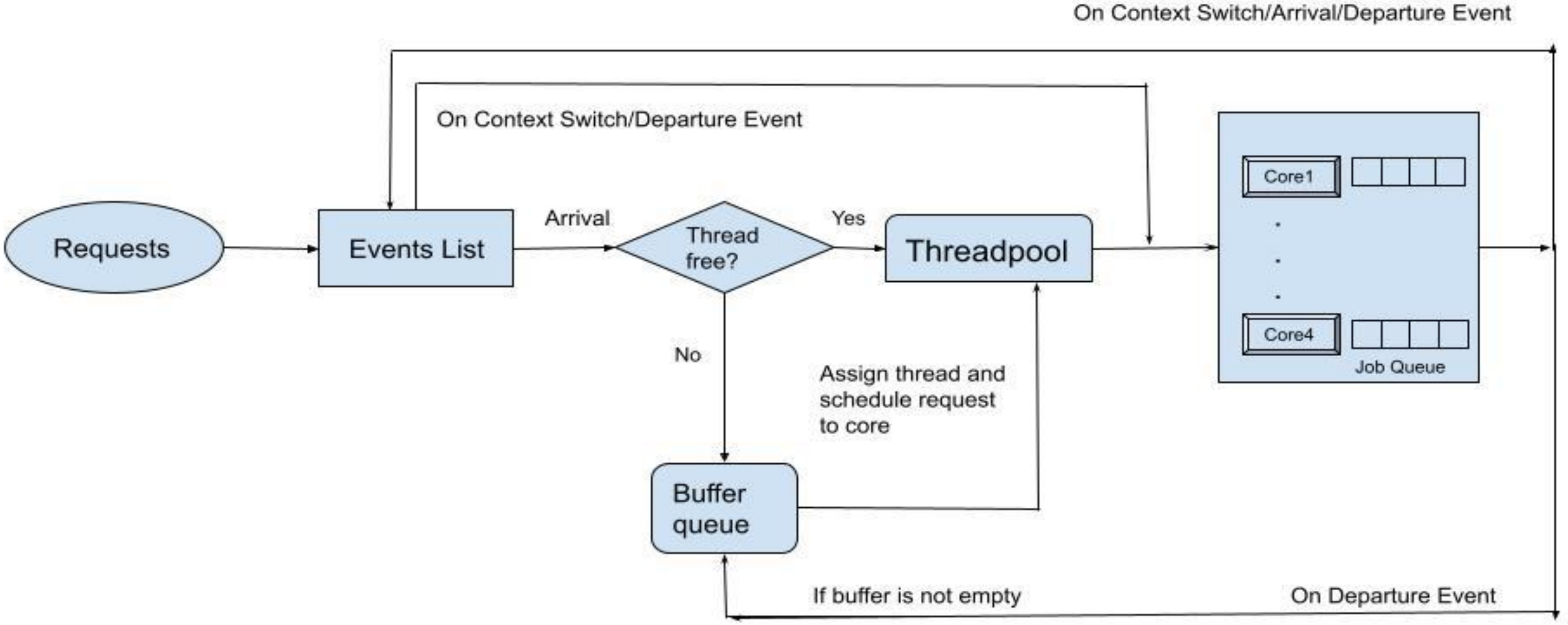
CS-681
Assignment 2

Simulation Analysis of a Web Application

Shivaji Chirumamilla (203050056)
Mohan Rajasekhar Ajjampudi (213050060)

Design of Simulation program



System Assumption

- Initially all requests are mapped with events with arrival time 0 and pushed into eventList.
- From eventList if thread is available in threadpool, request is mapped to thread and is inserted into job queue of core.
- If free thread is not available then buffer is checked to see if space is available or not.
- If buffer space is available then event is pushed into buffer else it is counted as drops and next request is scheduled with current time + think time as arrival time.
- Once thread is scheduled on core, events like departure/context_switch are generated based on the remaining service time.
- On context_switch current job will popped & pushed back of job queue and next job in job queue will be scheduled for its events.
- On Departure, thread is made free and buffer will be checked to assign thread if any request is waiting for core.

Classes and Events

- **Request:** It stores all the details related to request like service time, timeout time etc
- **Event:** Request object is mapped to event object followed by setting of time field which is used priority field of heap.
- **EventList:** It is a Priority Queue Data structure, to store the event objects.
- **Request Buffer:** It is a queue of fixed size, which is used to store request objects if a thread is not available.

Classes and Events

- **Thread**: represents the process for request in core. A request can execute

- **Threadpool**: this object is used to keep track of threads status.

- **Core**: simulates the cpu in web server system. It uses the job queue to keep track of current set of request waiting for core usage.

- **Job queue**: It is queue data structure. Maintains the jobs waiting for core in ordered way.

Events Considered:

1) Arrival
2) Quantum_Done
3) Departure

Simulation models and Configured parameters

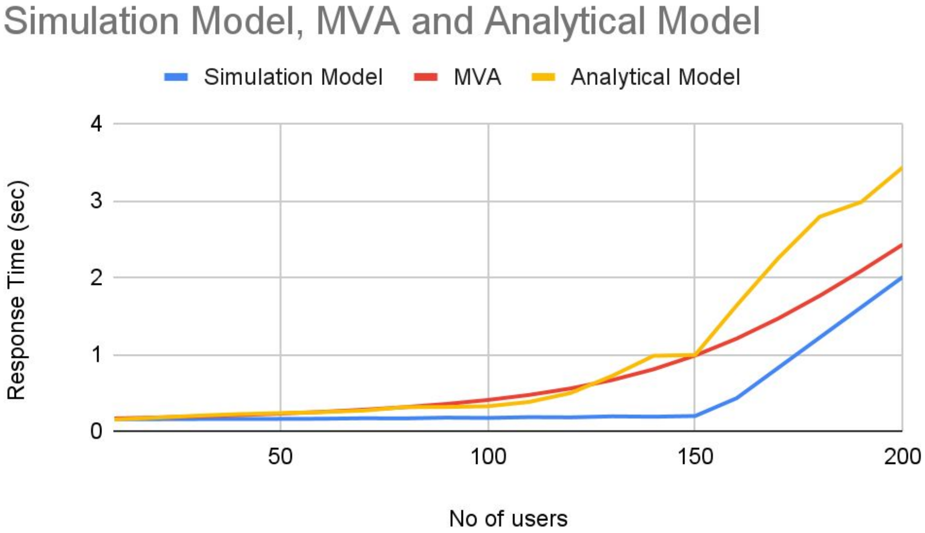
- Number of cores: 4
- Mean Service Time: 160ms
- Mean Think time : 6000 ms
- Theoretical Saturation Limit: 154
  - c(1+(mu)/lambda) = 4+(4*6000/160) = 154
- Analytical Model Saturation Limit: 130-140
- Simulation Model Saturation Limit: 150

Model 1
Scheduling algo : FCFS

Model 2
Scheduling algo : Round Robin

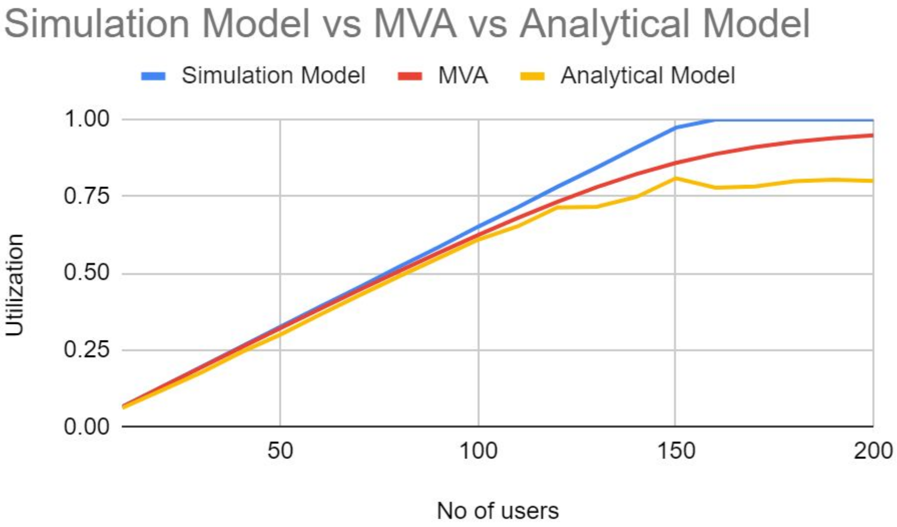
FCFS:Response time graph

- Response time increases gradually and had a sudden spike after certain point.
- Sudden spikes can be observed when no of users reaches 150.
- Slope(saturation) gives (tau/c)
- Average slope of asymptote = 39.4
- Tau = c*(slope of line)
  = 4 * 39.4 = 157.6 ms.
- Service-Time we configured ~ 160ms.



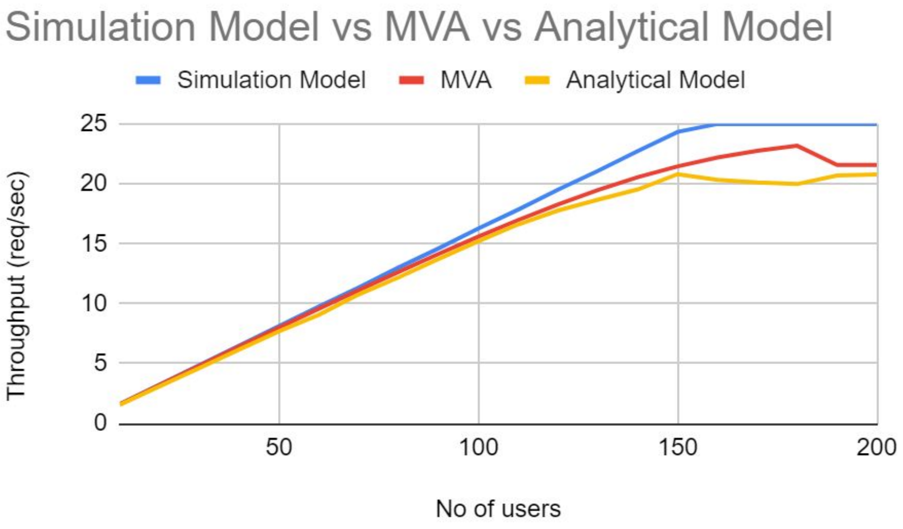
FCFS:Utilization graph

- Trivial behaviour is observed in all models which is increase in utilization as number of users increases and then flattens after saturation.
- Each model is saturating after crossing kleinrock maximum number but converging to different values.
- At m = 150 , rho=0.9732 ,
  tpt = 24.33,
  #cores = 4
  Tau = (150*0.9732)/24.33
  = 160ms.



FCFS:Throughput graph

- Usual behaviour, which is increase in throughput as number of users increases, is observed from all graphs.
- Simulation model giving a slightly increased throughput when compared to other two models.

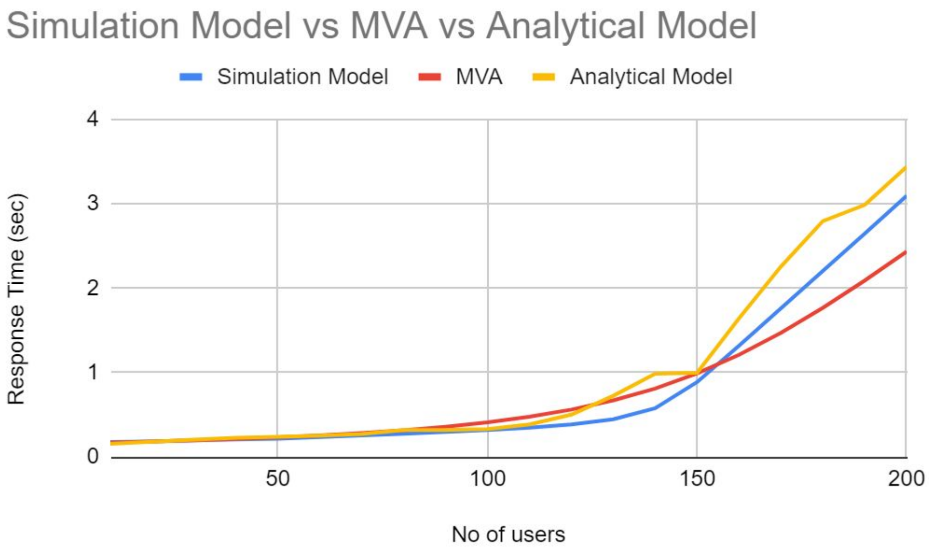


Round Robin Model

- Number of cores: 4
- Mean Service Time: 160ms
- Time Quantum: 90ms
- Context Switch Time: 10ms
- Mean Think time : 6000 ms

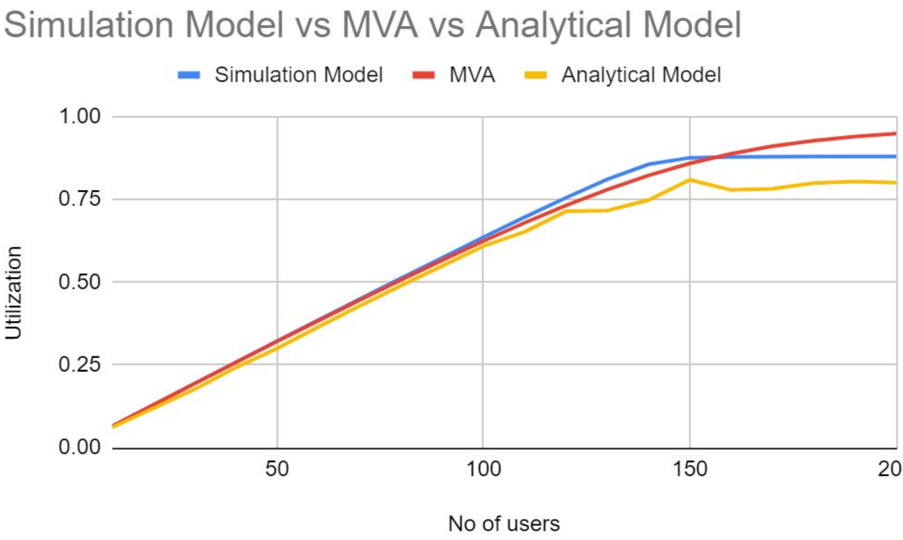
Round Robin: Response time graph

- Response time increases gradually and had a sudden spike after certain point.
- All of the 3 graph behaves in the same manner till no of users reaches to 130.
- Post that, MVA and Simulation converges to same pattern but experimental model diverges a bit from other two.
- Slope(saturation) gives (tau/c)
- Average slope of asymptote =
  Tau = c*(slope of line)
  = 4 * 44.1 = 176 ms
- Service-Time configured ~160ms.



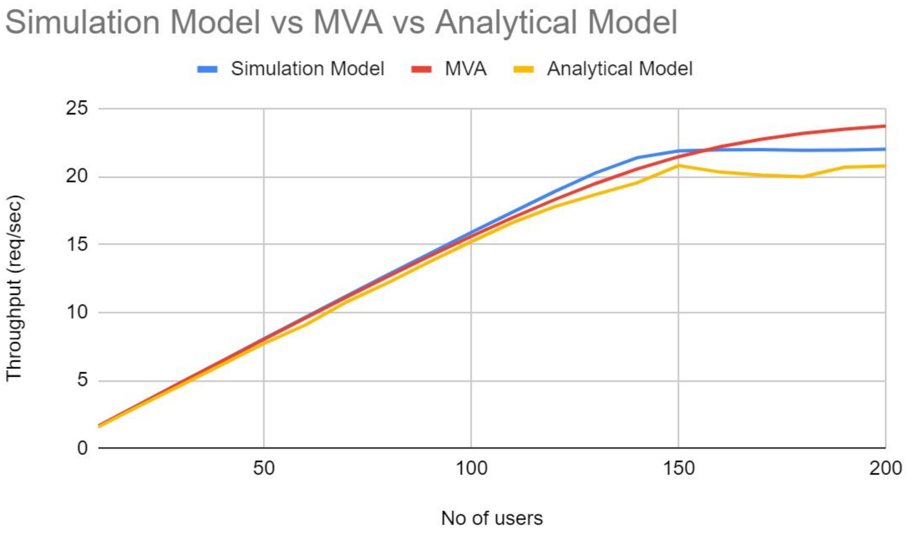
Round Robin: Utilization graph

- Trivial behaviour is observed in all models which is increase in utilization as number of users increases and then flattens after saturation.
- Each model is saturating after crossing kleinrock maximum number but converging to different values.
- At m = 140 , rho=0.85578085 ,
  tpt = 21.38,
  #cores = 4
  Tau =
  (140*0.8557)/21.38=160ms.



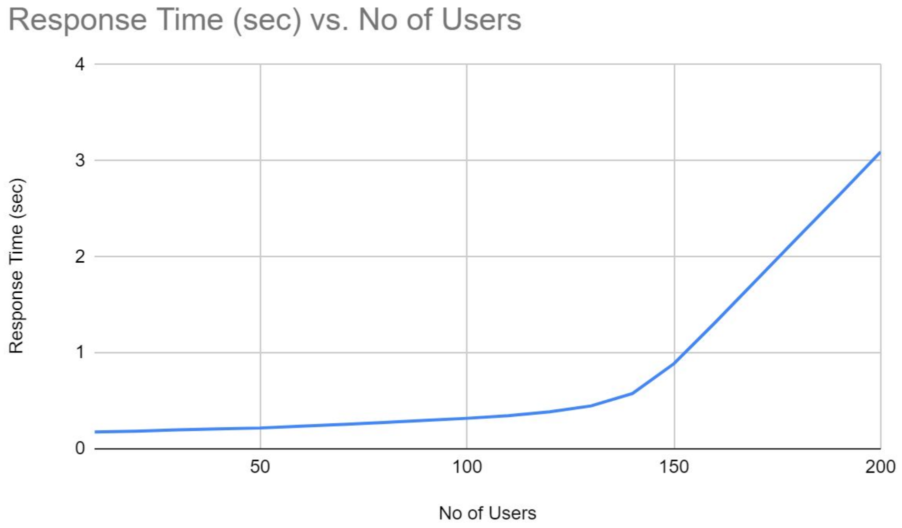
Round Robin:Throughput graph

- Usual behaviour, which is increase in throughput as number of users increases, is observed from all graphs.
- Simulation model giving slightly increased throughput when compared to other two models.



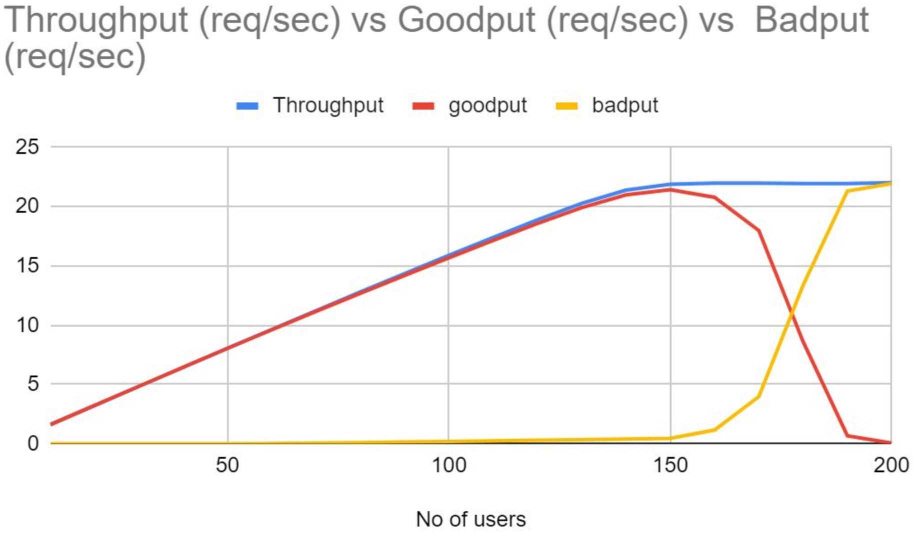
Response Times with 95% Confidence Intervals

No of Users	Mean	Minimum	Maximum
10	0.1758684	0.1754287438	0.1763080562
20	0.1836153	0.1831062796	0.1841243204
30	0.19758775	0.1968527618	0.1983227382
40	0.20785375	0.2068621985	0.2088453015
50	0.2163356	0.2154594852	0.2172117148
60	0.23638095	0.2352359574	0.2375259426
70	0.2533571	0.2519261772	0.2547880228
80	0.27309225	0.2713636021	0.2748208979
90	0.29709255	0.2950275296	0.2991575704
100	0.31732345	0.315780747	0.318866153
110	0.3452297	0.3430268598	0.3474325402
120	0.38569355	0.3838149785	0.3875721215
130	0.4465985	0.444358365	0.448838635
140	0.5763728	0.5731989415	0.5795466585
150	0.8859319	0.8828124078	0.8890513922
160	1.316438	1.312642536	1.320233464
170	1.7603465	1.756932952	1.763760048
180	2.2035615	2.200262106	2.206860894
190	2.6451205	2.640617152	2.649623848
200	3.0929695	3.088704303	3.097234697



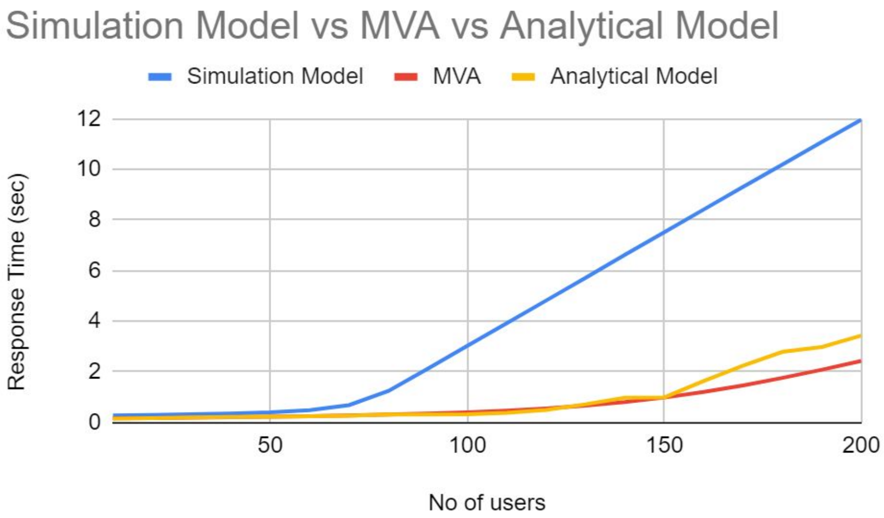
Goodput vs Badput

- As the number of users increases, average goodput decreases after crossing saturation and reverse behaviour is observed in badput graph.

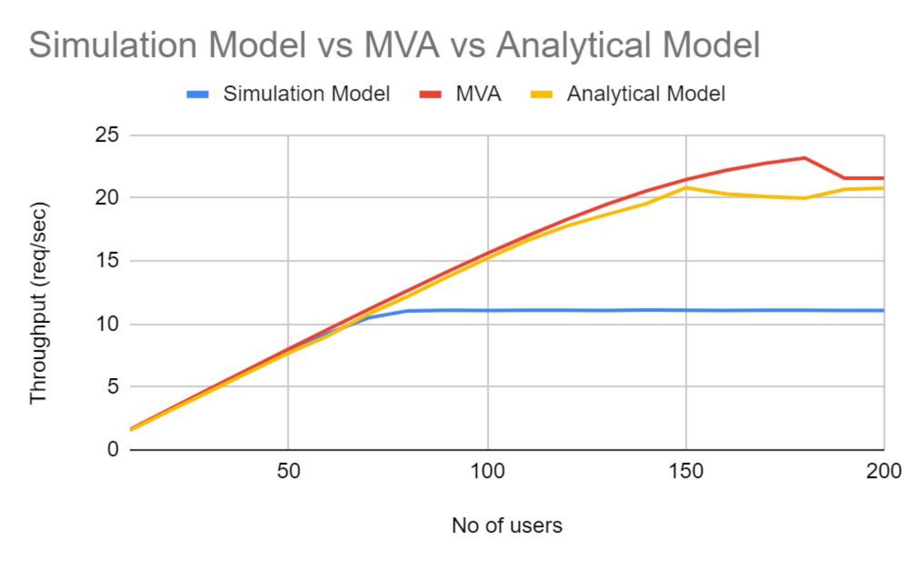


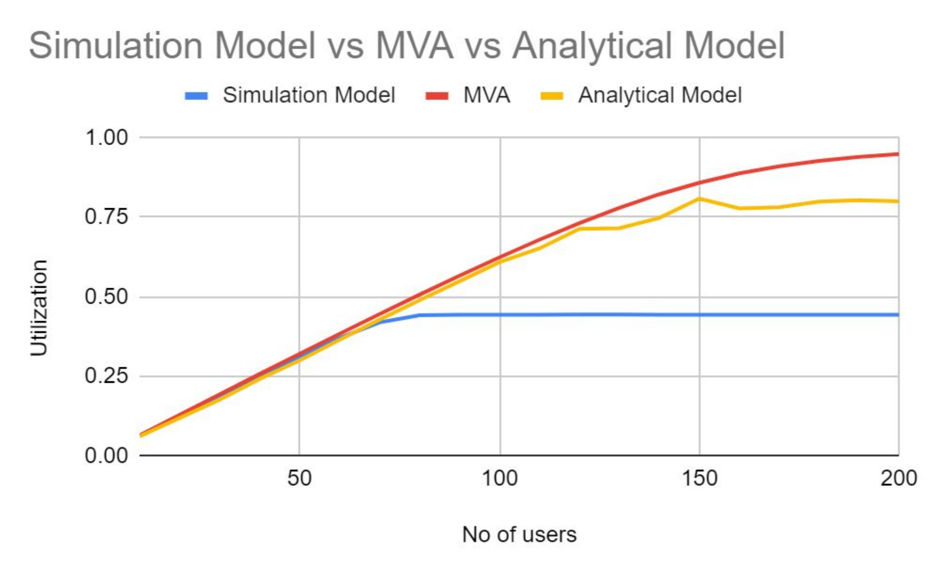
Round Robin 2: Response time graph

- Time Quantum: 80ms
- Context Switch Time: 80ms
- Since, the quantum time and context switch time are same, cpu will spend half the time for context switching.



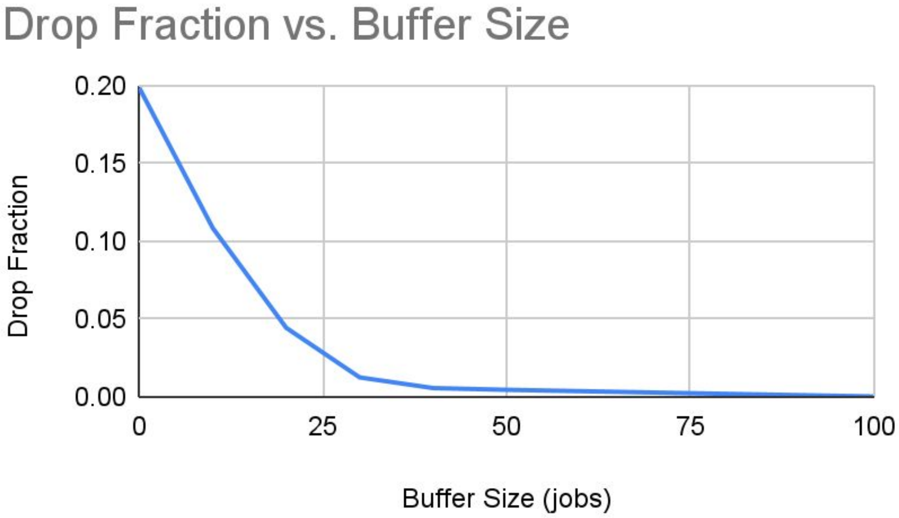
Round Robin 2: Throughput and Utilization vs Users





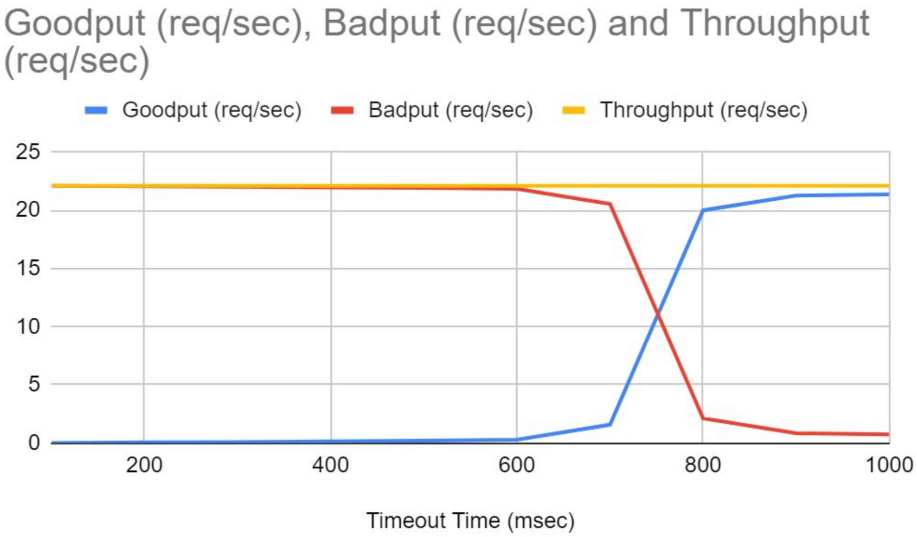
Drop rate vs Buffer Size

- Varied buffers size to observe the drops fraction.
- #users = 200 (fixed).
- #threads = 100(fixed).
- Buffer size is varied from 0 to 100. At 0 buffer size drops are maximum and at 100 buffer size all requests are served without drops.



Goodput vs Badput vs Timeout Time

- Varied timeout time to observe the throughput variation.
- #users = 150(fixed)



Links of experiment results

- https://docs.google.com/spreadsheets/d/1UJXkr7lsE__FYLdyybeLkTT0YF55KgaXrFuOkyOsBK4/edit?usp=sharing