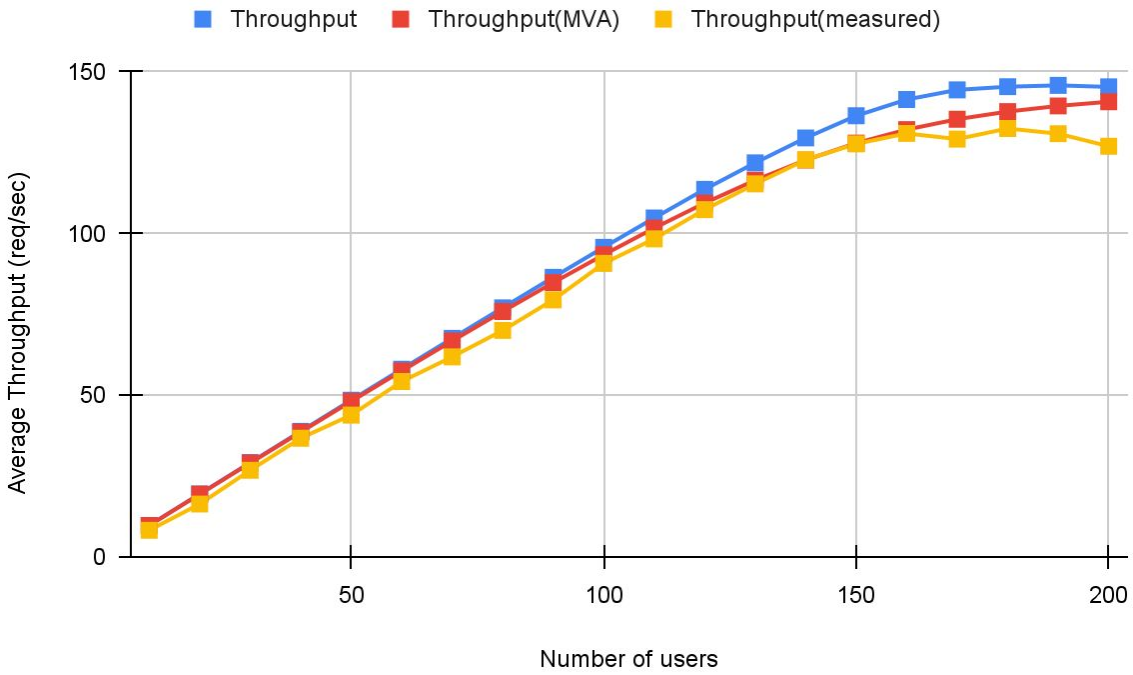
  $\Lambda sys = c * \mu = 4 * (1 / 0.027) = 149$ reqs/sec

- Saturation number of users:

  $M^* = c * (1 + thinktime/service\ time) = 4 * (1+ 1/0.027) = 153$

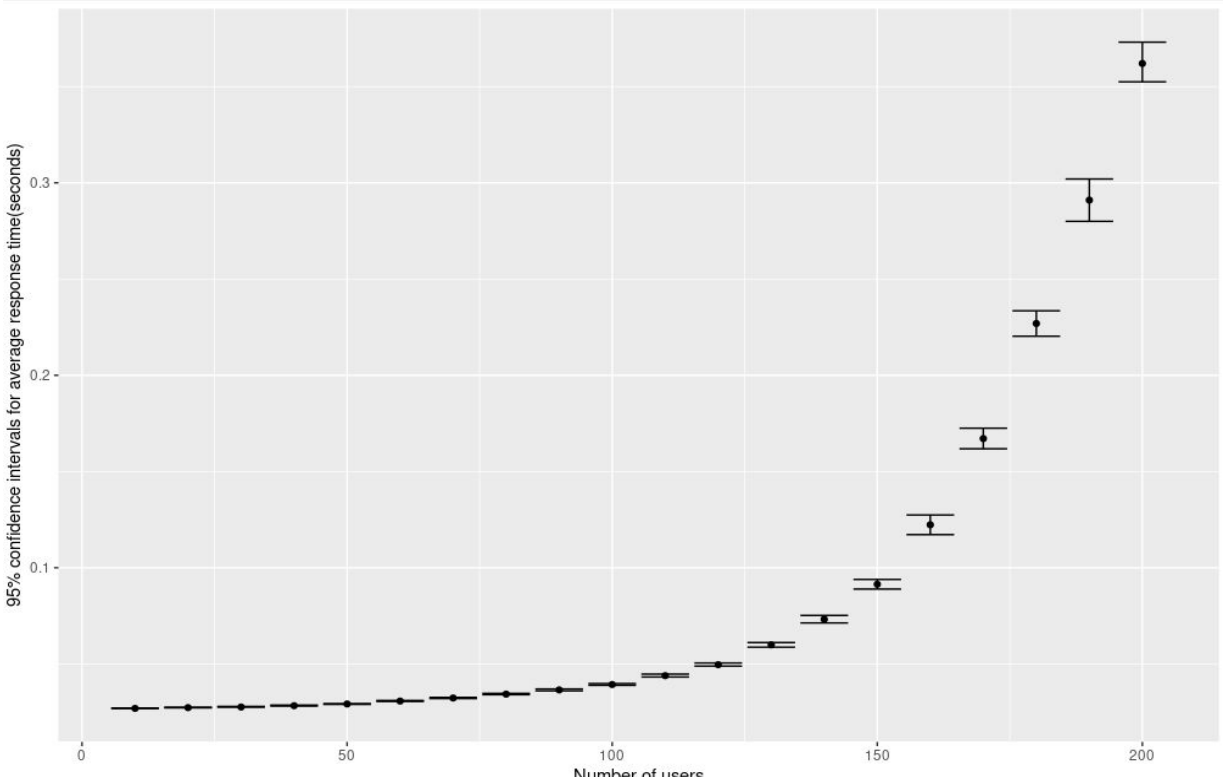
- Maximum number in system = 200(request buffer length) + 128 (total threads) = 328
- Maximum average response time = (328/4 + 1) * τ = 2.241 seconds.

Average Throughput and Average throughput (measured) (reqs/sec) vs number of users



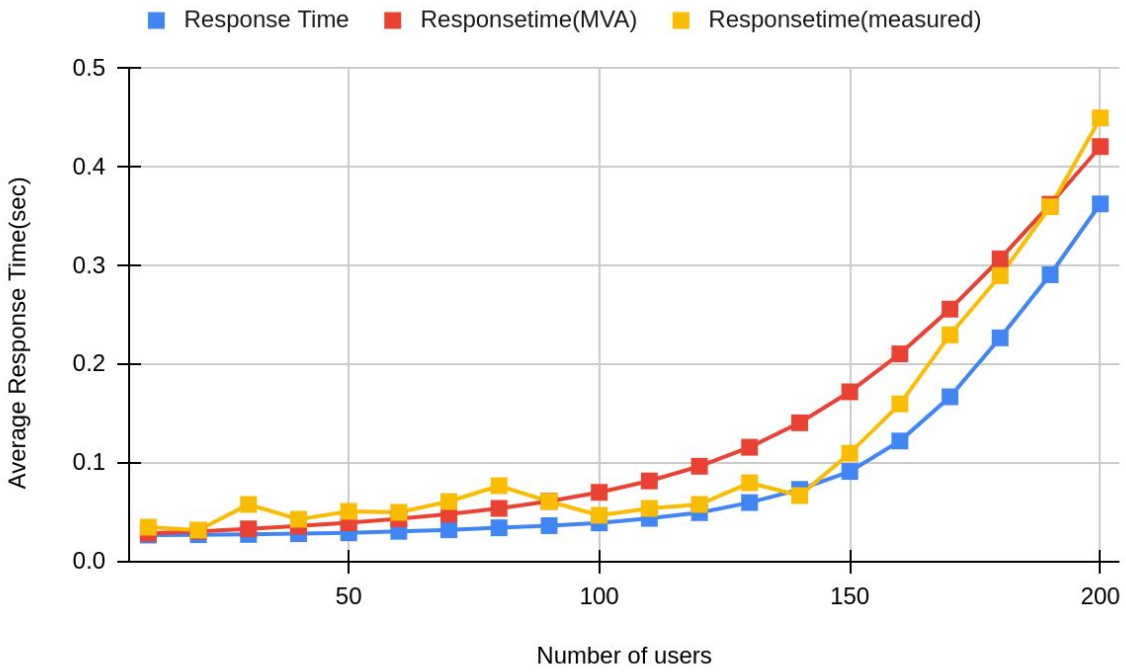
In all the graphs the throughput stabilizes after 160 number of users.

Average response time(seconds) vs Number of users



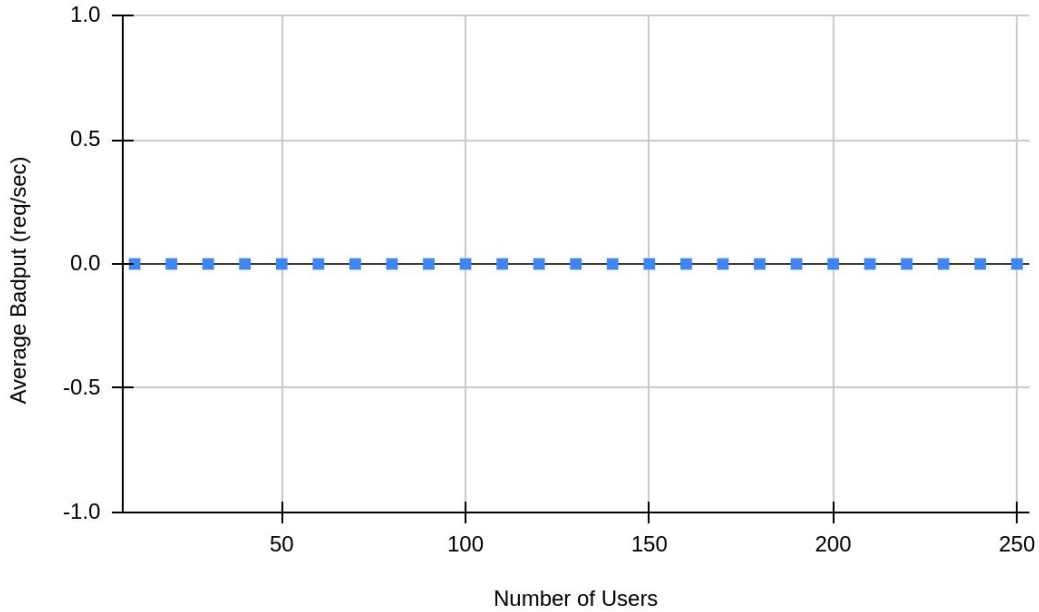
The average response time increases linearly after 160 number of users.

Average Response time(sec) vs Number of users



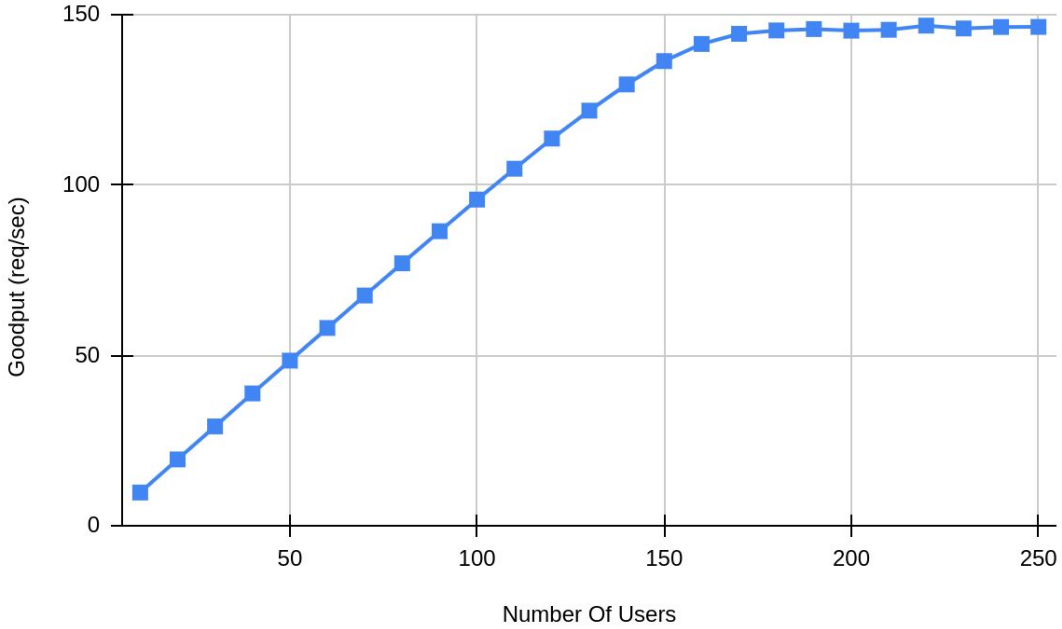
Average response time is increasing linearly after 150 number of users and can be observed in all graphs.

Average bad put rate(reqs/sec) vs number of users



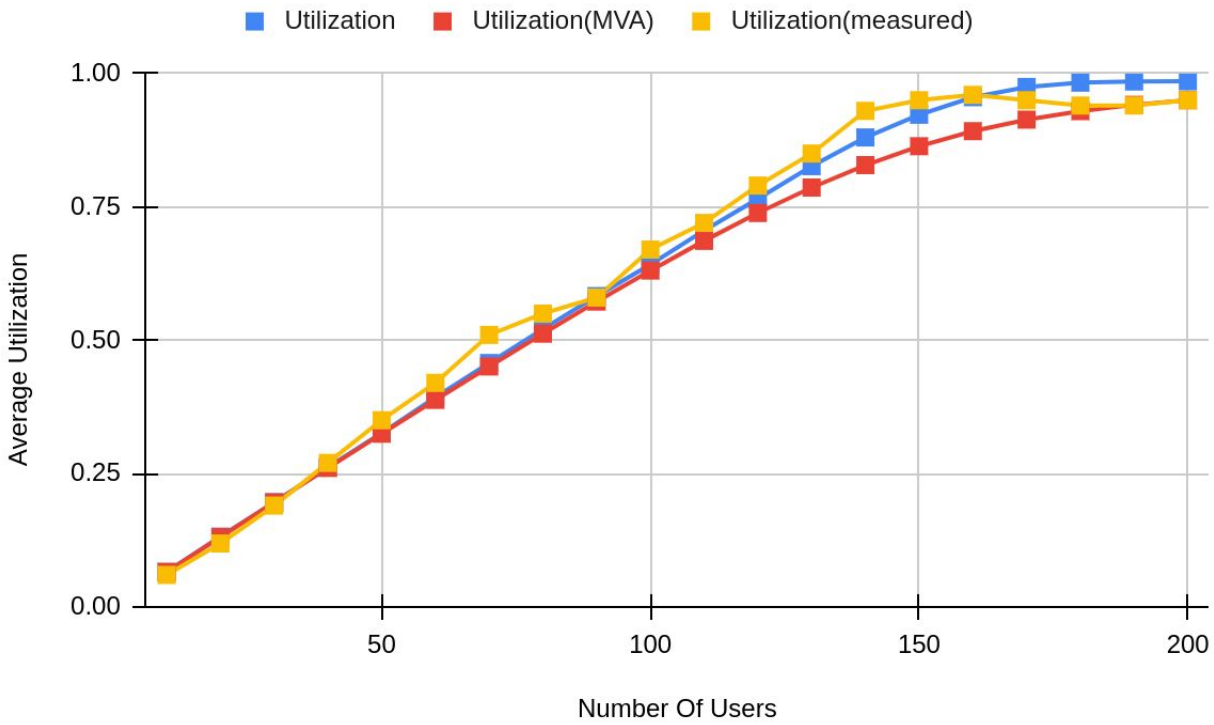
In this graph the bad put rate for all number of users is 0 reqs/sec as every request is processed before its timeout.

Average goodput rate(reqs/sec) vs number of users



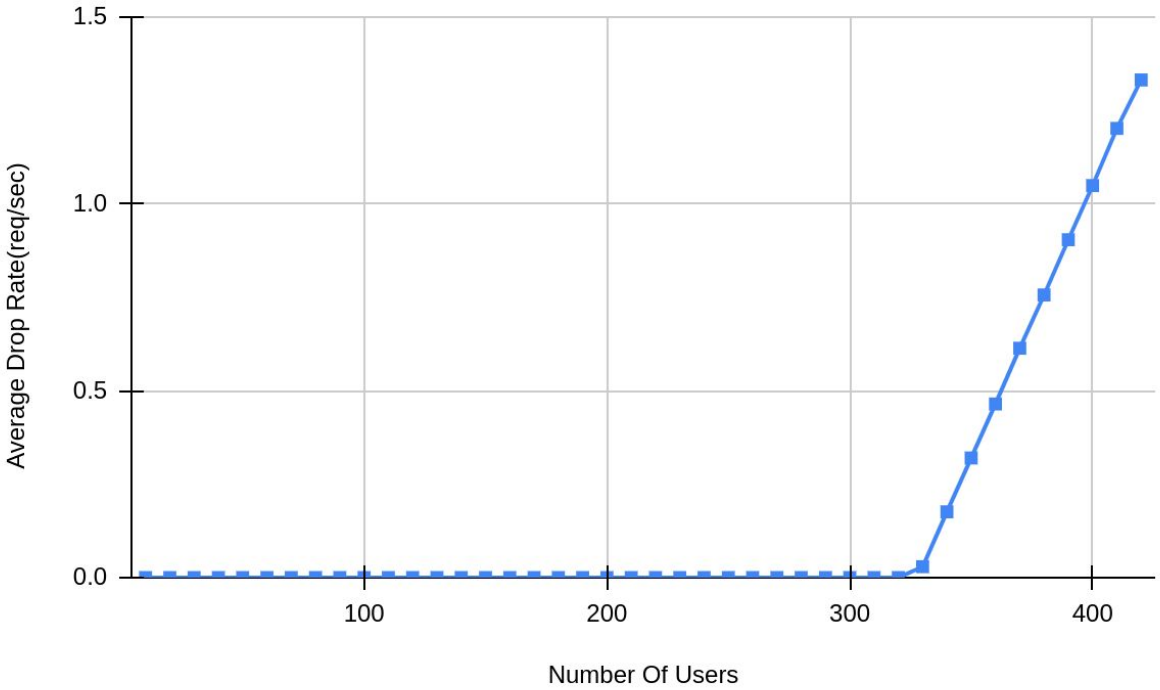
The average goodput rate is equal to the average throughput of the web application, and it stabilizes after 160 number of users and the maximum average goodput is 146 reqs/sec.

Average core utilization vs number of users



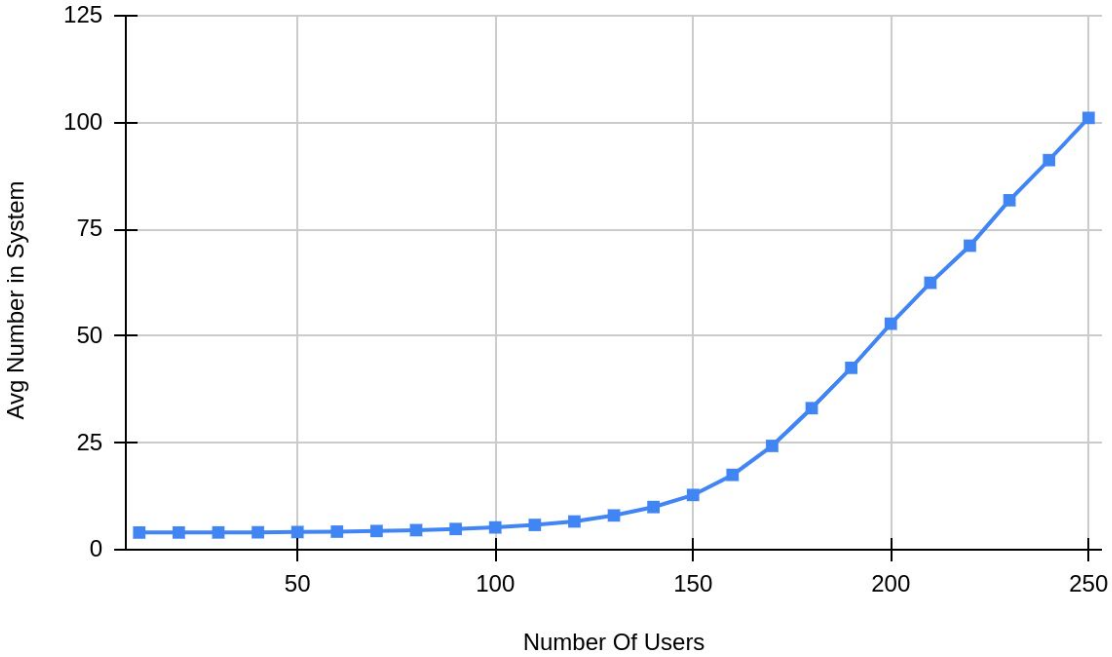
The average core utilization of the web server is stabilized after 160 number of users and maximum core utilization is 0.985.

Average request drop rate(req/sec) vs number of users



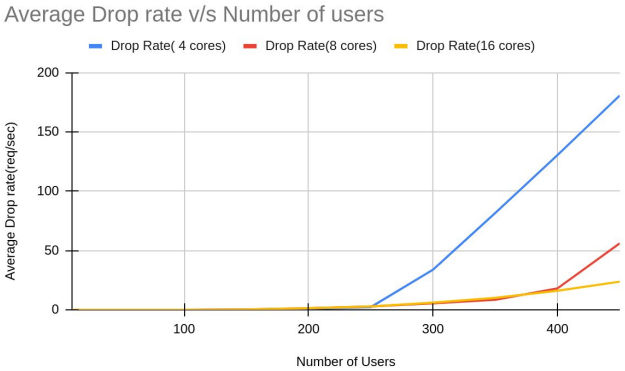
The average request drop rate(reqs/sec) is 0 initially upto 330 number of users. It increases drastically after 330 as the maximum buffer length is 200. And maximum number in system is 328.

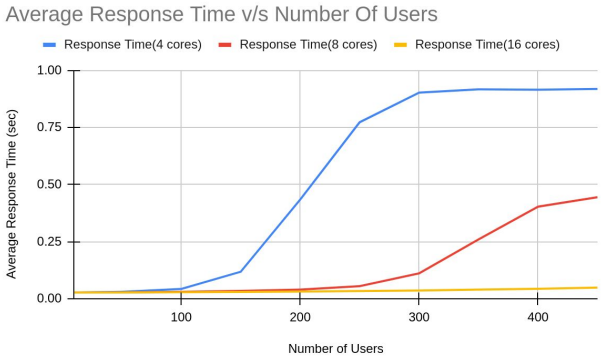
Average number in system vs number of users

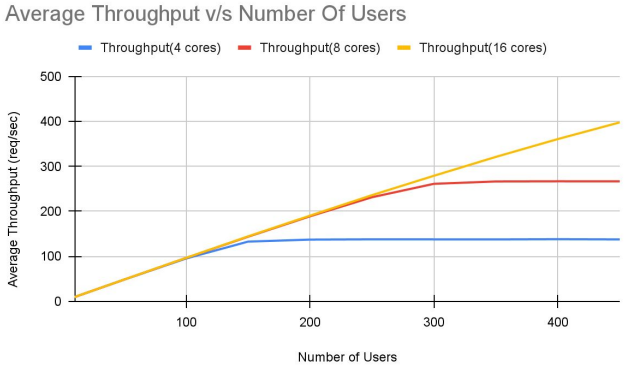


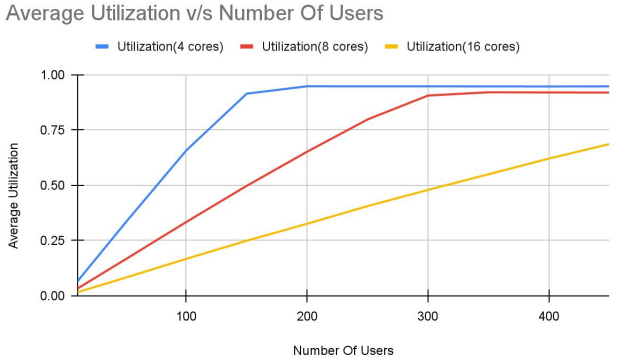
The average number in system increases drastically after 160 number of users and stabilizes.

Observations made during changing cores

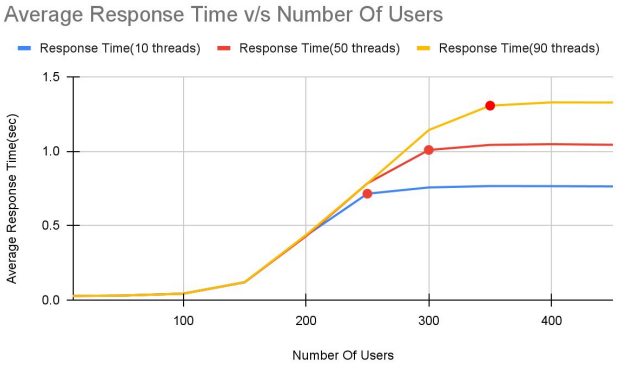


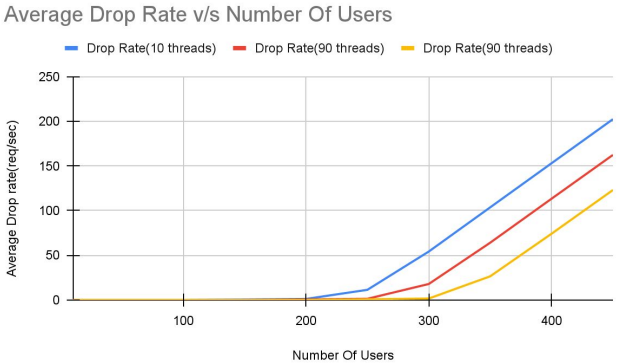


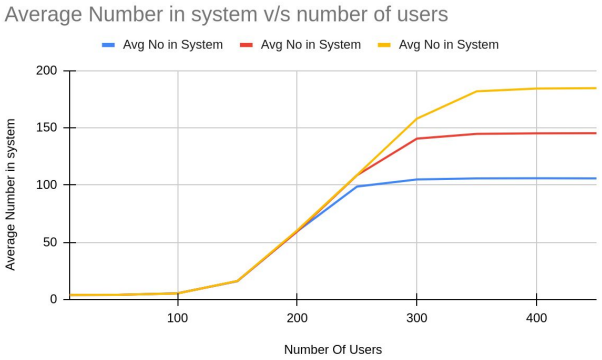


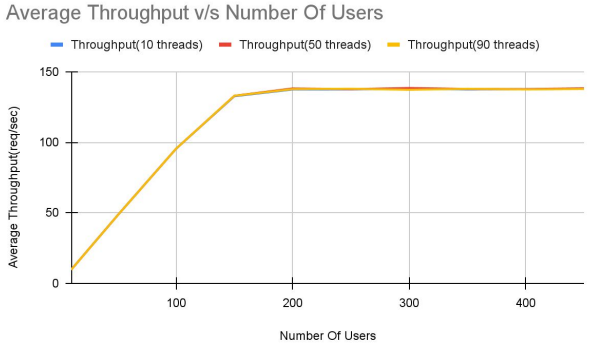


Observations made by changing number of threads



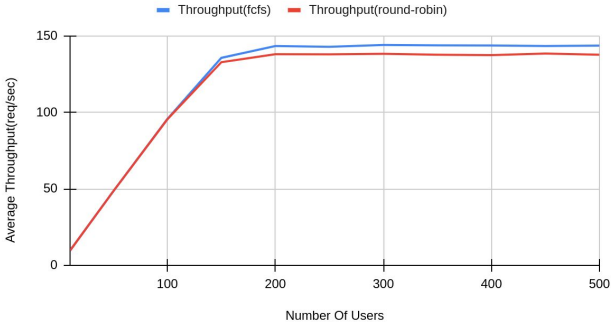




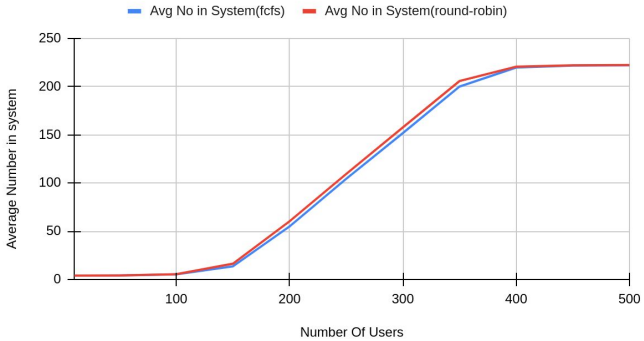


FCFS vs Round robin scheduling

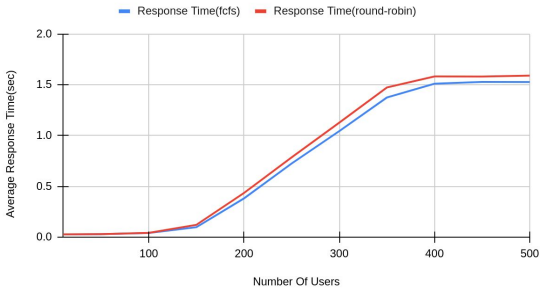
Average Throughput v/s Number of Users



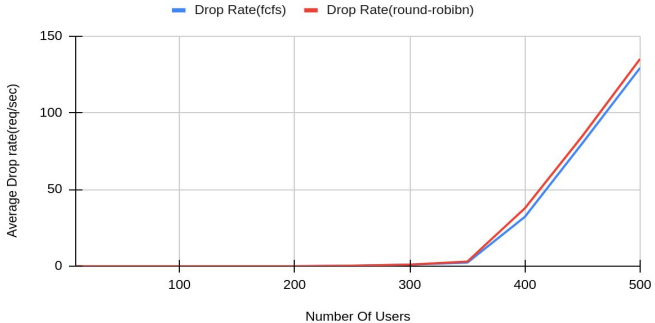
Average Number in System v/s Number Of Users



Average Response Time v/s Number of Users

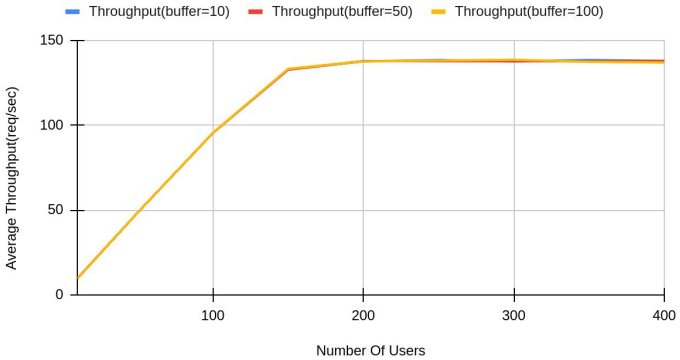


Average Drop rate v/s Number Of Users

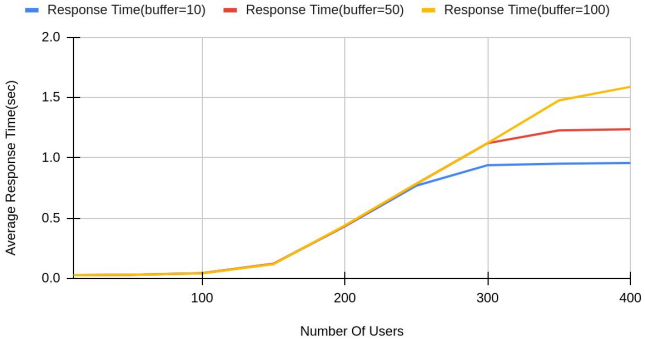


Observations by changing buffer length

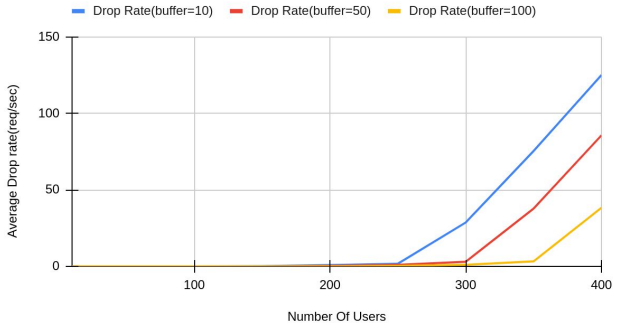
Average Throughput v/s Number Of Users



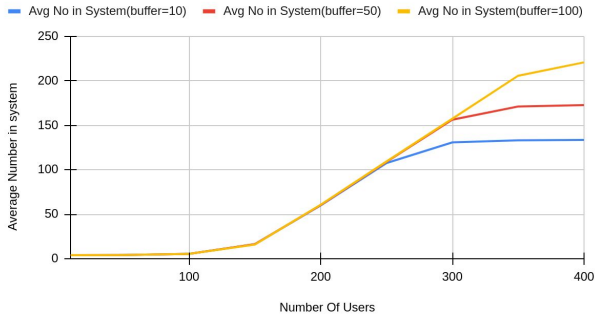
Average response time v/s number of users



Average Drop Rate v/s Number of Users



Average Number in system v/s Number Of Users



Observations by changing context switch overhead