Stochastic Modelling and Simulation of Queues

Kinga Bagyo

Supervisor: Dr Burak Buke

September 2022

Abstract

Every type of queue can be in (often infinitely) many different states, depending on the number of customers being in the system, the individual servers being idle or busy, and the routing method used to allocate the next customer to an idle server. These states are represented by transition diagrams for three different routing methods, from which the steady-state probabilities of being in each state are expressed using Chapman-Kolmogorov equations. Using the steady state probabilities, the theoretical average queue length is calculated and compared with the mean results obtained from a computer simulation of identical queues. Furthermore, a more complex queue involving two types of customer problems is examined in terms of how dependence of mean service rates of type I and type II problems affects average queue length. Results show that average queue length is decreasing with closer dependency of mean service rates in case of all customer allocation techniques.

1 Introduction

The simplest type of queue is the M/M/1 queue, where a single server handles all customers arriving one by one. To model this and all further queues, we assume them to be Poisson processes, i.e. inter-arrival times and service times are exponential random variables with mean rate λ and μ, respectively. This situation could be modelled by a Continuous-Time Markov Chain (CTMC), and the embedded Discrete-Time Markov Chain's (DTMC) state transition diagram is shown in Fig. 1.

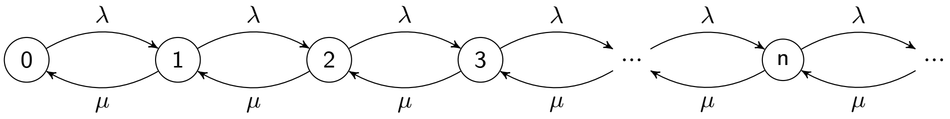
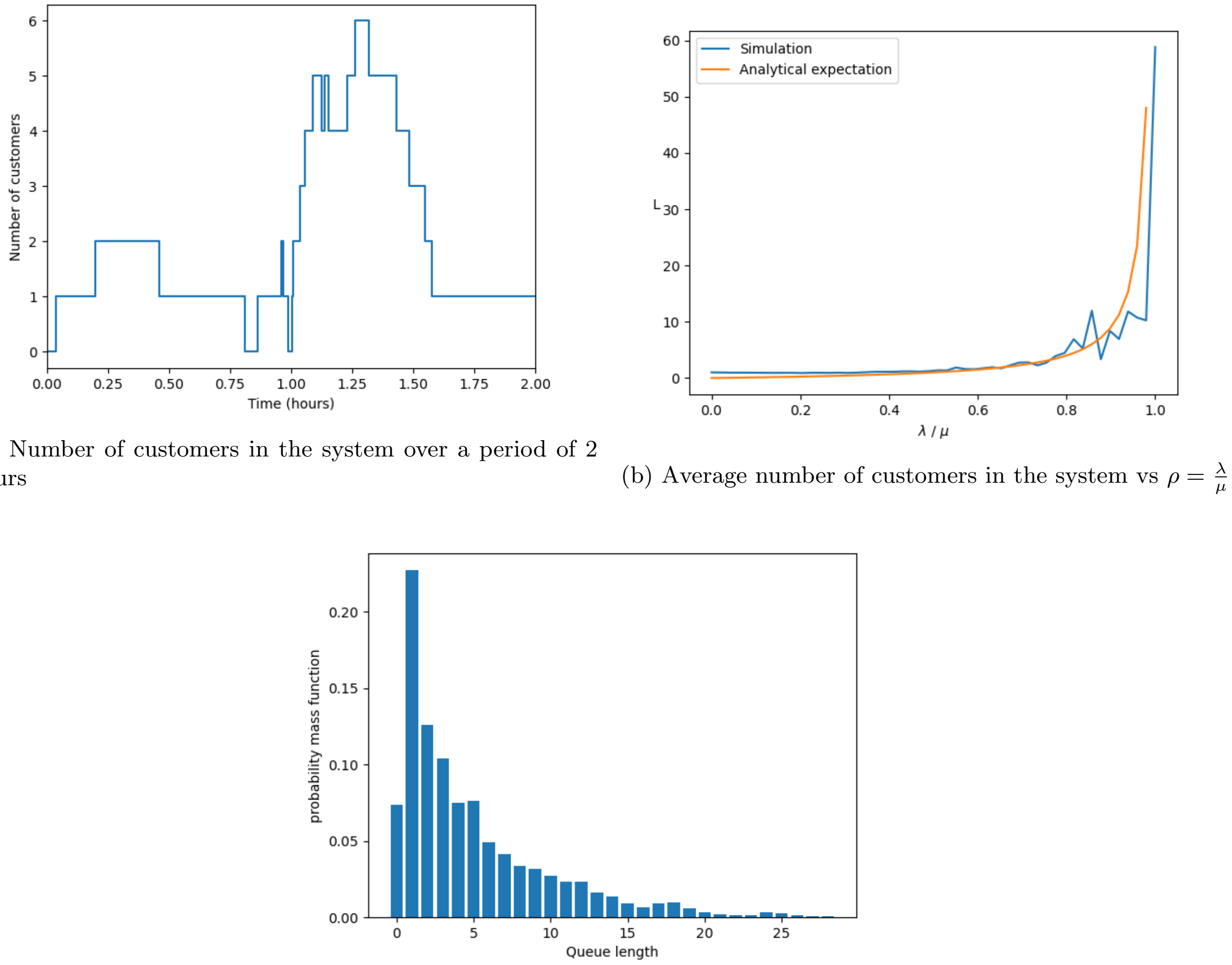


Figure 1: State transition diagram of M/M/1 queue

However, in most real life settings, our model is much more complex. Often we have several servers handling arrivals, and customers usually have a finite patience time, which, upon expiry, makes the customer leave without being served. This event is called 'reneging' or 'abandonment'. Similar to arrival and service rates, abandonment rates are assumed to be exponentially distributed with mean γ. In the sections below, we will also examine the different methods of allocating customers to different servers. For the simulations, some of the most important performance indicators which we will examine are:[1]

- Average queue length (L_{avg}): The average number of customers in the system. Note that average queue length will be used here as referring to both the customers waiting in line and the customers being served by the servers.
- System utilization: The fraction of time when at least one of the servers is working. The individual utilization of a server is the fraction of time when that server is working.
- Throughput: The average number of customers released from the system at unit time.

[1] Definitions taken from Burak Büke.



(a) Number of customers in the system over a period of 2 hours

(b) Average number of customers in the system vs $\rho = \frac{\lambda}{\mu}$

(c) Probability density function of L_{avg}

Figure 2: Plots of M/M/1 Queue

2 Routing types

Note that we only need to make a decision about how to allocate customers between servers when there are less customers than servers in the system. As such, we will consider different routing methods until all of the servers are busy (states up to n customers), since afterwards it is a simple birth-death process as seen on fig. 3, i.e.

2.1 Index-based routing

The simplest type of routing is when the first customer gets allocated to server 1, the second customer to server 2, and so on, always sending the next customer to the idle server with the smallest index. Note that in the simulation the mean service times are in increasing order, therefore this allocation is the same as choosing the server with the shortest service time. Intuitively, we expect this method to be the most efficient.

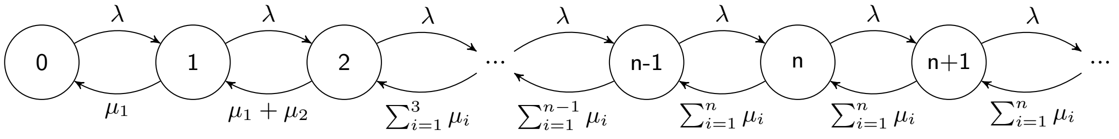


Figure 3: State transition diagram of index-based routing with n servers

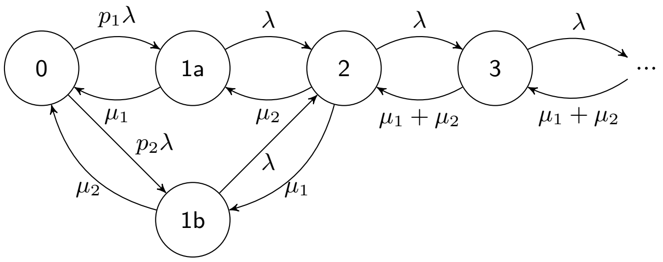


Figure 4: State diagram of random routing with 2 servers

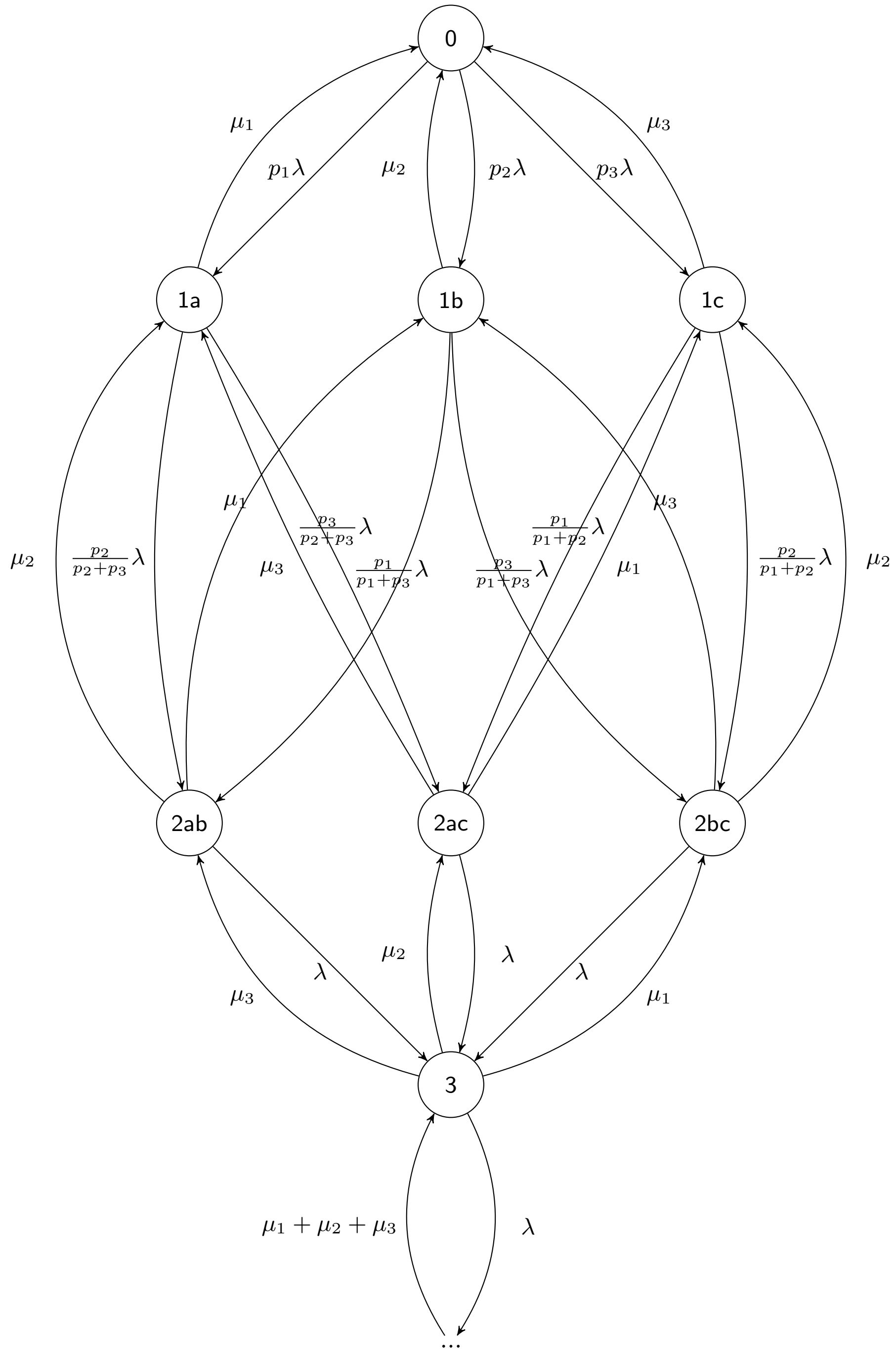


Figure 5: State diagram of random routing with 3 servers

2.2 Random routing

Although index-based routing might be efficient, it forces the faster employees to do more work than the others in total. Thus, in case we would like to distribute the customers more evenly among the servers, random routing might provide a solution. Here, we allocate a customer with probability p_i to server i. Figures 4 and 5 show the transition diagrams for such queues with 2 and 3 servers.

2.3 Routing to longest idle server

Lastly, another alternative is to allow each of the employees an equal amount of breaks. To achieve that, we may keep track of the idle time of each server, and always allocate customers to the server with maximal idle time. Fig.s10 and 7 show the transition diagrams of such queues. In the notation of the states, a capital letter means that the server is working, and a small letter means that the server is idle. Servers are listed from left to right in decreasing order of idle time.

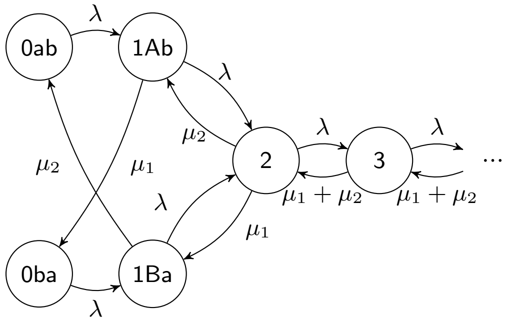


Figure 6: State diagram of routing to longest idle server with 2 servers

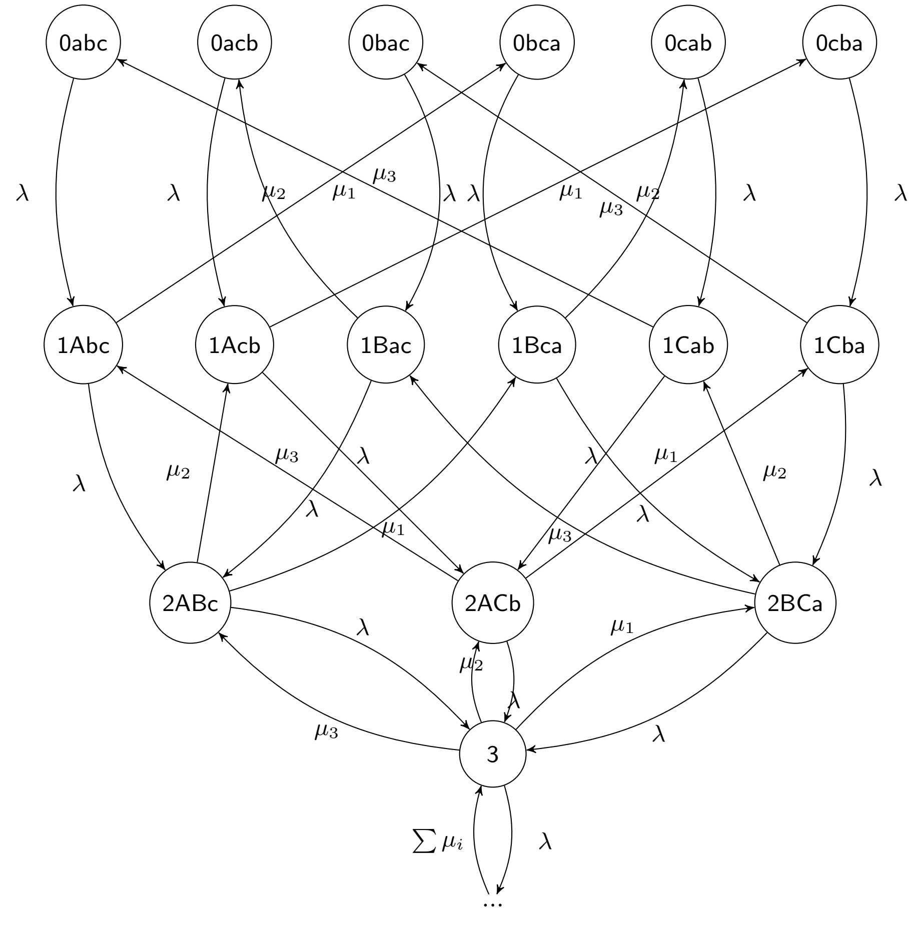


Figure 7: State diagram of routing to longest idle server with 3 servers

Note that for n servers, there are $\binom{n}{k}$ $(n-k)!$ possible states when $k(\leq n)$ customers are in the system, since k out of n servers are busy and there are $(n-k)!$ orderings of idle servers.

3 Chapman-Kolmogorov Equations

We start by examining the probability of 'jumping' from one state to another in some number of steps.[2] The (n+m)-step transition probability from state i to state j is given as

$$p_{ij}^{(n+m)} = \sum_{k \in S} p_{ik}^{n} p_{kj}^{m} \tag{1}$$

or in matrix form this can also be expressed as

$$P^{(n+m)} = P^{(n)} P^{(m)} = P^n P^m \tag{2}$$

where $P^{(n)}$ is the matrix with entry $p_{ij}^{(n)}$ in the i^{th} row and j^{th} column, therefore we can find the n-step transition probability matrix through matrix multiplication.

Given some initial state of the system $\pi(0)$, we can obtain the probability distribution after k steps as $\pi(k) = \pi(0)P^k$. Some Markov chains converge toward a given state called a steady state. The steady state probabilities can be obtained by considering the behaviour of P^k as $k \to \infty$, so the steady state vector is given as

$$\pi = \lim_{k \to \infty} \pi(k) = \lim_{k \to \infty} \pi(0) P^k. \tag{3}$$

Note that $\lim_{k \to \infty} \pi(k)$ may

- exist and converge to a fixed distribution independent of the starting distribution;
- exist and converge to some fixed distribution depending on the starting distribution;
- not exist.

If a unique π exists, and S is the set of all possible states, then it has to satisfy

$$\pi = \pi P, \sum_{j \in S} \pi_j = 1, \tag{4}$$

which will provide us with the global balance equations.

We can also express these for CTMC-s, letting P(t) be the transition matrix and Q be the generator of a CTMC. Then, P(t) is the unique solution of both

$$\frac{dP(t)}{dt} = P(t)Q \tag{5}$$

and

$$\frac{dP(t)}{dt} = QP(t). \tag{6}$$

It can be shown that in case of irreducible CTMC-s the limiting distribution p is also the unique solution of

$$pQ = 0, \sum_{i \in S} p_i = 1. \tag{7}$$

4 Steady state probabilities

Using the method described in the previous section, we can find the limiting probability distribution (also referred to as stationary probabilities, steady state probabilities orlong-term probabilities) in the form of analytical equations in terms of λ, μ_i.

4.1 The M/M/1 Queue

Transition probability matrix of the M/M/1 Queue is given as

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & \dots \\ \frac{\mu}{\mu+\lambda} & 0 & \frac{\lambda}{\mu+\lambda} & 0 & 0 & \\ 0 & \frac{\mu}{\mu+\lambda} & 0 & \frac{\lambda}{\mu+\lambda} & 0 & \\ 0 & 0 & \frac{\mu}{\mu+\lambda} & 0 & \frac{\lambda}{\mu+\lambda} & \\ \vdots & & & & & \ddots \end{bmatrix},$$

[2] This section derives the theoretical background based on Papoulis p.705-707., Stewart p.7-12., Miller p.329-332.

and its transition rate matrix is

$$Q = \begin{bmatrix} -\lambda & \lambda & 0 & 0 & 0 & \dots \\ \mu & -(\mu+\lambda) & \lambda & 0 & 0 & \\ 0 & \mu & -(\mu+\lambda) & \lambda & 0 & \\ 0 & 0 & \mu & -(\mu+\lambda) & \lambda & \\ \vdots & & & & & \ddots \end{bmatrix}.$$

It has been established before that the steady state probabilities of an M/M/1 queue are such that for all n we have

$$\pi_{n+1} = \frac{\lambda}{\mu}\pi_n \tag{8}$$

since $\pi Q = 0$ has to hold. The probabilities sum to one, so letting $\rho = \frac{\lambda}{\mu}$ gives

$$\pi_0 + \pi_1 + \pi_2 + ... = \pi_0 + \rho\pi_0 + \rho^2\pi_0 + ... = \frac{\pi_0}{1-\rho} = 1. \tag{9}$$

Thus, $\pi_0 = 1 - \rho$ and $\pi_n = \rho^n \pi_0$. The expected system size is $\frac{\rho}{1-\rho}$.

4.2 N-server queue queue with index-based routing

We have the transition rate matrix from the transition diagram in fig.3 as

$$Q = \begin{bmatrix} -\lambda & \lambda & & & & & \\ \mu_1 & -(\mu_1+\lambda) & \lambda & & & & \\ & \mu_1+\mu_2 & -(\mu_1+\mu_2+\lambda) & \lambda & & & \\ & & \mu_1+\mu_2+\mu_3 & -(\mu_1+\mu_2+\mu_3+\lambda) & \lambda & & \\ & & & & \ddots & & \\ & & & & \sum_{i=1}^{n}\mu_i & -(\sum_{i=1}^{i}\mu_i+\lambda) & \lambda \\ & & & & & & \ddots \end{bmatrix}.$$

Given two servers, similarly to the M/M/1 queue, we can write up the steady state probabilities as $\pi_1 = \frac{\lambda}{\mu_1}\pi_0$ and $\pi_{n+1} = \frac{\lambda}{\mu_1+\mu_2}\pi_n$ for $n \geq 1$. The steady state probabilities sum to one, so

$$\pi_0 + \pi_1 + \pi_2 + ... = \pi_0 + \frac{\frac{\lambda}{\mu_1}\pi_0}{1 - \frac{\lambda}{\mu_1+\mu_2}} = 1.$$

Thus, $\pi_0 = \frac{\mu_1+\mu_2-\lambda}{\mu_1+\mu_2-\lambda+\lambda/\mu_1}$ and $\pi_n = \rho^n\pi_0$ where $\rho = \frac{\lambda}{\mu_1+\mu_2}$. It can be shown that the expected queue length is $L_q = \frac{2\rho}{1-\rho^2}$.

For 3 servers, we can write $\pi_1 = \frac{\lambda}{\mu_1}\pi_0$, $\pi_2 = \frac{\lambda}{\mu_1+\mu_2}\pi_1$ and $\pi_{n+1} = \frac{\lambda}{\mu_1+\mu_2+\mu_3}\pi_n$ for $n \geq 2$. Thus,

$$\pi_0 + \pi_1 + \pi_2 + ... = \pi_0 + \frac{\lambda}{\mu_1}\pi_0 + \frac{\lambda^2}{\mu_1(\mu_1+\mu_2)}\pi_0 + \frac{\frac{\lambda^3}{\mu_1(\mu_1+\mu_2)(\mu_1+\mu_2+\mu_3)}\pi_0}{1 - \frac{\lambda}{\mu_1+\mu_2+\mu_3}} = 1.$$

In general, for n servers, we have $\pi_k = \frac{\lambda^k}{\prod_{j=1}^{k}\sum_{i=1}^{j}\mu_i}\pi_0$ if $0 < k < n$, and $\pi_k = \frac{\lambda^k}{\prod_{j=1}^{n}\sum_{i=1}^{j}\mu_i}\pi_0$ for $k \geq n$.

4.3 Random routing

The transition rate matrix of a 2-server queue is given from fig.4 as

	0	1a	1b	2	3	4
0	$-\lambda$	$p_1\lambda$	$p_2\lambda$	0	0	0
1a	μ_1	$-(\mu_1+\lambda)$	0	λ	0	0
1b	μ_2	0	$-(\mu_2+\lambda)$	λ	0	0
2	0	μ_2	μ_1	$-(\mu_1+\mu_2+\lambda)$	λ	0
3	0	0	0	$\mu_1+\mu_2$	$-(\mu_1+\mu_2+\lambda)$	λ

Using the global balance equations, we get that

$$\begin{bmatrix} -\lambda & \mu_1 & \mu_2 \\ p\lambda & -(\mu_1+\lambda) & 0 \\ (1-p)\lambda & 0 & -(\mu_2+\lambda) \end{bmatrix} \cdot \begin{bmatrix} \pi_0 \\ \pi_{1a} \\ \pi_{1b} \end{bmatrix} = \begin{bmatrix} 0 \\ -\mu_2 \\ -\mu_1 \end{bmatrix} \pi_2.$$

Thus,

$$\begin{aligned}
\begin{bmatrix} \pi_0 \\ \pi_{1a} \\ \pi_{1b} \end{bmatrix} &= \begin{bmatrix} -\lambda & \mu_1 & \mu_2 \\ p\lambda & -(\mu_1+\lambda) & 0 \\ (1-p)\lambda & 0 & -(\mu_2+\lambda) \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ -\mu_2 \\ -\mu_1 \end{bmatrix} \pi_2 \\
&= \begin{bmatrix} \frac{(\lambda+\mu_1)(\lambda+\mu_2)}{\lambda^2(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} & \frac{\mu_1(\lambda+\mu_2)}{\lambda^2(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} & \frac{\mu_2(\lambda+\mu_1)}{\lambda^2(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} \\ \frac{p(\lambda+\mu_2)}{\lambda(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} & \frac{\lambda+\mu_2 p}{\lambda(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} & \frac{\mu_2 p}{\lambda(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} \\ \frac{(\lambda+\mu_1)(p-1)}{\lambda(\lambda-\mu_1 p+\mu_1+\mu_2 p)} & \frac{\mu_1(p-1)}{\lambda(\lambda-\mu_1 p+\mu_1+\mu_2 p)} & \frac{-\lambda+\mu_1 p-\mu_1}{\lambda(\lambda-\mu_1 p+\mu_1+\mu_2 p)} \end{bmatrix} \begin{bmatrix} 0 \\ -\mu_2 \\ -\mu_1 \end{bmatrix} \pi_2 \\
&= \begin{bmatrix} \frac{-\mu_1\mu_2(2\lambda+\mu_1+\mu_2)}{\lambda^2(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} \\ \frac{-\mu_2(\lambda+\mu_2 p+\mu_1)}{\lambda(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} \\ \frac{\mu_1(-\lambda+\mu_1 p-\mu_1+\mu_2(p-1))}{\lambda(-\lambda+\mu_1 p-\mu_1-\mu_2 p)} \end{bmatrix} \pi_2.
\end{aligned}$$

Hence, the steady state probabilities are obtained as below using the condition $\sum_i^\infty \pi_i = 1$ and the fact that the sum of terms above π_2 form a geometric series with common ratio $\frac{\lambda}{\sum_i \mu_i}$, therefore

$$\sum_{i=2}^{\infty} \pi_i = \frac{\pi_2}{1 - \frac{\lambda}{\mu_1+\mu_2}}.$$

Hence,

$$\pi_2 \; (\frac{\mu_1\lambda(-\lambda+(\mu_2-\mu_1)(p-1)) - \mu_2\lambda(\lambda+\mu_2 p+\mu_1) - \mu_1\mu_2(2\lambda+\mu_1+\mu_2)}{\lambda^2(-\lambda+\mu_1(p-1)-\mu_2 p)} + \frac{\mu_1+\mu_2}{\mu_1+\mu_2-\lambda}) = 1,$$

$$\pi_2 = (\frac{\mu_1\lambda(-\lambda+(\mu_2-\mu_1)(p-1)) - \mu_2\lambda(\lambda+\mu_2 p+\mu_1) - \mu_1\mu_2(2\lambda+\mu_1+\mu_2)}{\lambda^2(-\lambda+\mu_1(p-1)-\mu_2 p)} + \frac{\mu_1+\mu_2}{\mu_1+\mu_2-\lambda})^{-1}.$$

The transition rate matrix of the 3-server queue follows from fig.5 in a similar fashion as below.

	0	1a	1b	1c	2ab	2ac	2bc	3
0	$-\lambda$	$p_1\lambda$	$p_2\lambda$	$p_3\lambda$	0	0	0	0
1a	μ_1	$-(\mu_1+\lambda)$	0	0	$\frac{p_2}{p_2+p_3}\lambda$	$\frac{p_3}{p_2+p_3}\lambda$	0	0
1b	μ_2	0	$-(\mu_2+\lambda)$	0	$\frac{p_1}{p_1+p_3}\lambda$	0	$\frac{p_3}{p_1+p_3}\lambda$	0
1c	μ_3	0	0	$-(\mu_3+\lambda)$	0	$\frac{p_1}{p_1+p_2}\lambda$	$\frac{p_2}{p_1+p_2}\lambda$	0
2ab	0	μ_2	μ_1	0	$-(\mu_1+\mu_2+\lambda)$	0	0	λ
2ac	0	μ_3	0	μ_1	0	$-(\mu_1+\mu_3+\lambda)$	0	λ
2bc	0	0	μ_3	μ_2	0	0	$-(\mu_2+\mu_3+\lambda)$	λ
3	0	0	0	0	μ_3	μ_2	μ_1	$-(\sum \mu_i + \lambda)$

Then,

$$\begin{bmatrix} -\lambda & \mu_1 & \mu_2 & \mu_3 & 0 & 0 & 0 \\ p_1\lambda & -(\mu_1+\lambda) & 0 & 0 & \mu_2 & \mu_3 & 0 \\ p_2\lambda & 0 & -(\mu_2+\lambda) & 0 & \mu_1 & 0 & \mu_3 \\ p_3\lambda & 0 & 0 & -(\mu_3+\lambda) & 0 & \mu_1 & \mu_2 \\ 0 & \frac{p_2}{p_2+p_3}\lambda & \frac{p_1}{p_1+p_3}\lambda & 0 & -(\mu_1+\mu_2+\lambda) & 0 & 0 \\ 0 & \frac{p_3}{p_2+p_3}\lambda & 0 & \frac{p_1}{p_1+p_2}\lambda & 0 & -(\mu_1+\mu_3+\lambda) & 0 \\ 0 & 0 & \frac{p_3}{p_1+p_3}\lambda & \frac{p_2}{p_1+p_2}\lambda & 0 & 0 & -(\mu_2+\mu_3+\lambda) \end{bmatrix} \begin{bmatrix} \pi_0 \\ \pi_{1a} \\ \pi_{1b} \\ \pi_{1c} \\ \pi_{2ab} \\ \pi_{2ac} \\ \pi_{2bc} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -\mu_3 \\ -\mu_2 \\ -\mu_1 \end{bmatrix} \pi_3.$$

Left-multiplying both sides by the inverse of the matrix will give us $\pi_0, ...\pi_{2bc}$ in terms of π_3. Also,

$$\sum_{i=3}^{\infty} \pi_i = \frac{\pi_3}{1 - \frac{\lambda}{\sum_i \mu_i}}.$$

Using $\sum_i \pi_i = 1$ will give us π_3. This can be more easily calculated using a computer program.[3]

[3] See appendix for code on GitHub.

4.4 Routing to longest idle server

Lastly, the transition matrix of 2-server queue is given from fig.10 as

	0ab	0ba	1Ab	1Ba	2	3
0ab	$-\lambda$	0	λ	0	0	0
0ba	0	$-\lambda$	0	λ	0	0
1Ab	0	μ_1	$-(\mu_1+\lambda)$	0	λ	0
1Ba	μ_2	0	0	$-(\mu_2+\lambda)$	λ	0
2	0	0	μ_2	μ_1	$-(\mu_1+\mu_2+\lambda)$	λ

.

Similarly as before,

$$\begin{bmatrix} -\lambda & 0 & 0 & \mu_2 \\ 0 & -\lambda & \mu_1 & 0 \\ \lambda & 0 & -(\mu_1+\lambda) & 0 \\ 0 & \lambda & 0 & -(\mu_2+\lambda) \end{bmatrix} \begin{bmatrix} \pi_{0ab} \\ \pi_{0ba} \\ \pi_{1Ab} \\ \pi_{1Ba} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\mu_2 \\ -\mu_1 \end{bmatrix} \pi_2. \tag{10}$$

Then, left-multiplying both sides of equation 10 by the inverse of the matrix will give us $\pi_0, ...\pi_{2bc}$ in terms of π_2. Also,

$$\sum_{i=2}^{\infty} \pi_i = \frac{\pi_2}{1 - \frac{\lambda}{\sum_i \mu_i}}.$$

. Using $\sum_i \pi_i = 1$ will give us π_2.[4]

Since the transition rate matrix of a 3-server queue can be constructed from the state diagram in fig. 7, and the calculations of the steady state probabilities are exactly the same as before, the details about this queue will be omitted here. However, the Python code to calculate it is available on GitHub.[5]

5 Average queue lengths of systems with abandonment

5.1 Abandonment

Abandonment can only happen when all of the servers are occupied, thus we only need to look at the states where there are more than n customers in the system. Fig.8 shows how the abandonment rate, γ is multiplied by the number of people waiting in line in the transition rates to a lower state.

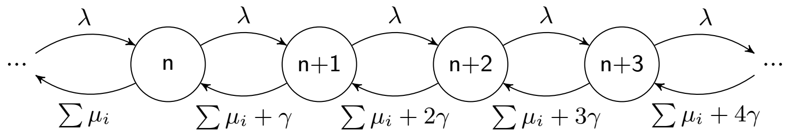


Figure 8: State transition diagram of n-server queue with abandonment

Thus, for $k \geq n$, we have $\pi_k = \frac{\lambda}{\sum \mu_i + (k-n)\gamma} \pi_{k-1}$. Again, the calculations are much more easily done in Python (with numerical values).[6]

5.2 Average queue length

The simulations shown in fig.11 were run with the following parameters:

- Termination time: 10 hours
- mean service rate: $\mu_i \sim U[1,3]$/hour or $U[1.9, 2.1]$/hour
- mean arrival rate: $\lambda = 5$/hour
- mean abandonment rate: $\gamma = 2$/hour (i.e. average patience time is 0.5 hours)

[4] See appendix for code on GitHub.

[5] See appendix.

[6] See appendix for code on GitHub.

Note that examining average queue length of queues without abandonment would be of little use since the expected queue length is infinity if $\rho = \frac{\lambda}{\sum \mu_i} > 1$ and the observed average queue lengths for $\rho < 1$ that will also somewhat increase with termination time, i.e. we would observer longer queues as the simulation is run for longer due to the randomness of the system. Fig. 9 shows how a system without abandonment becomes unstable when $\lambda \geq \sum \mu_i$. Since here a 3-server queue with service rates $\mu_i \sim [1, 3]$ /hour was simulated for 100 hours, the average queue length starts to tend to infinity from $\lambda = 3 \cdot 2 = 6$.

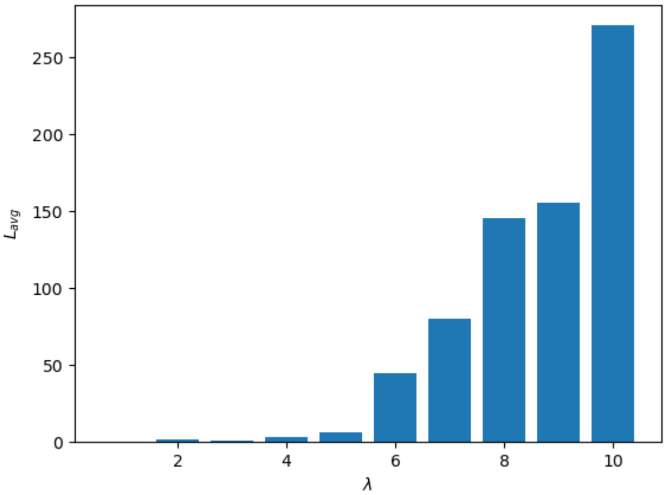


Figure 9: Average number of customers in the system for different values of arrival rate in case of a 3-server queue with random routing, equal service rates of $\mu_i \sim [1, 3]$ and no abandonment

By the Tower Property,

$$\mathbb{E}(L_{avg}) = \sum_k \mathbb{E}(L_{avg} | \mu = \mu_k) \mathbb{P}(\mu = \mu_k)$$

.

Since the mean service rates are assumed to have a uniform distribution, it is sufficient to divide up the interval on which μ is uniform (here: [1,3] or [1.9, 2.1]) into sub-intervals of equal length so that the probability of μ falling into any of the sub-intervals is equal. Then, take a uniformly distributed random sample from each interval, run the simulation for each combinations of samples and find the average of the queue lengths. Note that this is done for each combination of the 20 samples taken from μ_1, μ_2 and μ_3, so we average the findings of the simulation run 20^3 times again. Since we want to graph L_{avg} against $\rho = \frac{\lambda}{\sum_{i=1}^{n} \mu_i}$, we calculate the average queue length for different values of λ.

We can observe in fig. 10 how the analytical solutions of mean queue lengths compare in the case of different routing types. As expected, index-based routing (i.e. always choosing the fastest server) is the most efficient producing the lowest average queue lengths. Interestingly, routing to the longest idle server produces almost identical results. However, as discussed before, routing to the longest idle server treats employees more fairly, loading them equally for equal pay, therefore it appears to be an ideal routing mechanism.

In figure 11 it is clear that a smaller variance in μ_i will always result in slightly shorter queues, as we could expect. Also, if we were to run the simulation for longer, the results obtained for average queue length would get gradually closer to the analytical solution.

For further analysis, other performance measures could be calculated, too, such as throughput, the utilization of each server and the proportion of customers who abandoned the queue without being served to estimate from what value of λ would it be worth to employ another server.

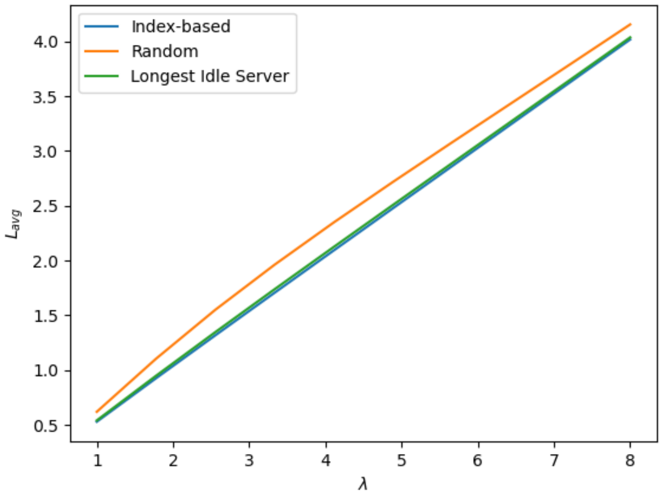
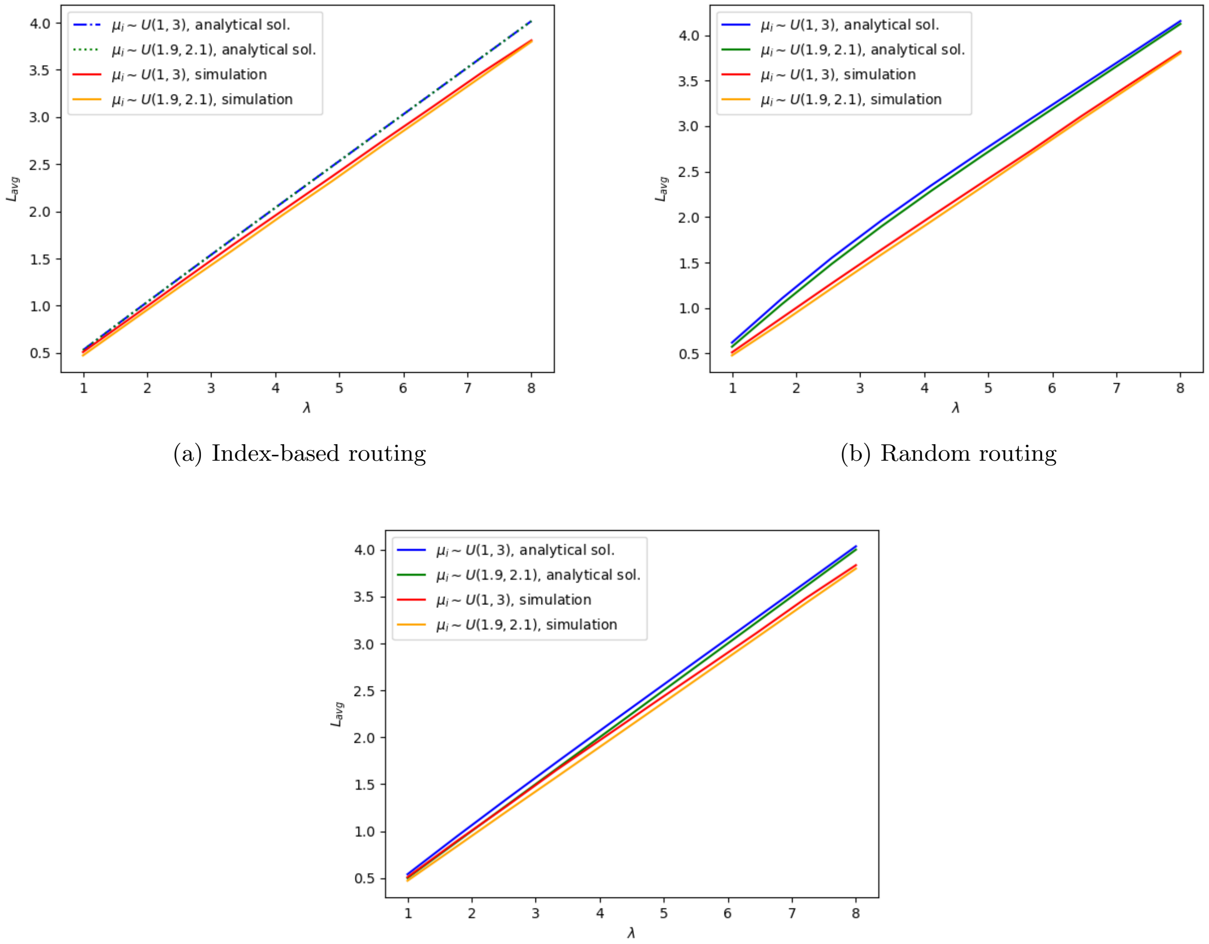


Figure 10: Average number of customers in the system against mean arrival rate for different routing types. Analytical solutions of 3-server queues with mean service rates $\mu_i \overset{i.i.d.}{\sim} U[1,3]$.



(a) Index-based routing

(b) Random routing

(c) Routing to longest idle server

Figure 11: Average number of customers in the system vs mean arrival rate for different routing methods and different variances of μ_i

6 Two types of customer problems

An even more complex queuing scenario arises when we distinguish between two types of customers arriving with different kinds of problems to be solved, and every server is capable of handling both types of customers, but with different mean service rates as shown in fig. 12.

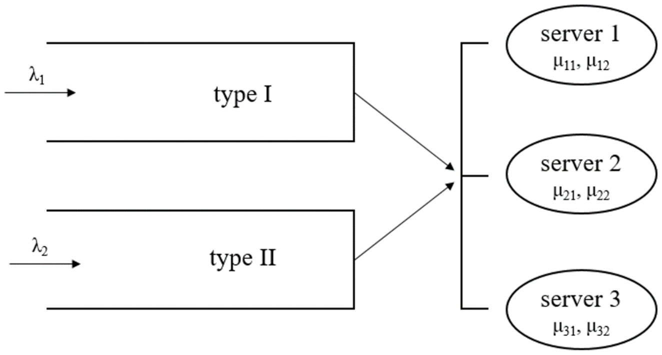


Figure 12: Two types of customers arriving at different rates, forming two queues and 3 servers cater for them with different mean service rates.

Since there are several servers and several queues, a routing method is required both for selecting a server and for choosing the queue from which the next customer will be served. The techniques of choosing the next customer's type are:

- First come, first served: irrespective of the type of customer problem, the one who arrived the earliest is served next.
- Random: the queue from which the next customer is served is chosen randomly.
- Longest queue first: the next customer is chosen from the longer queue.

We could examine all three of these methods in combination with all three of the server routing techniques used before, however, here we will limit focus on index-based routing.[7] Once the simulation is obtained, an interesting question to examine is whether the dependence of service rates for type I and type II customers has any effect on average queue length.

Since mean service rates are assumed to be uniformly distributed as $\mu_{i1} \overset{i.i.d.}{\sim} U[1,3]$, it is still sufficient to take uniform random samples of μ_{11}, μ_{21} and μ_{31} from 20 sub-intervals of [1,3] of equal width, calculate L_{avg} for each combination and take the average to find $\mathbb{E}(L_{avg})$ as before. In this case, we keep the arrival rate constant at $\lambda_1 = 8, \lambda_2 = 5$.[8]

Then, we need to choose μ_{i2} such that it is either completely dependent, somewhat dependent or completely independent of μ_{i1}. To achieve complete dependence, we can set $\mu_{i2} = 0.5\mu_{i1}$. If we let $\mu_{i2} = u \cdot \mu_{i1}$ where u is a random variable such that $u \sim U[0,1]$, then μ_{i1} and μ_{i2} are somewhat dependent. To calculate the correlation between them, let $\mu_{i1} = X$ and $\mu_{i2} = Y$. Note that u and X are two uniformly distributed independent variables.

$$Corr(X,Y) = \frac{Cov(X,Y)}{\sigma_X \sigma_Y}$$

where

$$\begin{aligned} Cov(X,Y) &= \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y), \\ \mathbb{E}(XY) &= \mathbb{E}(X^2 \cdot u) = \mathbb{E}(X^2)\mathbb{E}(u), \\ \mathbb{E}(Y) &= \mathbb{E}(uX) = \mathbb{E}(u)\mathbb{E}(X). \end{aligned}$$

Thus,

$$Corr(X,Y) = \frac{\mathbb{E}(X^2)\mathbb{E}(u) - \mathbb{E}(X)^2\mathbb{E}(u)}{\sigma_X \sigma_Y} = \frac{\mathbb{E}(u)[\mathbb{E}(X^2) - \mathbb{E}(X)^2]}{\sigma_X \sigma_Y} = \frac{\mathbb{E}(u)\sigma_X^2}{\sigma_X \sigma_Y} = \frac{\mathbb{E}(u)\sigma_X}{\sigma_Y}$$

[7] See appendix for code on GitHub.
[8] See appendix for code on GitHub.

To find σ_X, we need

$$\mathbb{E}(X) = 2,$$
$$\mathbb{E}(X^2) = \int_1^3 x^2 f_X(x)dx.$$

Since X is uniformly distributed between 1 and 3, its probability density function is given as

$$f_X(x) = \begin{cases} \frac{1}{2}, & \text{if } x \in [1,3] \\ 0, & \text{otherwise.} \end{cases}$$

Thus,

$$\mathbb{E}(X^2) = \int_1^3 \frac{1}{2}x^2 dx = \frac{13}{3},$$

and

$$\sigma_X = \sqrt{\mathbb{E}(X^2) - \mathbb{E}(X)^2} = \sqrt{\frac{13}{3} - 2^2} = \frac{1}{\sqrt{3}}.$$

Similarly, for σ_Y

$$\mathbb{E}(Y) = \mathbb{E}(uX) = \mathbb{E}(u)\mathbb{E}(X) = \frac{1}{2} \cdot 2 = 1,$$
$$\mathbb{E}(Y^2) = \mathbb{E}(u^2X^2) = \mathbb{E}(u^2)\mathbb{E}(X^2),$$
$$\mathbb{E}(u^2) = \int_0^1 u^2 f_u(u)du.$$

And since u is uniformly distributed over $[0,1]$,

$$f_u(u) = \begin{cases} 1, & \text{if } x \in [0,1] \\ 0, & \text{otherwise.} \end{cases}$$

Thus,

$$\mathbb{E}(u^2) = \int_0^1 u^2 du = \frac{1}{3},$$
$$\mathbb{E}(Y^2) = \frac{1}{3}\frac{13}{3} = \frac{13}{9},$$
$$\sigma_Y = \sqrt{\mathbb{E}(Y^2) - \mathbb{E}(Y)^2} = \sqrt{\frac{13}{9} - 1^2} = \frac{2}{3}.$$

And substituting in gives that

$$Corr(X,Y) = \frac{\frac{1}{2}\frac{1}{\sqrt{3}}}{\frac{2}{3}} = \frac{\sqrt{3}}{4} \approx 0.433.$$

We can choose different bounds for the uniform random variable u, e.g. $U[0.15, 0.85]$, $U[0.25, 0.75]$, $U[0.375, 0.625]$ or $U[0.45, 0.55]$; gradually decreasing the interval on which u is uniform until we reach perfect dependence at $u = 0.5$.[9] Fig. 13a shows where the points (μ_{i1}, μ_{i2}) might fall depending on how close the correlation is, with region A demonstrating complete dependence.

One problem with this design is that it does not account for correlations between μ_{i1} and μ_{i2} below 0.43. Thus, an alternative design is shown in fig. 13b. Clearly, if the points (μ_{i1}, μ_{i2}) are randomly scattered over the square defined by $1 \le \mu_{i1} \le 3$; $4 \le \mu_{i2} \le 6$; there is 0 correlation between them. Then, we can narrow the region down to get more closely correlated variables. Assuming that the points (μ_{i1}, μ_{i2}) are uniform over the regions demonstrated in fig. 13b, we cannot continue taking samples of $\mu_i 1$ from sub-intervals of [1,3] of uniform width as they would not be uniformly distributed over the shaded regions. Instead, we would over-sample the areas close to the endpoints and under-sample the middle section. Thus, we need to divide up each of the shaded regions into 20 parts of equal areas. The bounds of μ_{i1} for splitting up 9 different regions (including region B and C) into 20 sub-regions can be found on GitHub.

[9] Find calculated correlations for other bounds of u on GitHub.

Note that neither μ_{i1}, not μ_{i2} is uniformly distributed in this example, but instead the combination of the two, the data points (μ_{i1}, μ_{i2}) are assumed to have a uniform distribution over each of the shaded regions in fig. 13b. The probability density function of the service rates μ_{i1} and μ_{i2} is shown in fig. 14. These are also useful when calculating the correlation between the service rates. It is clear that $\mathbb{E}(\mu_{i1}) = 2$ and $\mathbb{E}(\mu_{i2}) = 5$, so now we only need to determine $\mathbb{E}(\mu_{i1}^2), \mathbb{E}(\mu_{i2}^2)$ and $\mathbb{E}(\mu_{i1}\mu_{i2})$. Let $\mu_{i1} = X$, and $\mu_{i2} = Y$ as before. Then, for some region R of area A, we have

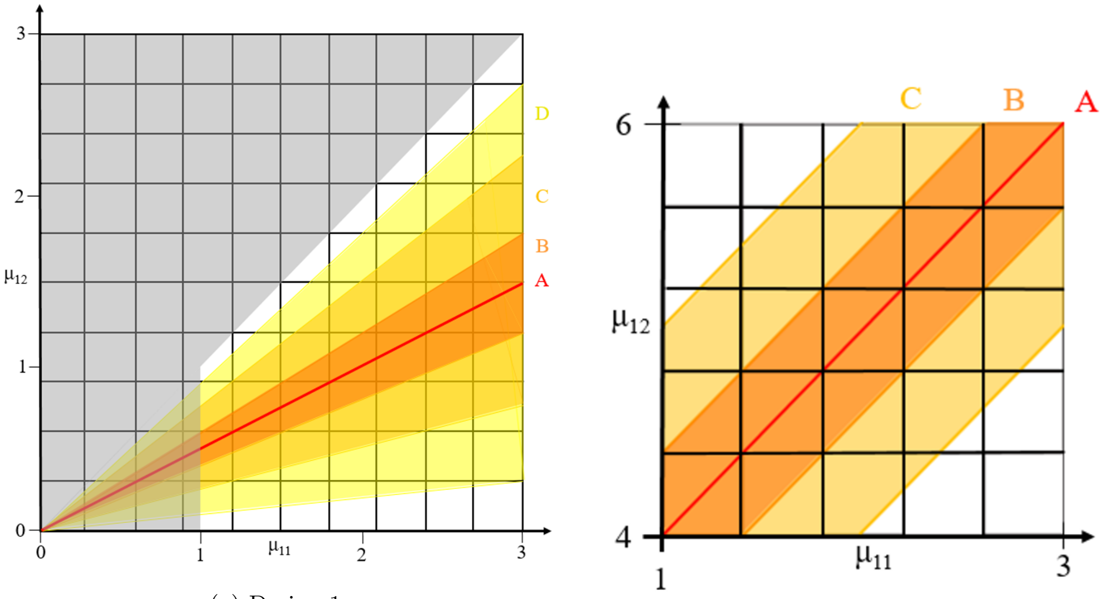
$$\begin{aligned}
\mathbb{E}(X^2) &= \int_1^3 x^2 f_X(x)\,dx, \\
\mathbb{E}(Y^2) &= \int_4^6 y^2 f_Y(y)\,dx, \\
\mathbb{E}(XY) &= \iint_R xy f_{XY}(x,y)\,dy\,dx.
\end{aligned}$$

where

$$f_{XY}(x,y) = \begin{cases} \frac{1}{A}, & \text{if } (x,y) \in R \\ 0, & \text{otherwise.} \end{cases}$$

Furthermore, using the probability density functions of $f_X(x)$ and $f_Y(y)$ shown in fig. 14, we can compute $\text{Corr}(\mu_{i1}, \mu_{i2})$ for any region.[10]

Alternatively, we could have estimated the correlations using Monte-Carlo simulation by choosing several uniformly distributed data points from each of the regions and finding Pearson's r statistic.[11]



(a) Design 1.
A) Perfect dependence, $\text{Corr}(\mu_{i1}, \mu_{i2}) = 1$;
B) $u \sim U[0.4, 0.6]$;
C) $u \sim U[0.25, 0.75]$;
D) $u \sim U[0.1, 0.9]$.

(b) Design 2.
A) Perfect dependence, $\text{Corr}(\mu_{i1}, \mu_{i2}) = 1$;
B) $\text{Corr}(\mu_{i1}, \mu_{i2}) = 0.91$;
C) $\text{Corr}(\mu_{i1}, \mu_{i2}) = 0.5$.

Figure 13: Dependence of service rates

[10] Calculations can be found on GitHub.
[11] Estimated correlations are also available on GitHub.

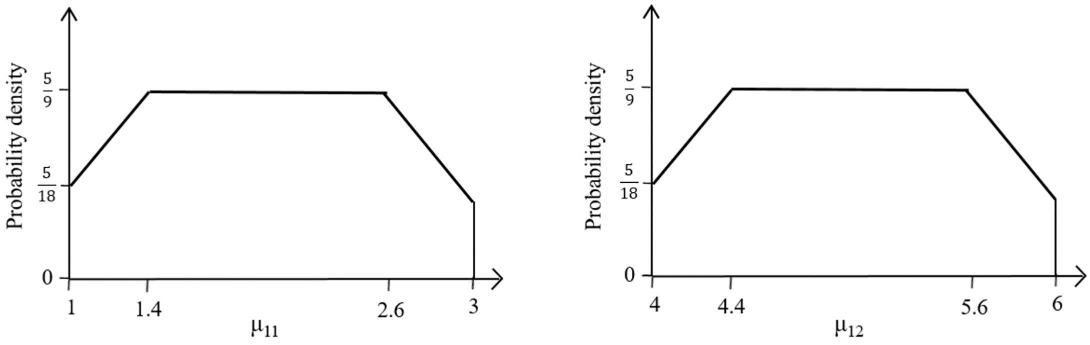
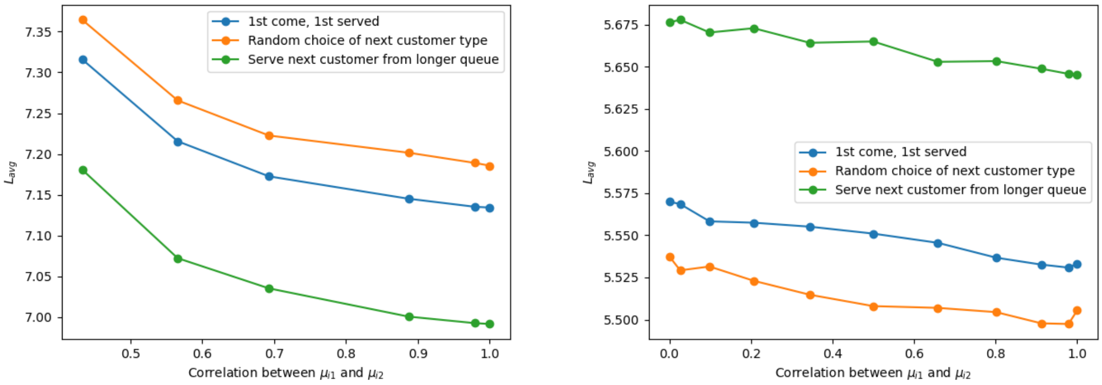


Figure 14: Probability density function of μ_{i1} and μ_{i2} for region B in fig.13b



(a) Design 1

(b) Design 2

Figure 15: Mean queue length against correlation of type I and type II service rates for different methods of customer routing.

Looking at fig.15a and 15b, average queue length appears to be decreasing the more dependent μ_{i1} and μ_{i2} are in all cases. However, we cannot generalise any statement about the relative performance of the different routing methods since they are the direct consequence of the parameters used. For example, the two designs produced vastly different average queue lengths since μ_i2 was much less in design 1 than in design 2. Furthermore, choosing a customer from the longer queue appears to work best for design 1, however, random routing is better for design 2.

Also note that this is not a good indicator whether the system is as efficient as possible and whether supply meets demand. We should further examine the abandonment ratios and utilization of the servers to make conclusions on this matter.

Appendix

GitHub Repository containing all code. `https://github.com/bagyokinga/Queue-Simulation.git`

References

[1] Büke, Burak. ”Simulation Lecture Notes.” Simulation MATH11028, University of Edinburgh.

[2] Gans, Noah, et al. “Telephone Call Centers: Tutorial, Review, and Research Prospects.” *Manufacturing & Service Operations Management*, vol. 5, no. 2, 2003, pp. 79–141., https://doi.org/10.1287/msom.5.2.79.16071.

[3] Grogan, Paul. ”Queuing System Discrete Event Simulation in Python (Event-scheduling).” YouTube, 10 Aug., 2022, `https://www.youtube.com/watch?v=oJyf8Q0KLRY.`

[4] Koole, Ger. *Call Center Mathematics: A Scientific Method for Understanding and Improving Contact Centers.* Department of Mathematics, Vrije Universiteit Amsterdam, Amsterdam, 2007.

[5] Ledder, Glenn. *Introduction to Queueing Theory: A Modeling Perspective.* Operations Research, University of Nebraska-Lincoln, 2019.

[6] Medhi, Jyotiprasad. *Stochastic Models in Queueing Theory.* Second ed., Academic Press, 2003.

[7] Miller, Scott L., and Donald G. Childers. *Probability and Random Processes: With Applications to Signal Processing and Communications.* Academic Press, 2012.

[8] Papoulis, Athanasios, and S. Unnikrishna Pillai. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, 2002.

[9] Shortle, John F., et al. *Fundamentals of Queueing Theory.* Fifth ed., John Wiley &; Sons, Inc., 2018.

[10] Stewart, William J. *Introduction to the Numerical Solution of Markov Chains.* Princeton University Press, 1994.