July 19, 2024

Tandem Queue Simulation

Goal

Simulate a series of N M/M/1 queues (called a tandem queue) where the input to the first queue is Poisson of rate λ and each server serves at rate μ (for simplicity). The output of queue i becomes the input of queue $(i+1)$. Choose some manageable N such as $N = 10$ etc. and more if simulation time allows it. Reuse your previous code and use indexes to manage the logic at various queues. Remember that an event will occur at only one queue at a time and the Calendar helps with this.

Experiments

Parameters

- Number of queues N: 10 (for most part, it is varied to plot time simulation run time)
- Arrival rate λ: Varied from 0.08 to 0.96 in steps of 0.08, 12 values such that $0 < \rho < 1$
- Service rate μ: Fixed at 1 for all the queues
- Number of simulations: 30 per data point.
- Total number of customers: 3000 per simulation.
- Confidence interval: 95% using z=1.96.
- Random number generator: Python's built-in random module with seed 1234
- Performance measures: Mean delay and mean number of customers in the system.

Experiment Setup

- Using one code files for M/M/1 Tandem Queue function and related plotting function in Python
- For each value of $\rho = \lambda/\mu$, perform 30 simulations to obtain reliable estimates.
- In each simulation, initialize the queueing system and simulate until 3000 customers have been served.

- Send the customer from currently served queue to the next queue, after service
- Record the data starting from 801th customer to calculate the mean delay and mean number of customers in the system for each simulation.
- Calculate the mean and confidence intervals for each ρ value based on the 30 simulations.
- Study simulation run-time for number of queues varied from 1 to 19
- Queue 5 is chosen as bottleneck queue and varied its service rate to 0.8 and 0.9

Results

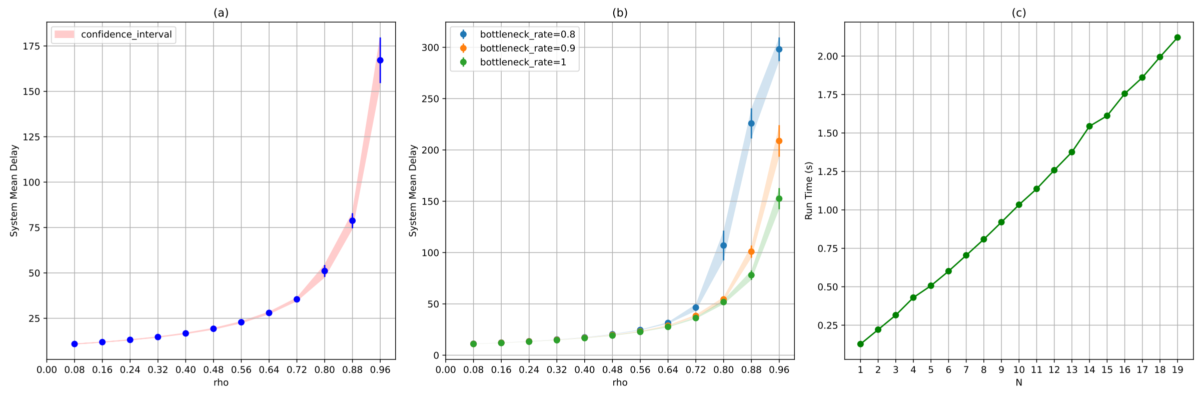

Figure 1: (a) The average time a customer spends in the system versus ρ, (b) Comparing the effect of bottleneck queue on average time customer spend in the system versus ρ, (c) Simulation-run time versus number of queues(N)

I focused on three aspects: the mean system delay as a function of traffic intensity (ρ), the impact of different bottleneck rates on system mean delay, and the computational run time for varying numbers of queues (N).

- Figure 1.(a) depicts the relationship between traffic intensity (ρ) and the system mean delay, illustrating that as ρ approaches 1, the mean delay rapidly increases, which is indicative of a system approaching capacity. The confidence interval broadens significantly for higher values of ρ, suggesting greater variability in system performance under heavy traffic conditions.
- Figure 1.(b) examines the effects of varying the bottleneck rates (0.8, 0.9, and 1.0). The trends demonstrate that a lower bottleneck rate worsens the system mean delay, especially as traffic intensity nears the system's maximum capacity. This finding is evident of bottleneck can really slow down the whole system, even if it's just one part.
- Lastly, Figure 1.(c) shows a linear increase in run time with the number of queues(N). This indicates that the computational complexity of the simulation scales linearly with N, which is consistent with the expectation that each additional queue introduces a proportional amount of additional processing.

Conclusion

The results of the simulations clarifies several key points about the behavior of tandem queue systems. First, the system mean delay is highly sensitive to traffic intensity, particularly as it approaches the system's capacity. Secondly, the presence of a bottleneck can significantly degrade system performance, and its effect is more pronounced under higher traffic intensity levels. Lastly, the simulation's computational cost increases linearly with the number of queues, which suggests that the simulation approach used is scalable.

These findings have implications for the design and management of queuing systems in various real-world contexts. Strategies to mitigate the impact of traffic intensity and to manage or eliminate bottlenecks are crucial for maintaining system efficiency.

Comparing Systems with Shared Queues

Goal

To compare the performance of two distinct queueing system designs. Both systems are equipped with two identical processors handling two independent Poisson input streams, with different arrival rates for each stream. I investigate whether two independent queues, each dedicated to a single processor, outperform a single shared queue where customers are attended by the first available processor.

Experiments

Parameters

- Arrival rates λ_1 and λ_2: The incoming jobs arrive with rates $\lambda_1 = \frac{1}{3}$ per minute and $\lambda_2 = \frac{1}{4}$ per minute for processors 1 and 2 respectively.
- Service rate μ: The service rate for both processors is $\mu = \frac{1}{2}$ per minute, equivalent to an average service time of 2 minutes.
- Number of simulations: The simulation is run until the total time reaches 100,000 units to ensure stable averages.

Experiment Setup

- The experiments are set up to simulate two independent queues and a shared queue using discrete-event simulation code based on previous tandem queue simulation.
- For the independent queue scenario, each queue receives its own input stream and has its own dedicated processor. The shared queue scenario amalgamates the input streams and directs customers to the first available processor.
- The primary metric of interest is the average delay experienced by customers in the system. This metric serves as a measure of system efficiency and customer satisfaction.

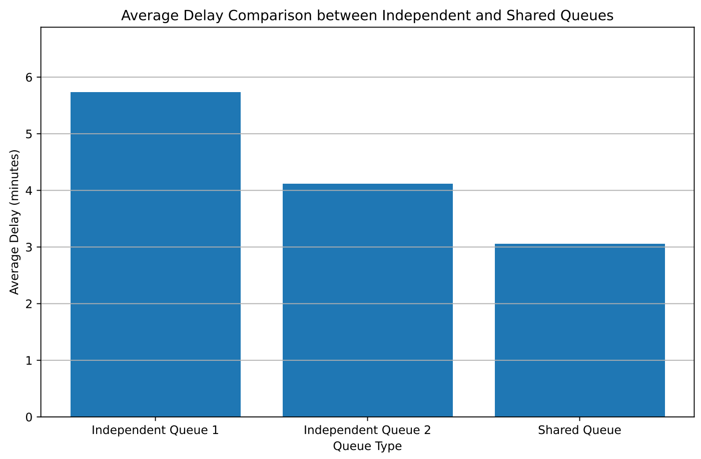

Figure 2: Average delay comparison between independent and shared queues.

Results and Conclusion

- The simulation results are depicted in Figure 2, which illustrates the average delay customers experience in each queueing scenario.
- It is observed that the average delay in the shared queue is less than that of the independent queues. This suggests that pooling resources and allowing flexibility in processor allocation can lead to improved system performance.
- These findings can influence system design decisions, especially in scenarios where service times are variable and arrival rates are different but related, as in the case of the shared queue system.

In conclusion, the shared queue design demonstrates a superior average delay performance compared to the two independent queue systems. This study provides a foundation for further exploration into different queueing strategies and their impacts on system efficiency.