Analysis of Queueing Systems with Sample Paths and Simulation
Companion

Nicky D. van Foreest

March 4, 2020

Changes:

- Figure 8
- Section 1.7.
- Section 1.3, added python code to a few exercises to show how to implement the recursions.
- Moved part of 1.5.9 to 1.5.13.
- Moved old intro of Section 2.4 to 2.1.6.
- Formulas in intro Section 2.6.
- Section 2.9: rewritten

CONTENTS

INTRODUCTION

MOTIVATION AND EXAMPLES

Queueing systems abound, and the analysis and control of queueing systems are major topics in the control, performance evaluation and optimization of production and service systems.

At my local supermarket, for instance, any customer that joins a queue of 4 or more customers gets his/her groceries for free. Of course, there are some constraints: at least one of the cashier facilities has to be unoccupied by a server and the customers in queue should be equally divided over the cashiers that are open (and perhaps there are some further rules, of which I am unaware). When $\pi(n)$ denotes fraction of customers that 'see upon arrival' the system with n customers, the manager that controls the occupation of the cashier positions is focused on keeping $\pi(4)+\pi(5)+\cdots$, i.e., the fraction of customers that see upon arrival a queue length exceeding 3, very small. In a sense, this is easy enough: just hire many cashiers. However, the cost of personnel may then outweigh the yearly average cost of paying the customer penalties. Thus, the manager's problem becomes to plan and control the service capacity in such a way that both the penalties and the personnel cost are small.

Fast food restaurants also deal with many interesting queueing situations. Consider, for instance, the preparation of hamburgers. Typically, hamburgers are made-to-stock, in other words, they are prepared before the actual demand has arrived. Thus, hamburgers in stock can be interpreted as customers in queue waiting for service, where the service time is the time between the arrival of two customers that buy hamburgers. The hamburgers have a typical lifetime, and they have to be scrapped if they remain on the shelf longer than a specified amount of time. Thus, the waiting time of hamburgers has to be closely monitored. Of course, it is easy to achieve zero scrap cost, simply by keeping no stock at all. However, to prevent lost-sales, it is very important to maintain a certain amount of hamburgers in stock. Thus, the manager has to balance the scrap cost against the cost of lost sales. In more formal terms, the problem is to choose a policy to prepare hamburgers such that the cost of excess waiting time (scrap) is balanced against the cost of an empty queue (lost sales).

Service systems, such as hospitals, call centers, courts, and so on, have a certain capacity available to serve customers. The performance of such systems is, in part, measured by the total number of jobs processed per year and the fraction of jobs processed within a certain time frame between receiving and closing the job. Here the problem is to organize the capacity such that the sojourn time, i.e., the typical time a job spends in the system, does not exceed some threshold, and such that the system achieves a certain throughput, i.e., jobs served per year.

Clearly, all of the above systems can be seen as queueing systems that have to be monitored and controlled to achieve a certain performance. The performance analysis of such systems can, typically, be characterized by the following performance measures:

1. The fraction of time $p(n)$ that the system contains n customers. In particular, $1-p(0)$, i.e., the fraction of time the system contains jobs, is important, as this is a measure of the time-average occupancy of the servers, hence related to personnel cost.

2. The fraction of customers $\pi(n)$ that 'see upon arrival' the system with n customers. This measure relates to customer perception and lost sales, i.e., fractions of arriving customers that do not enter the system.

3. The average, variance, and/or distribution of the waiting time.

4. The average, variance, and/or distribution of the number of customers in the system.

Here the system can be anything that is capable of holding jobs, such as a queue, the server(s), an entire court, patients waiting for an MRI scan in a hospital, and so on.

It is important to realize that a queueing system can, typically, be decomposed into *two subsystems*: the queue itself and the service system. Thus, we are concerned with three types of waiting: waiting in queue, i.e., *queueing time*, waiting while being in service, i.e., the *service time*, and the total waiting time in the system, i.e., the *sojourn time*.

ORGANIZATION

In these notes, we will be primarily concerned with making models of queueing systems such that we can compute or estimate the above-mentioned performance measures.

In Chapter 1 we construct queueing systems in discrete time and continuous time. By implementing these constructions in Python code, we can then simulate and analyze such systems. Besides that, simulation provides ample motivation for why and how we deal with queueing systems and is useful for analyzing realistic systems, as mathematical models have severe shortcomings in such cases. Consider, for example, the service process at a check-in desk of KLM. Business customers and economy customers are served by two separate queueing systems. The business customers are served by one server, server A say, while the economy class customers by three servers, say. What would happen to the sojourn time of the business customers if server A would be allowed to serve economy class customers when the business queue is empty? For the analysis of such complicated control policies, simulation is the most natural approach.

In Chapter 2 and Chapter 3 we derive exact and approximate models, respectively, for single-station queueing systems. The benefit of such models is that they offer structural insights into the behavior of the system and scaling laws, such as that the average waiting time scales (more or less) linearly in the variance of the service times of individual customers. The main idea is to consider the *sample paths of a queueing process*, and assume that a typical sample path captures the 'normal' stochastic behavior of the system. This sample-path approach has two advantages. In the first place, many of the theoretical results follow from very concrete aspects of these sample paths. Second, the analysis of sample-paths carries over right away to simulation. In fact, simulation of a queueing system offers us one (or more) sample path(s), and based on such sample paths, we derive behavioral and statistical properties of the system. In fact, the performance measures defined for sample paths are precisely those we compute with simulation.

In Chapter 4 we construct algorithms to analyze open and closed queueing networks. Many of the sample path results developed for the single-station case can be applied to these networks. Thus, via sample-path methods we relate the theory, algorithms, and simulation of queueing systems. For this part we refer to the book of Prof. Zijm; the present set of notes augments the discussion there.

Our aim is not to provide rigorous proofs for all results derived in the book. For this we refer to the following books.

1. Bolch et al. [2006]
2. Capiński and Zastawniak [2003]
3. El-Taha and Stidham Jr. [1998]
4. Tijms [1994] and/or Tijms [2003]

EXERCISES

I urge you to try to make as many exercises as possible. The main text contains hardly any examples or derivations: the exercises *illustrate* the material and force you to *think* about the technical parts. The exercises require many of the tools you learned previously in courses on calculus, probability, and linear algebra. Here you can see them applied. Moreover, many of these tools will be useful for other, future, courses. Thus, the investments made here will pay off for the rest of your (student) career.

You'll notice that some of these problems are quite difficult, often not because the problem itself is difficult, but because you need to combine a substantial amount of knowledge all at the same time. All this takes time and effort. Realize that the exercises are not intended to be easy (otherwise we could have been satisfied with computing $1+1$).

The companion document gives hints and solutions to all problems. The solutions spell out nearly every intermediate step. For most of you all, this detail is not necessary, but over the years I got many questions like: "how do you go from 'here' to 'there'?" As service, I then added such intermediate steps. The companion document also contains many additional simple exercises and old exam questions.

Exercises marked as 'not obligatory' are interesting, but hard. I will not use this in an exam.

ACKNOWLEDGEMENTS

I would like to acknowledge dr. J.W. Nieuwenhuis for our many discussions on queueing theory. To convince him about the more formal aspects, sample-path arguments proved very useful. Prof. dr. W.H.M. Zijm allowed me to use the first few chapters of his book. Finally, I thank my students for submitting many improvements via github. It's very motivating to see a book like this turn into a joint piece of work.

1

CONSTRUCTION AND SIMULATION OF QUEUEING SYSTEMS

In this chapter we start with a discussion of the Poisson process. We then construct queueing processes in discrete time and apply the Poisson process to model the number of arrivals in periods of fixed length. In Section 1.4 we relate the exponential distribution to the Poisson distribution. The exponential distribution often serves as a good model for inter-arrival times of individual jobs. As such, this is a key component of the construction of queueing processes in continuous time. As it turns out, both ways to construct queueing processes are easily implemented as computer programs, thereby allowing us to use simulation to analyze queueing systems. In passing, we develop a number of performance measures to provide insight into the (transient and average) behavior of queueing processes. We then introduce a useful set of shorthands to distinguish between different queueing models, and finish the chapter with a motivation why we focus on a steady-state analysis of queueing systems in the remainder of the book.

We assume that you *know all* results of Section 1.1.

1.1 PRELIMINARIES

Theory and Exercises

Here is an overview of concepts you are supposed to have seen in earlier courses. We will use these concepts over and over in the rest of the course.

We use the notation:

$$\begin{aligned}
[x]^+ &= \max\{x, 0\},\\
f(x-) &= \lim_{y \uparrow x} f(y),\\
f(x+) &= \lim_{y \downarrow x} f(y),\\
\mathbb{1}_A &= \begin{cases} 1, & \text{if } A \text{ is true,} \\ 0, & \text{if } A \text{ is false.} \end{cases},
\end{aligned}$$

where the last equation defines an *indicator variable*.

The function $f(h) = o(h)$ means that f is such that $f(h)/h \to 0$ as $h \to 0$. If we write $f(h) = o(h)$ it is implicit that $|h| \ll 1$. We call this *small o notation*.

1.1.1. *Let c be a constant (in $\mathbb{R}$) and the functions f and g both of $o(h)$. Then show that (1) $f(h) \to 0$ when $h \to 0$, (2) $c \cdot f = o(h)$, (3) $f + g = o(h)$, and (4) $f \cdot g = o(h)$.*

You should know that:

$$(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^{n-i} b^i, \tag{1.1.1a}$$

$$e^x = \lim_{n \to \infty} (1 + x/n)^n, \tag{1.1.1b}$$

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots = \sum_{k=0}^{\infty} \frac{x^k}{k!}, \tag{1.1.1c}$$

$$\sum_{n=0}^{N} \alpha^n = \frac{1-\alpha^{N+1}}{1-\alpha}. \tag{1.1.1d}$$

1.1.2. *Why is* $e^x = 1 + x + o(x)$*?*

You should know that for a non-negative, integer-valued random variable X with *probability mass function* $f(k) = \mathsf{P}(X = k)$,

$$X = \sum_{n=0}^{\infty} X\, \mathbb{1}_{X=n} = \sum_{n=0}^{\infty} n\, \mathbb{1}_{X=n}, \tag{1.1.2a}$$

$$\mathsf{E}[X] = \sum_{n=0}^{\infty} n f(n), \tag{1.1.2b}$$

$$\mathsf{E}[g(X)] = \sum_{n=0}^{\infty} g(n) f(n), \tag{1.1.2c}$$

$$\mathsf{E}[\mathbb{1}_{X\le x}] = \mathsf{P}(X \le x), \tag{1.1.2d}$$

$$\mathsf{V}[X] = \mathsf{E}\left[X^2\right] - (\mathsf{E}[X])^2. \tag{1.1.2e}$$

1.1.3. *Why is* (1.1.2a) *true?*

1.1.4. *Define the* survivor function *of* X *as* $G(k) = \mathsf{P}(X > k)$*. Show that*

$$G(k) = \sum_{m=0}^{\infty} \mathbb{1}_{m>k} f(m).$$

As you will see below, this idea makes the computation of certain expressions quite a bit easier.

1.1.5. *Express the probability mass* $f(k)$ *and the survivor function* $G(k)$ *in terms of the* distribution function $F(k) = \mathsf{P}(X \le k)$ *of* X*.*

1.1.6. *Which of the following is true:* $G(k) = 1 - F(k)$*,* $G(k) = 1 - F(k-1)$*, or* $G(k) = 1 - F(k+1)$*?*

1.1.7. *Use indicator functions to prove that* $\mathsf{E}[X] = \sum_{k=0}^{\infty} G(k)$*.*

1.1.8. *Use indicator functions to prove that* $\sum_{i=0}^{\infty} iG(i) = \mathsf{E}\left[X^2\right]/2 - \mathsf{E}[X]/2$*.*

Let X be a continuous non-negative random variable with distribution function F. We write

$$\mathsf{E}[X] = \int_0^{\infty} x\, \mathrm{d}F(x)$$

for the expectation of X. Here $\mathrm{d}F(x)$ acts as a shorthand for $f(x)\mathrm{d}x$[1]. Recall that

$$\mathsf{E}[g(X)] = \int_0^{\infty} g(x)\, \mathrm{d}F(x).$$

1.1.9. *Use indicator functions to prove that* $\mathsf{E}[X] = \int_0^{\infty} x\, dF(x) = \int_0^{\infty} G(y)\, dy$*, where* $G(x) = 1 - F(x)$*.*

You should be able to use indicator functions and integration by parts to show that $\mathsf{E}\left[X^2\right] = 2\int_0^{\infty} yG(y)\,\mathrm{d}y$, where $G(x) = 1 - F(x)$, provided the second moment exists.

[1] For the interested reader, $\int x\,\mathrm{d}F(x)$ is a Lebesgue-Stieltjes integral with respect to the distribution function F.

1.1.10. *Use indicator functions to prove that for a continuous non-negative random variable X with distribution function F, $\mathsf{E}\left[X^2\right] = \int_0^\infty x^2\, dF(x) = 2\int_0^\infty yG(y)\, dy$, where $G(x) = 1 - F(x)$.*

1.1.11. *Show that $\mathsf{E}\left[X^2\right]/2 = \int_0^\infty yG(y)\, dy$ for a continuous non-negative random variable X with survivor function G.*

You should know that for the *moment-generating function* $M_X(s)$ of a random variable X and $s \in \mathbb{R}$ sufficiently small is defined as:

$$M_X(s) = \mathsf{E}\left[e^{sX}\right], \tag{1.1.3a}$$

$$M_X(s) \text{ uniquely characterizes the distribution of } X, \tag{1.1.3b}$$

$$\mathsf{E}[X] = M_X'(0) = \left.\frac{\mathrm{d}M_X(s)}{ds}\right|_{s=0}, \tag{1.1.3c}$$

$$\mathsf{E}\left[X^2\right] = M_X''(0), \tag{1.1.3d}$$

$$M_{X+Y}(s) = M_X(s) \cdot M_Y(s), \quad \text{if } X \text{ and } Y \text{ are independent.} \tag{1.1.3e}$$

1.1.12. *What is the value of $M_X(0)$?*

To help you recall the concept of *conditional probability* consider the following question.

1.1.13. *We have one gift to give to one out of three children. As we cannot divide the gift into parts, we decide to let 'fate decide'. That is, we choose a random number in the set $\{1,2,3\}$. The first child that guesses this number wins the gift. Show that the probability of winning the gift is the same for each child.*

You should know that:

$$\mathsf{P}(A \,|\, B) = \frac{\mathsf{P}(AB)}{\mathsf{P}(B)}, \quad \text{if } \mathsf{P}(B) > 0, \tag{1.1.4a}$$

$$\mathsf{P}(A) = \sum_{i=1}^{n} \mathsf{P}(AB_i) = \sum_{i=1}^{n} \mathsf{P}(A \,|\, B_i)\mathsf{P}(B_i), \quad \text{if } A = \bigcup_{i=1}^{n} B_i \text{ and } \mathsf{P}(B_i > 0) \text{ for all } i. \tag{1.1.4b}$$

Hints

h.1.1.5. This exercise is just meant to become familiar with the notation.

h.1.1.7. Write $\sum_{k=0}^{\infty} G(k) = \sum_{k=0}^{\infty}\sum_{m=k+1}^{\infty} \mathsf{P}(X = m)$, reverse the summations. Then realize that $\sum_{k=0}^{\infty} \mathbb{1}_{k<m} = m$. You should be aware that this sort of problem is just a regular probability theory problem, nothing fancy. We use/adapt the tools you learned in calculus to carry out 2D integrals (or in this case 2D summations).

h.1.1.8. $\sum_{i=0}^{\infty} iG(i) = \sum_{n=0}^{\infty} \mathsf{P}(X = n)\sum_{i=0}^{\infty} i\, \mathbb{1}_{n \geq i+1}$, and reverse the summations.

h.1.1.9. $\mathsf{E}[X] = \int_0^\infty x\, \mathrm{d}F(x) = \int_0^\infty \int_0^\infty \mathbb{1}_{y \leq x}\, \mathrm{d}y\, \mathrm{d}F(x)$.

h.1.1.10. $\int_0^\infty yG(y)\mathrm{d}y = \int_0^\infty y \int_0^\infty \mathbb{1}_{y \leq x} f(x)\, \mathrm{d}x\, \mathrm{d}y$.

h.1.1.13. For the second child, condition on the event that the first does not choose the right number. Use the definition of conditional probability: $\mathsf{P}(A|B) = \mathsf{P}(AB)/\mathsf{P}(B)$ provided $\mathsf{P}(B) > 0$.

Solutions

s.1.1.1. In fact (1) is trivial: $|f(h)| \le |f(h)/h|$ when $|h| < 1$. But it is given that the right-hand side goes to zero. For (2) and (3):

$$\lim_{h\to 0} \frac{cf(h)}{h} = c \lim_{h\to 0} \frac{f(h)}{h} = 0, \text{ as } f = o(h),$$
$$\lim_{h\to 0} \frac{f(h)+g(h)}{h} = \lim_{h\to 0} \frac{f(h)}{h} + \lim_{h\to 0} \frac{g(h)}{h} = 0.$$

For (4), use the Algebraic Limit Theorem,

$$\begin{aligned}\lim_{h\to 0} \frac{f(h)g(h)}{h} &= \lim_{h\to 0} h \frac{f(h)}{h} \frac{g(h)}{h} \\ &= \lim_{h\to 0} h \lim_{h\to 0} \frac{f(h)}{h} \lim_{h\to 0} \frac{g(h)}{h} \\ &= 0 \cdot 0 \cdot 0 = 0.\end{aligned}$$

s.1.1.2. When $|x| \ll 1$, the terms with $n \ge 2$ in (1.1.1c) are $x^n = o(x)$. Then applying $x^n + x^m = o(x)$ to the Taylor series gives the result.

s.1.1.3. To see (1.1.2a), note first that $X\,\mathbb{1}_{X=n} = n\,\mathbb{1}_{X=n}$ because $X = n$ when $\mathbb{1}_{X=n} = 1$, and second that $\sum_{n=0}^{\infty} \mathbb{1}_{X=n} = 1$, since X takes one of the values in $\mathbb{N}$, and events $\{X = n\}$ and $\{X = m\}$ are non-overlapping when $n \ne m$.

s.1.1.4. This is just rewriting the definition:

$$G(k) = \mathsf{P}(X > k) = \sum_{m=k+1}^{\infty} \mathsf{P}(X = m) = \sum_{m=k+1}^{\infty} f(m) = \sum_{m=0}^{\infty} \mathbb{1}_{m>k} f(m).$$

s.1.1.5.

$$\begin{aligned}f(k) &= \mathsf{P}(X = k) = \mathsf{P}(X \le k) - \mathsf{P}(X \le k-1) = F(k) - F(k-1), \\ G(k) &= \mathsf{P}(X > k) = 1 - \mathsf{P}(X \le k) = 1 - F(k).\end{aligned}$$

s.1.1.6. $G(k) = 1 - F(k) = 1 - \mathsf{P}(X \le k) = \mathsf{P}(X > k)$. It is all too easy to make, so called, off-by-one errors, such as in the three alternatives above. I nearly always check simple cases to prevent such simple mistakes. I advise you to acquire the same habit.

s.1.1.7. Observe first that $\sum_{k=0}^{\infty} \mathbb{1}_{m>k} = m$, since $\mathbb{1}_{m>k} = 1$ if $k < m$ and $\mathbb{1}_{m>k} = 0$ if $k \ge m$. With this,

$$\begin{aligned}\sum_{k=0}^{\infty} G(k) &= \sum_{k=0}^{\infty} \mathsf{P}(X > k) = \sum_{k=0}^{\infty} \sum_{m=k+1}^{\infty} \mathsf{P}(X = m) \\ &= \sum_{k=0}^{\infty} \sum_{m=0}^{\infty} \mathbb{1}_{m>k} \mathsf{P}(X = m) = \sum_{m=0}^{\infty} \sum_{k=0}^{\infty} \mathbb{1}_{m>k} \mathsf{P}(X = m) \\ &= \sum_{m=0}^{\infty} m\mathsf{P}(X = m) = \mathsf{E}[X].\end{aligned}$$

In case you are interested in mathematical justifications: the interchange of the two summations is allowed by Tonelli's theorem because the summands are all positive. (Interchanging the order of summations or integration is not always allowed because the results can be different when part of the integrand is negative. Check Fubini's theorem for more on this if you are interested.)

s.1.1.8.

$$\begin{aligned}\sum_{i=0}^{\infty} iG(i) &= \sum_{i=0}^{\infty} i \sum_{n=i+1}^{\infty} \mathsf{P}(X=n) = \sum_{n=0}^{\infty} \mathsf{P}(X=n) \sum_{i=0}^{\infty} i\, \mathbb{1}_{n\geq i+1} \\ &= \sum_{n=0}^{\infty} \mathsf{P}(X=n) \sum_{i=0}^{n-1} i = \sum_{n=0}^{\infty} \mathsf{P}(X=n) \frac{(n-1)n}{2} \\ &= \sum_{n=0}^{\infty} \frac{n^2}{2} \mathsf{P}(X=n) - \frac{\mathsf{E}[X]}{2} = \frac{\mathsf{E}\left[X^2\right]}{2} - \frac{\mathsf{E}[X]}{2}.\end{aligned}$$

s.1.1.9.

$$\begin{aligned}\mathsf{E}[X] &= \int_0^{\infty} x\, \mathrm{d}F(x) = \int_0^{\infty} \int_0^{x} \mathrm{d}y\, \mathrm{d}F(x) \\ &= \int_0^{\infty} \int_0^{\infty} \mathbb{1}_{y\leq x}\, \mathrm{d}y\, \mathrm{d}F(x) = \int_0^{\infty} \int_0^{\infty} \mathbb{1}_{y\leq x}\, \mathrm{d}F(x)\, \mathrm{d}y \\ &= \int_0^{\infty} \int_y^{\infty} \mathrm{d}F(x)\, \mathrm{d}y = \int_0^{\infty} G(y)\, \mathrm{d}y.\end{aligned}$$

s.1.1.10.

$$\begin{aligned}\int_0^{\infty} yG(y)\, \mathrm{d}y &= \int_0^{\infty} y \int_y^{\infty} f(x)\, \mathrm{d}x\, \mathrm{d}y = \int_0^{\infty} y \int_0^{\infty} \mathbb{1}_{y\leq x} f(x)\, \mathrm{d}x\, \mathrm{d}y \\ &= \int_0^{\infty} f(x) \int_0^{\infty} y\, \mathbb{1}_{y\leq x}\, \mathrm{d}y\, \mathrm{d}x = \int_0^{\infty} f(x) \int_0^{x} y\, \mathrm{d}y\, \mathrm{d}x \\ &= \int_0^{\infty} f(x) \frac{x^2}{2}\, \mathrm{d}x = \frac{\mathsf{E}\left[X^2\right]}{2}.\end{aligned}$$

s.1.1.11. Use integration by parts.

$$\int_0^{\infty} yG(y)\, \mathrm{d}y = \frac{y^2}{2} G(y)\Big|_0^{\infty} - \int_0^{\infty} \frac{y^2}{2} g(y)\, \mathrm{d}y = \int_0^{\infty} \frac{y^2}{2} f(y)\, \mathrm{d}y = \frac{\mathsf{E}\left[X^2\right]}{2}, \tag{1.1.5}$$

since $g(y) = G'(y) = -F'(y) = -f(y)$. Note that we used $\frac{y^2}{2} G(y)\Big|_0^{\infty} = 0 - 0 = 0$, which follows from our assumption that $\mathsf{E}\left[X^2\right]$ exists, implying that $\lim_{y\to\infty} y^2 G(y) = 0$.

s.1.1.12. $M_X(0) = \mathsf{E}\left[e^{0X}\right] = \mathsf{E}\left[e^0\right] = \mathsf{E}[1] = 1$.

s.1.1.13. The probability that the first child to guess also wins is 1/3. What is the probability for child number two? Well, for him/her to win, it is necessary that child one does not win and that child two guesses the right number of the remaining numbers. Assume, without loss of generality that child 1 chooses 3 and that this is not the right number. Then

$$\begin{aligned}&\mathsf{P}(\text{Child 2 wins}) \\ &= \mathsf{P}(\text{Child 2 guesses the right number and child 1 does not win}) \\ &= \mathsf{P}(\text{Child 2 guesses the right number} \mid \text{ child 1 does not win}) \cdot \mathsf{P}(\text{Child 1 does not win}) \\ &= \mathsf{P}(\text{Child 2 makes the right guess in the set } \{1,2\}) \cdot \frac{2}{3} \\ &= \frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}.\end{aligned}$$

Similar conditional reasoning gives that child 3 wins with probability 1/3.

1.2 POISSON DISTRIBUTION

Theory and Exercises

In this section, we provide motivation for the use of the Poisson process as an arrival process of customers or jobs at a shop, service station, or machine to receive service. In the exercises we derive numerous properties of this exceedingly important distribution; in the rest of the book we will use these results time and again.

Consider a stream of customers that enter a shop over time. Let us write $N(t)$ for the number of customers that enter during the time interval $[0,t]$ and $N(s,t] = N(t) - N(s)$ for the number that arrive in the time period $(s,t]$. Clearly, as we do not know in advance how many customers will enter, we model the set $\{N(t), t \geq 0\}$ as a family of random variables.

Our first assumption is that the rate at which customers enter stays constant over time. Then it is reasonable to assume that the expected number of arrivals is proportional to the length of the interval. Hence, it is reasonable to assume that there exists some constant λ such that

$$\mathsf{E}[N(s,t]] = \lambda(t-s). \tag{1.2.1}$$

The constant λ is called the *arrival rate* of the arrival process.

The second assumption is that the process $N_\lambda = \{N(t), t \geq 0\}$ has *stationary and independent increments*. Stationarity means that the distributions of the number of arrivals are the same for all intervals of equal length, that is, $N(s,t]$ has the same distribution as $N(u,v]$ if $t-s = v-u$. Independence means, roughly speaking, that knowing that $N(s,t] = n$, does not help to make any predictions about the value of $N(u,v]$ if the intervals $(s,t]$ and $(u,v]$ do not overlap.

To find the distribution of $N(t)$ for some given t, let us split the interval $[0,t]$ into n sub-intervals, all of equal length, and ask: 'What is the probability that a customer arrives in some given sub-interval?' By our first assumption, the arrival rate is constant over time. Therefore, the probability p of an arrival in each interval should be constant. Moreover, if the time intervals are very small, we can safely neglect the probability that two or more customers arrive in one interval.

As a consequence, then, we can model the occurrence of an arrival in some period i as a Bernoulli distributed random variable B_i such that $p = \mathsf{P}(B_i = 1)$ and $\mathsf{P}(B_i = 0) = 1 - \mathsf{P}(B_i = 1)$, and we assume that B_i and B_j are independent whenever $i \neq j$. The total number of arrivals $N_n(t)$ that occur in n intervals is then *binomially distributed*, i.e.,

$$\mathsf{P}(N_n(t) = k) = \binom{n}{k} p^k (1-p)^{n-k}. \tag{1.2.2}$$

If we take $n \to \infty$, $p \to 0$ such that $np = \lambda t$, then $N_n(t)$ converges (in distribution) to a *Poisson distributed* random variable $N(t)$, i.e.,

$$\mathsf{P}(N(t) = k) = e^{-\lambda t} \frac{(\lambda t)^k}{k!}, \tag{1.2.3}$$

and then we write $N(t) \sim P(\lambda t)$.

1.2.1. *Show that* $\mathsf{E}[N_n(t)] = \sum_{i=1}^{n} \mathsf{E}[B_i] = np$. *Conclude that we need to choose* $p = \lambda t/n$ *if we want that* $\mathsf{E}[N_n(t)] = \mathsf{E}[N(t)]$.

1.2.2. *What is the difference between* $N_n(t)$ *and* $N(t)$*?*

1.2.3. *Show that the binomial distribution in* (1.2.2) *converges to the Poisson distribution, i.e.,*

$$\lim_{n\to\infty}\binom{n}{k}\left(\frac{\lambda t}{n}\right)^k\left(1-\frac{\lambda t}{n}\right)^{n-k}=e^{-\lambda t}\frac{(\lambda t)^k}{k!},$$

if $n\to\infty$, $p\to 0$ *such that* $np=\lambda t$.

We call the process $N_\lambda=\{N(t)\}$ a *Poisson process* with rate λ when N_λ is stationary, has independent increments, and its elements $N(t)\sim P(\lambda t)$ for all t. Observe that the process N_λ is a much more complicated object than a Poisson distributed random variable. The process is an uncountable set of random variables indexed by $t\in\mathbb{R}^+$, not just *one* random variable.

In the remainder of this section we derive a number of properties of the Poisson process that we will use time and again.

1.2.4. *Show that*

$$\mathsf{P}(N(t+h)=n\,|\,N(t)=n)=1-\lambda h+o(h)$$

when $N(t)\sim P(\lambda t)$ *and* h *is small.*

1.2.5. *Show that*

$$\mathsf{P}(N(t+h)=n+1\,|\,N(t)=n)=\lambda h+o(h)$$

when $N(t)\sim P(\lambda t)$ *and* h *is small.*

1.2.6. *Show that if* $N(t)\sim P(\lambda t)$, *we have for small* h,

$$\mathsf{P}(N(t+h)\geq n+2\,|\,N(t)=n)=o(h).$$

1.2.7. *Show that for a Poisson process* N_λ,

$$\mathsf{P}(N(s)=1\,|\,N(t)=1)=\frac{s}{t},$$

if $s\in[0,t]$. *Thus, if you know that an arrival occurred during* $[0,t]$, *the arrival is distributed uniformly on the interval* $[0,t]$. *Note that this probability is independent of* λ.

1.2.8. *Show that if* $N(t)\sim P(\lambda t)$, *the expected number of arrivals during* $[0,t]$ *is*

$$\mathsf{E}[N(t)]=\lambda t.$$

1.2.9. *Show that if* $N(t)\sim P(\lambda t)$, *the variance of the number of arrivals during* $[0,t]$ *is*

$$\mathsf{V}[N(t)]=\lambda t.$$

1.2.10. *Show that the moment-generating function of the random variable* $N(t)\sim P(\lambda t)$ *is*

$$M_{N(t)}(s)=\exp(\lambda t(e^s-1)).$$

1.2.11. *Use the moment-generating function of* $N(t)\sim P(\lambda t)$ *to compute* $\mathsf{E}[N(t)]$ *and* $\mathsf{V}[N(t)]$.

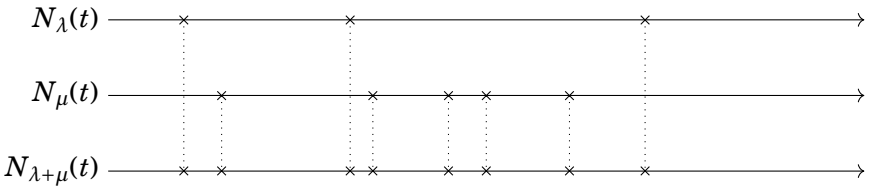
Define the *square coefficient of variation* (*SCV*) of a random variable X as

$$C^2=\frac{\mathsf{V}[X]}{(\mathsf{E}[X])^2}. \tag{1.2.4}$$

As will become clear later, the SCV is a very important concept in queueing theory. Memorize it as a measure of *relative variability*.

1.2.12. *Show that the SCV of $N(t) \sim P(\lambda t)$ is equal to $1/(\lambda t)$. What does this mean for t large?*

Merging Poisson processes occurs often in practice. We have two Poisson processes, for instance, the arrival processes N_λ of men and N_μ of women at a shop. In the figure below, each cross represents an arrival; in the upper line it corresponds to a man, in the middle line to a woman, and in the lower line to an arrival of a general customer at the shop. Thus, the shop 'sees' the superposition of these two arrival processes. In fact, this merged process $N_{\lambda+\mu}$ is also a Poisson process with rate $\lambda+\mu$.



1.2.13. *If the Poisson arrival processes N_λ and N_μ are independent, show with a conditioning argument that $N_\lambda + N_\mu$ is a Poisson process with rate $\lambda+\mu$.*

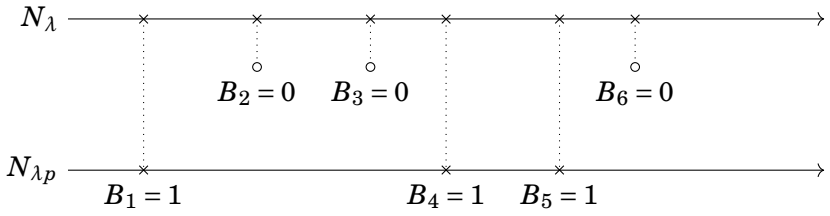
1.2.14. *If the Poisson arrival processes N_λ and N_μ are independent, use moment-generating functions to show that $N_\lambda + N_\mu$ is a Poisson process with rate $\lambda+\mu$.*

1.2.15. *If the Poisson arrival processes N_λ and N_μ are independent, show that*

$$\mathsf{P}\left(N_\lambda(h) = 1 | N_\lambda(h) + N_\mu(h) = 1\right) = \frac{\lambda}{\lambda+\mu}.$$

Note that the right-hand side does not depend on h, hence it holds for any time h, whether it is small or not.

Besides merging Poisson streams, we can also consider the concept of *splitting*, or *thinning*, a stream into sub-streams, as follows. Model the stream of people passing by a shop as a Poisson process N_λ. In the figure below these arrivals are marked as crosses at the upper line. With probability p a person decides, independent of anything else, to enter the shop; the crosses at the lower line are the customers that enter the shop. In the figure, the Bernoulli random variable $B_1 = 1$ so that the first passerby enters the shop; the second passerby does not enter as $B_2 = 0$, and so on.



1.2.16. *Show with conditioning that thinning the Poisson process N_λ by means of Bernoulli random variables with success probability p results in a Poisson process $N_{\lambda p}$.*

1.2.17. *Show with moment-generating functions that thinning the Poisson process N_λ by means of Bernoulli random variables with success probability p results in a Poisson process $N_{\lambda p}$.*

The concepts of merging and thinning are useful to analyze queueing networks. Suppose the departure stream of a machine splits into two sub-streams, e.g., a fraction p of the jobs moves on to another machine and the rest $(1-p)$ of the jobs leaves the system. Then we can model the arrival stream at the second machine as a thinned stream (with probability p) of the departures of the first machine. Merging occurs where the output streams of various stations arrive at another station.

1.2.18. *Use moment-generating functions to prove that $N_n(t)$ converges to N, that is, the right-hand side in* (1.2.2) *converges to the Poisson distribution* (1.2.3) *when $n \to \infty, p \to 0$ such that $pn = \lambda t$ remains constant.*

Hints

h.1.2.1. Use that $\mathsf{E}[X+Y] = \mathsf{E}[X] + \mathsf{E}[Y]$.

h.1.2.3. First find p, n, λ and t such that the rate at which an event occurs in both processes are the same. Then consider the binomial distribution and use the standard limit $(1-x/n)^n \to e^{-x}$ as $n \to \infty$.

h.1.2.4. Use the definition of the conditional probability and small o notation.

h.1.2.5. Use 1.2.4.

h.1.2.6. Use that $\sum_{i=2}^{\infty} x^i/i! = \sum_{i=0}^{\infty} x^i/i! - x - 1 = e^x - x - 1$.

h.1.2.7. Observe that

$$\mathbb{1}_{N(0,s]+N(s,t]=1}\,\mathbb{1}_{N(0,s]=1} = \mathbb{1}_{1+N(s,t]=1}\,\mathbb{1}_{N(0,s]=1} = \mathbb{1}_{N(s,t]=0}\,\mathbb{1}_{N(0,s]=1}.$$

Use independence and (1.1.2d).

h.1.2.8. Use (1.1.2c). Note that the term with $n = 0$ does not contribute in the following summation

$$\sum_{n=0}^{\infty} n\frac{\lambda^n}{n!} = \sum_{n=1}^{\infty} n\frac{\lambda^n}{n!} = \sum_{n=1}^{\infty} \frac{\lambda^n}{(n-1)!} = \lambda \sum_{n=0}^{\infty} \frac{\lambda^n}{n!} = \lambda e^{\lambda},$$

where we apply a change of notation in the second to last step.

h.1.2.9. Use (1.1.2e). Compute $\mathsf{E}\left[N^2\right]$ and use 1.2.8.

h.1.2.10. Use (1.1.2c) with $f(x) = e^{sx}$.

h.1.2.11. Use (1.1.3c) and (1.1.3d).

h.1.2.13. Use sets $\{N_\lambda(t) = i\}$ to decompose $\{N_\lambda(t) + N_\mu(t) = n\}$. With this observe that

$$\mathbb{1}_{N_\lambda(t)+N_\mu(t)=n} = \sum_{i=0}^{n} \mathbb{1}_{N_\lambda(t)=i, N_\mu(t)=n-i}.$$

Take expectations left and right, use (1.1.2d), and independence of N_λ and N_μ. Near the end of the computation, use (1.1.1a).

h.1.2.14. Use (1.1.3e) and (1.1.3b).

h.1.2.15. Use the standard formula for conditional probability and that $N_\lambda(t)+N_\mu(t) \sim \mathrm{P}((\lambda+\mu)t)$. Interpret the result.

h.1.2.16. Suppose that N_1 is the thinned stream, and N the original stream. Condition on the total number of arrivals $N(t)=n$ up to time t. Then, realize that the probability that a person is of type 1 is p. Hence when you consider n people in total, the number $N_1(t)$ of type 1 people is binomially distributed. Thus, given that n people arrived, the probability of k 'successes' (i.e., arrivals of type 1), is

$$\mathsf{P}(N_1(t)=k \,|\, N(t)=n) = \binom{n}{k} p^k (1-p)^{n-k}.$$

Use (1.1.4b) to decompose the $\{N_1=k\}$, and (1.1.1c) at the end.

h.1.2.17. Dropping the dependence of N on t for the moment for notational convenience, consider the random variable

$$Y = \sum_{i=1}^{N} Z_i,$$

with $N \sim P(\lambda)$ and $Z_i \sim B(p)$. Show that the moment-generating function of Y is equal to the moment-generating function of a Poisson random variable with parameter λp.

h.1.2.18. Solve 1.2.10 and 1.2.17 first. Perhaps 1.2.3 is also useful.

Solutions

s.1.2.1.

$$\mathsf{E}[N_n(t)] = \mathsf{E}\left[\sum_{i=1}^{n} B_i\right] = \sum_{i=1}^{n} \mathsf{E}[B_i] = n\,\mathsf{E}[B_i] = np.$$

By (1.2.1) $\mathsf{E}[N(t)] = \lambda t$. Now we want the expectations of $N_n(t)$ and $N(t)$ to be the same, thus, $np = \lambda t$.

s.1.2.2. $N_n(t)$ is a binomially distributed random variable with parameters n and p. The maximum value of $N_n(t)$ is n. The random variable $N(t)$ models the number of arrivals that can occur during $[0,t]$. As such it is not necessarily bounded by n. Thus, $N_n(t)$ and $N(t)$ cannot represent the same random variable.

s.1.2.3.

$$\begin{aligned}
\binom{n}{k}\left(\frac{\lambda t}{n}\right)^k\left(1-\frac{\lambda t}{n}\right)^{n-k} &= \frac{n!}{k!(n-k)!}\left(\frac{\lambda t}{n}\frac{n}{n-\lambda t}\right)^k\left(1-\frac{\lambda t}{n}\right)^n \\
&= \frac{(\lambda t)^k}{k!}\left(\frac{n}{n-\lambda t}\right)^k \frac{n!}{n^k(n-k)!}\left(1-\frac{\lambda t}{n}\right)^n \\
&= \frac{(\lambda t)^k}{k!}\left(\frac{n}{n-\lambda t}\right)^k \frac{n}{n}\frac{n-1}{n}\cdots\frac{n-k+1}{n}\left(1-\frac{\lambda t}{n}\right)^n.
\end{aligned}$$

Observe now that, as λt is finite, $n/(n-\lambda t) \to 1$ as $n \to \infty$. Also for any finite k, $(n-k)/n \to 1$. Finally, use (1.1.1b) to see that $\left(1-\frac{\lambda t}{n}\right)^n \to e^{-\lambda t}$.

s.1.2.4. Write $N(s,t]$ for the number of arrivals in the interval $(s,t]$. First we make a few simple observations: $N(t+h] = N(t) + N(t,t+h]$, hence

$$\mathbb{1}_{N(t+h)=n,N(t)=n} = \mathbb{1}_{N(t)+N(t,t+h]=n,N(t)=n} = \mathbb{1}_{N(t,t+h]=0,N(t)=n}.$$

Thus,

$$\begin{aligned}
\mathsf{P}(N(t+h)=n|N(t)=n) &= \frac{\mathsf{P}(N(t+h)=n,N(t)=n)}{\mathsf{P}(N(t)=n)} \\
&= \frac{\mathsf{P}(N(t,t+h]=0,N(t)=n)}{\mathsf{P}(N(t)=n)} \\
&= \frac{\mathsf{P}(N(t,t+h]=0)\mathsf{P}(N(t)=n)}{\mathsf{P}(N(t)=n)} \quad \text{(independence)} \\
&= \mathsf{P}(N(t,t+h]=0) \\
&= \mathsf{P}(N(0,h]=0) \quad \text{(stationarity)} \\
&= e^{-\lambda h}(\lambda h)^0/0! \\
&= e^{-\lambda h} = 1-\lambda h + o(h).
\end{aligned}$$

s.1.2.5.

$$\begin{aligned}
\mathsf{P}(N(t+h)=n+1|N(t)=n) &= \mathsf{P}(N(t+h)=n+1,N(t)=n)/\mathsf{P}(N(t)=n) \\
&= \mathsf{P}(N(t,t+h]=1) = e^{-\lambda h}(\lambda h)^1/1! \\
&= (1-\lambda h+o(h))\lambda h = \lambda h - \lambda^2 h^2 + o(h) \\
&= \lambda h + o(h).
\end{aligned}$$

s.1.2.6.

$$\begin{aligned}
\mathsf{P}(N(t+h)\geq n+2|N(t)=n) &= \mathsf{P}(N(t,t+h]\geq 2) \\
&= e^{-\lambda h}\sum_{i=2}^{\infty}\frac{(\lambda h)^i}{i!} = e^{-\lambda h}\left(\sum_{i=0}^{\infty}\frac{(\lambda h)^i}{i!} - \lambda h - 1\right) \\
&= e^{-\lambda h}(e^{\lambda h} - 1 - \lambda h) = 1 - e^{-\lambda h}(1+\lambda h) \\
&= 1-(1-\lambda h+o(h))(1+\lambda h) = 1-(1-\lambda^2 h^2 + o(h)) \\
&= \lambda^2 h^2 + o(h) = o(h).
\end{aligned}$$

We can also use the results of the previous parts to see that

$$\begin{aligned}
\mathsf{P}(N(t+h)\geq n+2|N(t)=n) &= \mathsf{P}(N(t,t+h]\geq 2) = 1-\mathsf{P}(N(t,t+h]<2) \\
&= 1-\mathsf{P}(N(t,t+h]=0)-\mathsf{P}(N(t,t+h]=1) \\
&= 1-(1-\lambda h+o(h))-(\lambda h+o(h)) \\
&= o(h).
\end{aligned}$$

s.1.2.7. From the hint,

$$\begin{aligned}
\mathsf{P}(N(0,s]=1\,|\,N(0,t]=1) &= \frac{\mathsf{P}(N(0,s]=1,N(0,t]=1)}{\mathsf{P}(N(0,t]=1)} \\
&= \frac{\mathsf{P}(N(0,s]=1,N(s,t]=0)}{\mathsf{P}(N(0,t]=1)}
\end{aligned}$$

$$= \frac{\mathsf{P}(N(0,s]=1)\mathsf{P}(N(s,t]=0)}{\mathsf{P}(N(0,t]=1)}$$
$$= \frac{\lambda s e^{-\lambda s} e^{-\lambda(t-s)}}{\lambda t e^{-\lambda t}} = \frac{s}{t}.$$

s.1.2.8. When a random variable N is Poisson distributed with parameter λt,

$$\begin{aligned}
\mathsf{E}[N] &= \sum_{n=0}^{\infty} n e^{-\lambda t} \frac{(\lambda t)^n}{n!} \\
&= \sum_{n=1}^{\infty} n e^{-\lambda t} \frac{(\lambda t)^n}{n!} \\
&= e^{-\lambda t} \lambda t \sum_{n=1}^{\infty} \frac{(\lambda t)^{n-1}}{(n-1)!} \\
&= e^{-\lambda t} \lambda t \sum_{n=0}^{\infty} \frac{(\lambda t)^n}{n!} \\
&= e^{-\lambda t} \lambda t e^{\lambda t} \\
&= \lambda t.
\end{aligned}$$

s.1.2.9.

$$\begin{aligned}
\mathsf{E}\left[N^2\right] &= \sum_{n=0}^{\infty} n^2 e^{-\lambda t} \frac{(\lambda t)^n}{n!} \\
&= e^{-\lambda t} \sum_{n=1}^{\infty} n \frac{(\lambda t)^n}{(n-1)!} \\
&= e^{-\lambda t} \sum_{n=0}^{\infty} (n+1) \frac{(\lambda t)^{n+1}}{n!} \\
&= e^{-\lambda t} \lambda t \sum_{n=0}^{\infty} n \frac{(\lambda t)^n}{n!} + e^{-\lambda t} \lambda t \sum_{n=0}^{\infty} \frac{(\lambda t)^n}{n!} \\
&\quad \text{now use the hint to simplify the first summation} \\
&= (\lambda t)^2 + \lambda t.
\end{aligned}$$

Hence, $\mathsf{V}[N] = \mathsf{E}\left[N^2\right] - (\mathsf{E}[N])^2 = (\lambda t)^2 + \lambda t - (\lambda t)^2 = \lambda t$.

s.1.2.10. Since $N(t)$ is Poisson distributed with parameter λt,

$$\begin{aligned}
M_{N(t)}(s) &= \mathsf{E}\left[e^{sN(t)}\right] \\
&= \sum_{k=0}^{\infty} e^{sk} \mathsf{P}(N(t)=k) \\
&= \sum_{k=0}^{\infty} e^{sk} \frac{(\lambda t)^k}{k!} e^{-\lambda t} \\
&= e^{-\lambda t} \sum_{k=0}^{\infty} \frac{(e^s \lambda t)^k}{k!} \\
&= \exp(-\lambda t + e^s \lambda t) = \exp(\lambda t (e^s - 1)).
\end{aligned}$$

s.1.2.11. Using the expression for the moment-generating function of 1.2.10,

$$M'_{N(t)}(s) = \lambda t e^s \exp(\lambda t (e^s - 1)).$$

Hence $\mathsf{E}[N(t)] = M'_{N(t)}(0) = \lambda t$. Next,

$$M''_{N(t)}(s) = (\lambda t e^s + (\lambda t e^s)^2)\exp(\lambda t(e^s - 1)),$$

hence $\mathsf{E}\left[(N(t))^2\right] = M''(0) = \lambda t + (\lambda t)^2$. And thus,

$$\mathsf{V}[N(t)] = \mathsf{E}\left[(N(t))^2\right] - (\mathsf{E}[N(t)])^2 = \lambda t + (\lambda t)^2 - (\lambda t)^2 = \lambda t.$$

s.1.2.12.

$$SCV = \frac{\mathsf{V}[N(t)]}{(\mathsf{E}[N(t)])^2} = \frac{\lambda t}{(\lambda t)^2} = \frac{1}{\lambda t}.$$

The relative variability of the Poisson process goes down as $t \to \infty$.

s.1.2.13.

$$\begin{aligned}
\mathsf{P}\left(N_\lambda(t) + N_\mu(t) = n\right) &= \sum_{i=0}^{n} \mathsf{P}\left(N_\mu(t) = n - i\right)\mathsf{P}\left(N_\lambda(t) = i\right) \\
&= \sum_{i=0}^{n} \frac{(\mu t)^{n-i}}{(n-i)!}\frac{(\lambda t)^i}{i!}e^{-(\mu+\lambda)t} \\
&= e^{-(\mu+\lambda)t}\sum_{i=0}^{n} \frac{(\mu t)^{n-i}}{(n-i)!}\frac{(\lambda t)^i}{i!} \\
&= e^{-(\mu+\lambda)t}\frac{1}{n!}\sum_{i=0}^{n}\binom{n}{i}(\mu t)^{n-i}(\lambda t)^i \qquad \text{(binomial formula)} \\
&= \frac{((\mu+\lambda)t)^n}{n!}e^{-(\mu+\lambda)t}.
\end{aligned}$$

s.1.2.14.

$$\begin{aligned}
M_{N_\lambda(t)+N_\mu(t)}(s) &= M_{N_\lambda(t)}(s)\cdot M_{N_\mu(t)}(s) \\
&= \exp(\lambda t(e^s - 1))\cdot\exp(\mu t(e^s - 1)) \\
&= \exp((\lambda+\mu)t(e^s - 1)).
\end{aligned}$$

The last expression is the moment-generating function of a Poisson random variable with parameter $(\lambda + \mu)t$.

s.1.2.15. With the above:

$$\begin{aligned}
&\mathsf{P}\left(N_\lambda(h) = 1 | N_\lambda(h) + N_\mu(h) = 1\right) \\
&= \frac{\mathsf{P}\left(N_\lambda(h) = 1, N_\lambda(h) + N_\mu(h) = 1\right)}{\mathsf{P}\left(N_\lambda(h) + N_\mu(h) = 1\right)} \\
&= \frac{\mathsf{P}\left(N_\lambda(h) = 1, N_\mu(h) = 0\right)}{\mathsf{P}\left(N_{\lambda+\mu}(h) = 1\right)} \\
&= \frac{\mathsf{P}\left(N_\lambda(h) = 1\right)\mathsf{P}\left(N_\mu(h) = 0\right)}{\mathsf{P}\left(N_{\lambda+\mu}(h) = 1\right)} \\
&= \frac{\lambda h\exp(-\lambda h)\exp(-\mu h)}{((\lambda+\mu)h)\exp(-(\lambda+\mu)h)} \\
&= \frac{\lambda h\exp(-(\lambda+\mu)h)}{((\lambda+\mu)h)\exp(-(\lambda+\mu)h)} \\
&= \frac{\lambda}{\lambda+\mu}.
\end{aligned}$$

Given that a customer arrived in $[0, t]$, the probability that it is of the first type is $\lambda/(\lambda+\mu)$.

s.1.2.16.

$$\begin{aligned}
\mathsf{P}(N_1(t)=k) &= \sum_{n=k}^{\infty} \mathsf{P}(N_1(t)=k, N(t)=n) = \sum_{n=k}^{\infty} \mathsf{P}(N_1(t)=k \mid N(t)=n)\mathsf{P}(N(t)=n) \\
&= \sum_{n=k}^{\infty} \mathsf{P}(N_1(t)=k \mid N(t)=n) e^{-\lambda t}\frac{(\lambda t)^n}{n!} \\
&= \sum_{n=k}^{\infty} \binom{n}{k} p^k (1-p)^{n-k} e^{-\lambda t}\frac{(\lambda t)^n}{n!}, \quad \text{by the hint} \\
&= e^{-\lambda t}\sum_{n=k}^{\infty} \frac{p^k(1-p)^{n-k}}{k!(n-k)!}(\lambda t)^n = e^{-\lambda t}\frac{(\lambda t p)^k}{k!}\sum_{n=k}^{\infty}\frac{(\lambda t(1-p))^{n-k}}{(n-k)!} \\
&= e^{-\lambda t}\frac{(\lambda t p)^k}{k!}\sum_{n=0}^{\infty}\frac{(\lambda t(1-p))^{n}}{n!} = e^{-\lambda t}\frac{(\lambda t p)^k}{k!}e^{\lambda t(1-p)} \\
&= e^{-\lambda t p}\frac{(\lambda t p)^k}{k!}.
\end{aligned}$$

s.1.2.17. Consider $Y = \sum_{i=1}^{N} Z_i$. Suppose that $N = n$, so that n arrivals occurred. Then we throw n independent coins with success probability p. It is clear that Y is indeed a thinned Poisson random variable.

Model the coins as a generic Bernoulli distributed random variable Z. We first need

$$\mathsf{E}\left[e^{sZ}\right] = e^0\mathsf{P}(Z=0) + e^s\mathsf{P}(Z=1) = (1-p) + e^s p.$$

Suppose that $N = n$, then since the outcomes Z_i of the coins are i.i.d.,

$$\mathsf{E}\left[e^{s\sum_{i=1}^{n} Z_i}\right] = \left(\mathsf{E}\left[e^{sZ}\right]\right)^n = \left(1 + p(e^s - 1)\right)^n,$$

where we use (1.1.3e).

With (1.1.2a),

$$\begin{aligned}
\mathsf{E}\left[e^{sY}\right] &= \mathsf{E}\left[\sum_{n=0}^{\infty} e^{s\sum_{i=1}^{N} Z_i}\mathbb{1}_{N=n}\right] \\
&= \mathsf{E}\left[\sum_{n=0}^{\infty} e^{s\sum_{i=1}^{n} Z_i}\mathbb{1}_{N=n}\right] \\
&= \sum_{n=0}^{\infty}\mathsf{E}\left[e^{s\sum_{i=1}^{n} Z_i}\mathbb{1}_{N=n}\right] \\
&= \sum_{n=0}^{\infty}\mathsf{E}\left[e^{s\sum_{i=1}^{n} Z_i}\right]\mathsf{E}[\mathbb{1}_{N=n}], \text{ by independence of } Z_i \text{ and } N, \\
&= \sum_{n=0}^{\infty}\left(1 + p(e^s-1)\right)^n \mathsf{P}(N=n) \\
&= \sum_{n=0}^{\infty}\left(1 + p(e^s-1)\right)^n e^{-\lambda}\frac{\lambda^n}{n!} = e^{-\lambda}\sum_{n=0}^{\infty}\frac{(1+p(e^s-1))^n\lambda^n}{n!} \\
&= e^{-\lambda}\exp(\lambda(1+p(e^s-1))) = \exp(\lambda p(e^s-1)).
\end{aligned}$$

s.1.2.18. Take $Y = \sum_{i=1}^{n} Z_i$ with $Z_i \sim B(p)$. Then,

$$M_Y(s) = \mathsf{E}\left[e^{s\sum_{i=1}^{n} Z_i}\right] = \left(\mathsf{E}\left[e^{sZ}\right]\right)^n = \left(1 + p(e^s-1)\right)^n.$$

Recall that $p = \lambda t/n$. Then, with (1.1.1b),

$$\lim_{n\to\infty}\left(1+\frac{\lambda t}{n}(e^s-1)\right)^n = \exp(\lambda t(e^s-1)).$$

Thus, the limit becomes equal to the moment-generating function of the Poisson distribution.

1.3 QUEUEING PROCESSES IN DISCRETE-TIME

We start with a description of a case to provide motivation to study queueing systems. Then we develop a set of recursions of fundamental importance to construct and simulate queueing systems. With these recursions, we analyze the efficacy of several suggestions to improve the case system. To illustrate the power of this approach, we close the section with a large number of exercises in which you develop recursions for many different queueing situations.

Case

At a mental health department, five psychiatrists do intakes of future patients to determine the best treatment process for the patients. There are complaints about the time patients have to wait for their first intake; the desired waiting time is around two weeks, but the realized waiting time is sometimes more than three months. The organization considers this to be unacceptably long, but... what to do about it?

To reduce the waiting times, the five psychiatrists have various suggestions.

1. Not all psychiatrists have the same amount of time available per week to do intakes. This is not a problem during weeks when all psychiatrists are present; however, psychiatrists tend to take holidays, visit conferences, and so on. So, if the psychiatrist with the most intakes per week would go on leave, this might affect the behavior of the queue length considerably. This raises the question about the difference in the allocation of capacity allotted to the psychiatrists. What are the consequences on the distribution and average of the waiting times if they would all have the same weekly capacity?

2. The psychiatrists tend to plan their holidays consecutively, to reduce the variation in the service capacity. What if they would synchronize their holidays, to the extent possible, rather than spread their holidays?

3. Finally, suppose the psychiatrists would do 2 more intakes per week in busy times and 2 fewer in quiet weeks. Assuming that the system is stable, i.e., the average service capacity is larger than the average demand, then on average the psychiatrists would not do more intakes, i.e., their workload would not increase, but the queue length may be controlled better.

As this case is too hard to analyze by mathematical means, we need to develop a model to simulate the queueing system in discrete time. With this simulator we can evaluate the effect of these suggestions on reducing the queueing dynamics. Interestingly, the structure of the simulation is very simple, so simple that it is also an exceedingly convincing tool to communicate the results of an analysis of a queueing system to managers (and the like).

Recursions

Let us start with discussing the essentials of the simulation of a queueing system. The easiest way to construct queueing processes is to 'chop up' time in periods and develop recursions for the behavior of the queue from period to period. Using fixed-sized periods has the advantage that we do not have to specify specific inter-arrival times or service times of individual customers; only the number of arrivals in a period and the number of potential services are relevant. Note that the length of such a period depends on the context for which the model is developed. For instance, to study queueing processes at a supermarket, a period can consist of 5 minutes, while for a production environment, e.g., a job shop, it can be a day, or even a week.

Let us define:

$a_k =$ number of jobs that arrive *in* period k,

$c_k =$ the capacity, i.e., the maximal number of jobs that can be served, during period k,

$d_k =$ number of jobs that depart *in* period k,

$L_k =$ number of jobs in the system at the *end* of period k.

In the sequel we also call a_k the *size of the batch* arriving in period k. Note that the definition of a_k is a bit subtle: we may assume that the arriving jobs arrive either at the start or at the end of the period. In the first case, the jobs can be served in period k, in the latter case, they *cannot* be served in period k.

Since L_{k-1} is the queue length at the *end* of period $k-1$, it must also be the number of customers at the *start* of period k. Assuming that jobs arriving in period k cannot be served in period k, the number of customers that depart in period k is

$$d_k = \min\{L_{k-1}, c_k\}, \tag{1.3.1a}$$

since only the jobs that are present at the start of the period, i.e., L_{k-1}, can be served if the capacity exceeds the queue length. Now that we know the number of departures, the number at the end of period k is given by

$$L_k = L_{k-1} - d_k + a_k. \tag{1.3.1b}$$

Like this, if we are given L_0, we can obtain L_1, and from this L_2, and so on.

Note that in this type of queueing system there is not a job in service, we only count the jobs in the system at the end of a period. Thus, the number of jobs in the system and in queue coincide in this model.

1.3.1. *Suppose that $c_k = 7$ for all k, and $a_1 = 5$, $a_2 = 4$ and $a_3 = 9$; also $L_0 = 8$. What are d_k and L_k for $k \geq 1$?*

1.3.2. *What is the interpretation of setting $d_k = \min\{L_{k-1}+a_k, c_k\}$ rather than definition* (1.3.1a)*?*

1.3.3. *Show that the scheme*

$$\begin{aligned} L_k &= [L_{k-1} + a_k - c_k]^+, \\ d_k &= L_{k-1} + a_k - L_k. \end{aligned} \tag{1.3.2}$$

is equivalent to a modification of (1.3.1) *in which we assume that jobs can be served in the period they arrive.*

Of course we are not going to carry out these computations by hand. Typically we use company data to model the arrival process $\{a_k\}_{k=1,2,\ldots}$ and the capacity $\{c_k\}_{k=1,2,\ldots}$ and feed this data into a computer to carry out the recursions (1.3.1). If we do not have sufficient data, we make a probability model for these data and use the computer to generate random numbers with, hopefully, similar characteristics as the real data. At any rate, from this point on, we assume that it is easy, by means of computers, to obtain numbers $a_1,\ldots,a_n$ for $n \gg 1000$, and so on.

Case analysis

Here we continue with the case of the five psychiatrists and analyze the proposed rules to improve the performance of the system. We mainly want to reduce the long waiting times.

As a first step in the analysis, we model the arrival process of patients as a Poisson process, cf. Section 1.2. The duration of a period is taken to be a week. The average number of arrivals per period, based on data of the company, was slightly less than 12 per week; in the simulation we set it to $\lambda = 11.8$ per week. We model the capacity in the form of a matrix such that row i corresponds to the weekly capacity of psychiatrist i:

$$C = \begin{pmatrix} 1 & 1 & 1 & \ldots \\ 1 & 1 & 1 & \ldots \\ 1 & 1 & 1 & \ldots \\ 3 & 3 & 3 & \ldots \\ 9 & 9 & 9 & \ldots \end{pmatrix}.$$

Thus, psychiatrists 1, 2, and 3 do just one intake per week, the fourth does 3, and the fifth does 9 intakes per week. The sum over column k is the total service capacity for week k of all psychiatrists together.

With the matrix C it is simple to make other capacity schemes. A more balanced scheme would be like this:

$$C = \begin{pmatrix} 2 & 2 & 2 & \ldots \\ 2 & 2 & 2 & \ldots \\ 3 & 3 & 3 & \ldots \\ 4 & 4 & 4 & \ldots \\ 4 & 4 & 4 & \ldots \end{pmatrix}.$$

Next, we include the effects of holidays on the capacity. This is easily done by setting the capacity of a certain psychiatrist to 0 in a certain week. Let us assume that just one psychiatrist is on leave in a week, each psychiatrist has one week per five weeks off, and the psychiatrists' holiday schemes rotate. To model this, we set $C_{1,1} = C_{2,2} = \cdots = C_{1,6} = C_{2,7} = \cdots = 0$, i.e.,

$$C = \begin{pmatrix} 0 & 2 & 2 & 2 & 2 & 0 & \ldots \\ 2 & 0 & 2 & 2 & 2 & 2 & \ldots \\ 3 & 3 & 0 & 3 & 3 & 3 & \ldots \\ 4 & 4 & 4 & 0 & 4 & 4 & \ldots \\ 4 & 4 & 4 & 4 & 0 & 4 & \ldots \end{pmatrix}.$$

Hence, the total average capacity must be $4/5 \cdot (2+2+3+4+4) = 12$ patients per week. The other holiday scheme—all psychiatrists take holiday in the same week–corresponds to setting

entire columns to zero, i.e., $C_{i,5} = C_{i,10} = \cdots = 0$ for week 5, 10, and so on. Note that all these variations in holiday schemes result in the same average capacity.

Now that we have modeled the arrivals and the capacities, we can use the recursions (1.3.1) to simulate the queue length process for the four different scenarios proposed by the psychiatrists, unbalanced versus balanced capacity, and spread out holidays versus simultaneous holidays.

The results are shown in Fig. 1. It is apparent that Suggestions 1 and 2 above do not significantly affect the behavior of the queue length process.

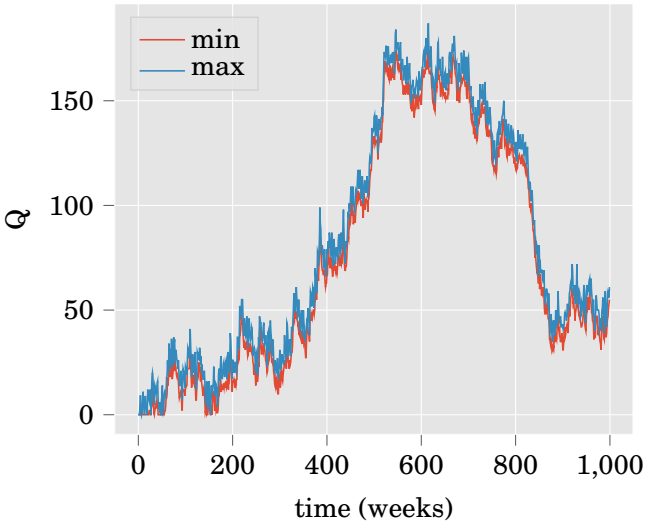


Figure 1: Effect of capacity and holiday plans. We plot for each time point the maximum and the minimum queue length for each of the policies. Apparently, the effect of each of these policies is, for all practical purposes, negligible.

Now we consider Suggestion 3, which comes down to doing more intakes when it is busy, and fewer when it is quiet. A simple rule would be to use week's queue L_{n-1}: if $L_{n-1} < 12$, serve e intakes less (in other, when the service capacity of week k is larger than the queue length serve e less), when $L_{n-1} > 24$, serve e more. Here, $e = 1$ or 2, or perhaps a larger number. The larger e, the larger the control we exercise.

Let's consider three different control levels, $e = 1$, $e = 2$, and $e = 5$; thus in the last case all psychiatrists do five extra intakes. The previous simulation shows that it is safe to disregard the holiday plans, so just assume a flat service capacity of 12 intakes a week. The results, see Fig. 2, show a striking difference indeed. The queue does not explode anymore, and already taking $e = 1$ has a large influence.

From this simulation experiment, we learn that changing holiday plans or spreading the work over multiple servers, i.e., psychiatrists, does not significantly affect the queueing behavior. However, controlling the service rate as a function of the queue length improves the situation quite dramatically.

Conclusion

From the above case we learn that even with these (deceitfully) simple recursions, we can obtain considerable insight into the behavior and performance of this quite complicated controlled

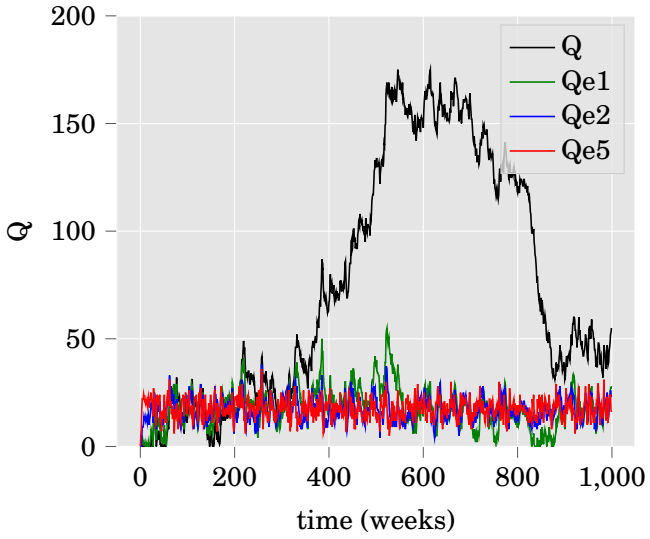


Figure 2: Controlling the number of intakes. Clearly, adapting the service rate 'does wonders' to control the queue length.

queueing system[2]. In general, the study of queueing systems is focused on studying the probabilistic properties of the queueing length process and related concepts such as waiting time, server occupancy, fraction of customers lost, and so on. Once we have constructed the queueing process, we can compute all relevant performance measures, such as the average waiting time. If it turns out that the performance of the system is not according to what we desire, we can change parts of the system with the aim to improve the situation and assess the effect of this change. For instance, if the average waiting time is too long, we might want to add service capacity. With simulation it is easy to study, hence evaluate, the effect of such decisions.

Exercises

The reader should understand from the above case that, once we have the recursions, we can analyze the system and make, for instance, plots to evaluate suggestions for improvement. Thus, formulating the recursions is crucial to construct queueing processes. For this reason, many of the exercises below concentrate on obtaining recursions for specific queueing systems.

It may be that the recursions you find are not identical to the recursions in the solution. The reason is that the assumptions you make might not be equal to the ones I make. I don't quite know how to get out of this paradoxical situation. In a sense, to completely specify the model, we need the recursions. However, if the problem statement would contain the recursions, there would be nothing left to practice anymore. Another way is to make the problem description five times as long, but this is also undesirable. So, let's be pragmatic: the aim is that you practice with modeling, and that you learn from the solutions. If you obtain *reasonable* recursions, but they are different from mine, then your answer is just as good.

1.3.4. *[Queue with Blocking] A queueing system under daily review, i.e., at the end of the day the queue length is measured. We assume that at the end of the day no jobs are still in service.*

[2] If you doubt the value of simulation, try to develop a mathematical method to analyze multi-server queueing systems with vacations, of which this case is an example.

We assume that jobs that arrive at day k cannot be served in day k. The queue length cannot exceed level K. Formulate a set of recursions to cover this case. What is the loss per period? What is the fraction of jobs lost?

1.3.5. *[Estimating the lead time distribution] Take $d_k = \min\{L_{k-1} + a_k, c_k\}$, and assume that jobs are served in FIFO sequence. Find an expression for the shortest possible waiting time $W_{-,k}$ of a job that arrives at time k, and an expression for the largest possible waiting time $W_{+,k}$.*

1.3.6. *[Yield loss] A machine produces items to serve customer demand. A fraction p of the items produced in a period turns out to be faulty, and has to be made anew. Develop a set of recursions to model the queue of customers waiting to be served from the machine.*

1.3.7. *[Rework]*

A machine produces items, but a fraction p of the items does not meet the quality requirements after the first pass at the server, but it requires a second pass. Assume that the repair of a faulty item requires half of the work of a new job, and that the faulty jobs are processed with priority over the new jobs. Also assume that faulty items do not need more than one repair (hence, faulty items that are repaired cannot be faulty anymore). Develop a set of recursions to model this case.

We remark that there are of course many different policies to treat rework. One possibility is to collect faulty items until there are N items, and then repair the entire batch of N items.

1.3.8. *[Cost models] A single-server queueing station processes customers. At the start of a period the server capacity is chosen, so that for period k the capacity is c_k. Demand that arrives in a period can be served in that period. It costs β per unit time per unit processing capacity to operate the machine, i.e., to have it switched on. There is also a cost h per unit time per job in the system. Make a cost model to analyze the long-run average cost for this case.*

1.3.9. *[N-policies] A machine can switch on and off. If the queue length hits N, the machine switches on, and if the system becomes empty, the machine switches off. It costs K to switch on the machine. There is also a cost β per unit time while the machine is switched on, and it costs h per unit time per customer in the system. Make a cost model.*

1.3.10. *[Queue length dependent server capacity] How would you model (in terms of recursions) a server whose capacity depends on the queue length? Consider, as an example, a rule such that the server only works if the queue is larger than a threshold t.*

1.3.11. *[Fair queueing] One server serves two queues. Each queue receives service capacity in proportion to the queue length. Derive a set of recursions to analyze this situation.*

1.3.12. *[Priority queueing] An interesting situation is a system with two queues served by one server, but such that one queue, queue A, gets priority over the other queue. Again find a set of recursions to describe this case.*

1.3.13. *[Queues with reserved service capacity] Consider a single-server that serves two parallel queues A and B. Each queue has a minimal service capacity every period, r_A for queue A and r_B for queue B. Reserved capacity unused for one queue can be used to serve the other queue. Any extra capacity beyond the reserved capacity is given to queue A with priority. Formulate a set of recursions to analyze this situation.*

1.3.14. *[Queue with protected service capacity and lost capacity] Consider a single-server that serves two parallel queues A and B. Each queue receives a minimal service capacity every period. Reserved capacity unused for one queue cannot be used to serve the other queue. Any extra capacity beyond the reserved capacity is given to queue A with priority. Formulate a set of recursions to analyze this situation.*

Let r_A be the reserved capacity for queue A, and likewise for r_B. We assume of course that $c_k \geq r_A + r_B$, for all k.

1.3.15. *[Tandem networks] Consider a production network with two production stations in tandem, that is, the jobs processed by station A are in the next period to the downstream Station B. Extend the recursions of* (1.3.1) *to simulate this situation.*

1.3.16. *[A tandem queue with blocking] Consider a production network with two production stations in tandem with blocking: when the intermediate queue, i.e., the queue in front of Station B, exceeds some level M, then station A has to stop producing, and when $L_k^B < M$ station A is not allowed to produce more than the intermediate queue can contain. Extend the recursions of* (1.3.1) *to simulate this situation.*

1.3.17. *[Merging departure streams] Consider another production situation with two machines, A and B say, that send their products to Station C. Derive a set of recursion relations to simulate this system.*

1.3.18. *[Merging incoming streams] Consider a single-server queue that serves two customer 'streams' in a FIFO discipline. Thus, both streams enter one queue that is served by the server. Let $\{a_k^a\}$ be the number of arrivals of stream a in period k and $\{a_k^b\}$ be the number of arrivals of stream b. Find a set of recursions by which it becomes possible to analyze the waiting time distribution of each of the streams. Assume that the service capacity c is constant for all periods, and that jobs that arrive in period k can also be served in period k.*

1.3.19. *[Splitting streams] Consider a machine (a paint mixing machine) that produces products for two separate downstream machines A and B (two different paint packaging machines), each with its own queue. Suppose we want to analyze the queue in front of station A. For this we need to know the arrivals to station A, that is, the departures of the mixing station that go to station A. Provide a set of recursions to simulate this system.*

1.3.20. *[Queue with setups, not obligatory] One server serves two parallel queues, one at a time. After serving one queue until it is empty, the server moves to the other queue. The change of queue requires one period setup time.*

Hints

h.1.3.5. Consider a numerical example. Suppose $L_{k-1} = 20$. Suppose that the capacity is $c_k = 3$ for all k. Then a job that arrives in the kth period, must wait at least 20/3 (plus rounding) periods before it can leave the system. Now generalize this numerical example.

Solutions

s.1.3.1. $d_1 = 7$, $L_1 = 8 - 7 + 5 = 6$, $d_2 = 6$, $L_2 = 6 - 6 + 4 = 4$, $d_3 = 4$, $L_3 = 4 - 4 + 9 = 9$, and so on.

s.1.3.2. The assumption is that the jobs arrive at the start of period k, before service in period k starts, rather than at the end of the period. Therefore the arrivals at period k can also be served during period k.

s.1.3.3. Starting from ,

$$\begin{aligned} d_k &= \min\{L_{k-1} + a_k, c_k\}, \\ L_k &= L_{k-1} + a_k - d_k, \end{aligned}$$

we get

$$\begin{aligned} L_k &= L_{k-1} + a_k - d_k \\ &= L_{k-1} + a_k - \min\{L_{k-1} + a_k, c_k\} \\ &= \max\{L_{k-1} + a_k - c_k, 0\}. \end{aligned}$$

In code it looks like this.

```
>>> a = [0, 10, 3, 6]
>>> c = [0, 5, 5, 5]
>>> L = [0] * len(a)
>>> d = [0] * len(a)

>>> for k in range(1, len(a)):
...     L[k] = max(L[k - 1] + a[k] - c[k], 0)
...     d[k] = L[k - 1] + a[k] - L[k]
...

>>> print(L)
[0, 5, 3, 4]
>>> print(d)
[0, 5, 5, 5]
```

s.1.3.4. All jobs that arrive such that the queue becomes larger than K must be dropped.

First $d_k = \min\{L_{k-1}, c_k\}$. Then, $L'_k = L_{k-1} + a_k - d_k$ is the queue without blocking. Then $L_k = \min\{L'_k, K\}$ is the queue with blocking. Finally, the loss $l_k = L'_k - L_k$, i.e., the excess arrivals. The fraction lost is l_k/a_k.

```
>>> a = [0, 10, 3, 6]
>>> c = [0, 5, 5, 5]
>>> L = [0] * len(a)
>>> d = [0] * len(a)
>>> l = [0] * len(a)  # loss

>>> K = 8

>>> for k in range(1, len(a)):
...     d[k] = min(L[k - 1], c[k])
...     Lp = L[k - 1] + a[k] - d[k]
```

```
...     L[k] = min(Lp, K)
...     l[k] = Lp - L[k]
...

>>> print(L)
[0, 8, 6, 7]
>>> print(l)
[0, 2, 0, 0]
```

s.1.3.5. Let's tag the first customer that arrives in period k. This tagged customer sees L_{k-1} customers in the system, hence the tagged customer's service can only start after all L_{k-1} customers have been served (assuming FIFO scheduling of course). Now, if $L_{k-1} - c_k \geq 0$, there are still people in front of the tagged customer, either in queue or in service. In fact, as long as $c_k + c_{k+1} + \cdots + c_m \leq L_{k-1}$, there are still customers in front of the tagged customer. Therefore, when the tagged customer leaves the system, the inequality $c_k + c_{k+1} + \cdots + c_m > L_{k-1}$ must hold. (To check your understanding, explain that $c_k + c_{k+1} + \cdots + c_m \geq L_{k-1}$ is not correct.)

We can also tag the last customer that arrives in period k. This customer will certainly have left if m is such that $c_k + \cdots + c_m \geq L_{k-1} + a_k$.

In formulas, the above comes down to the following. A job that arrives in period k cannot be served before period

$$W_{-,k} := \min\left\{m : \sum_{i=k}^{k+m} c_i > L_{k-1}\right\},$$

and it must have been served before period

$$W_{+,k} := \min\left\{m : \sum_{i=k}^{k+m} c_i \geq L_{k-1} + a_k\right\}.$$

Thus, the waiting time of jobs arriving in period k must lie in the interval $[W_{-,k}, W_{+,k}]$.

Here is some code to see how it works. It is less easy then you might think to get the details correct.

```
>>> a = [3] * 10
>>> c = [7] * len(a)
>>> L = [0] * len(a)
>>> d = [0] * len(a)

>>> L[0] = 20


>>> for k in range(1, len(a)):
...     d[k] = min(L[k - 1] + a[k], c[k])
...     L[k] = L[k - 1] + a[k] - d[k]
...

>>> print(L)
[20, 16, 12, 8, 4, 0, 0, 0, 0, 0]

>>> k = 1
```

```
>>> m = 0
>>> tot_service = c[k + m]
>>> print(m, L[k - 1] - tot_service)
0 13
>>> while tot_service <= L[k - 1]:
...     m += 1
...     tot_service += c[k + m]
...     print(m, L[k - 1] - tot_service)
...
1 6
2 -1
>>> W_min = m
>>> print(W_min)
2
>>> print(L[k - 1] / c[0])
2.857142857142857

>>> print(m, L[k - 1] + a[k] - tot_service)
2 2
>>> while tot_service <= L[k - 1] + a[k]:
...     m += 1
...     tot_service += c[k + m]
...     print(m, L[k - 1] + a[k] - tot_service)
...
3 -5
>>> W_max = m
>>> print(W_max)
3
>>> print((L[k - 1] + a[k]) / c[0])
3.2857142857142856
```

s.1.3.6. The machine amount produced in period k is d_k. Thus, pd_k is the amount lost, neglecting rounding errors for the moment. Thus, $(1-p)d_k$ items can be used to serve customer demand. Therefore

$$d_k = \min\{L_{k-1}, c_k\},$$
$$L_k = L_{k-1} - d_k(1-p) + a'_k.$$

Can you use these recursions to show that the long-run average service capacity $n^{-1}\sum_{i=1}^{n} c_i$ must be larger than $\lambda(1+p)$?

If you like, you can incorporate time-dependent failure rates $\{p_k\}$ too. Whether this makes practical sense depends on the context of course.

```
>>> a = [0, 4, 8, 2, 1]
>>> c = [3] * len(a)
>>> L = [0] * len(a)
>>> d = [0] * len(a)
>>> p = 0.2
```

```
>>> L[0] = 2

>>> for k in range(1, len(a)):
...     d[k] = (1 - p) * min(L[k - 1], c[k])
...     L[k] = L[k - 1] + a[k] - d[k]
...
>>> print(L)
[2, 4.4, 10.0, 9.6, 8.2]
```

s.1.3.7. Suppose again that a fraction p is faulty. Since these faulty items require less processing time than a new job, the service capacity c_k, i.e., the number of jobs that can be processed in period k, is a bit bigger; part of the capacity is spent on new jobs but another part is spent on the faulty jobs. By the assumptions above, the repair of a faulty requires half of the work of a new job, and the faulty jobs are processed with priority over the new jobs. Assume queue A contains the faulty items, and queue B the new jobs. Then the recursions become:

$$
\begin{aligned}
d_{A,k} &= \min\{L_{A,k-1}, 2c_k\}, (\text{ as faulty jobs require half of the processing time}) \\
c_{B,k} &= c_k - d_{A,k}/2, \\
d_{B,k} &= \min\{L_{B,k-1}, c_{B,k}\}, \\
a_{A,k} &= p d_{B,k-1}, \\
a_{B,k} &= (1-p) d_{B,k-1}, \\
L_{A,k} &= L_{A,k-1} + a_{A,k} - d_{A,k}, \\
L_{B,k} &= L_{B,k-1} + a_{B,k} - d_{B,k}.
\end{aligned}
$$

In code:

```
>>> a = [0, 4, 8, 2, 1]
>>> c = [3] * len(a)
>>> L_A = [0] * len(a)
>>> d_A = [0] * len(a)
>>> L_B = [0] * len(a)
>>> d_B = [0] * len(a)
>>> p = 0.2

>>> L_A[0] = 2
>>> L_B[0] = 8

>>> for k in range(1, len(a)):
...     d_A[k] = min(L_A[k - 1], 2 * c[k])
...     c_B = c[k] - d_A[k] / 2
...     d_B[k] = min(L_B[k - 1], c_B)
...     a_A = p * d_B[k - 1]
...     a_B = a[k]
...     L_A[k] = L_A[k - 1] + a_A - d_A[k]
...     L_B[k] = L_B[k - 1] + a_B - d_B[k]
...
```

```
>>> print(L_A)
[2, 0.0, 0.4, 0.6, 0.5599999999999999]
>>> print(L_B)
[8, 10.0, 15.0, 14.2, 12.5]
```

s.1.3.8. First consider the dynamics of the queue. Since the capacity is chosen at the start of the period (the machine is switched on for c_k units even if there is less demand):

$$\begin{aligned} d_k &= \min\{L_{k-1} + a_k, c_k\}, \\ L_k &= L_{k-1} + a_k - d_k. \end{aligned}$$

The cost to operate the server during period k is βc_k. Thus, the total cost up to some time T for the server must be $\beta \sum_{k=1}^{T} c_k$. In period k we also have to pay hL_k, since h is the cost per customer per period in the system. Thus, the long-run average cost is

$$\frac{1}{T} \sum_{k=1}^{T} \big(\beta c_k + hL_k\big).$$

It is an interesting problem to find a policy that minimizes (the expectation of) this cost. The policy is such that the capacity for period k can be chosen based on the queue length L_{k-1} and *estimates* of the demands $\hat{d}_k, \hat{d}_{k+1}, \ldots$. This problem is not easy, as far as I can see.

s.1.3.9. First we need to implement the N-policy. For this we need an extra variable to keep track of the state of the server. Let $I_k = 1$ if the machine is on in period k and $I_k = 0$ if it is off. Then $\{I_k\}$ must satisfy the relation

$$I_{k+1} = \begin{cases} 1 & \text{if } L_k \geq N, \\ I_k & \text{if } 0 < L_k < N, \\ 0 & \text{if } L_k = 0, \end{cases}$$

and assume that $I_0 = 0$ at the start, i.e., the machine is off. Thus, we can write:

$$I_{k+1} = \mathbb{1}_{L_k \geq N} + I_k \, \mathbb{1}_{0 < L_k < N} + 0 \cdot \mathbb{1}_{L_k = 0}.$$

With I_k it follows that $d_k = \min\{L_{k-1}, I_k c_k\}$, from which L_k follows, and so on.

The machine cost for period k is βI_k, because only when the machine is on we have to pay β, and the queueing cost is hL_k. To determine the total switching cost is harder as we need to determine how often the machine has been switched on up to time T. Observe that the machine is switched on in period k if $I_{k-1} = 0$ and $I_k = 1$. Thus, whenever $I_k - I_{k-1} = 1$ the machine is switched on, when $I_k - I_{k-1} = 0$ the state of the machine remains the same, and if $I_k - I_{k-1} = -1$ the machine is switched off. In other words $\max\{I_k - I_{k-1}, 0\}$ captures what we need. The total cost up to time T becomes:

$$\sum_{k=1}^{T} \big(\beta I_k + hL_k + K \max\{I_k - I_{k-1}, 0\}\big).$$

s.1.3.10. One model could be to let the server only switch on when the queue is larger than some threshold t, and when the server is on, it works at rate c per period. In that case, $c_k = c \, \mathbb{1}_{L_{k-1} > t}$.

s.1.3.11. Let c_k^i be the capacity allocated to queue i in period k. The fair rule gives that

$$c_k^1 = \frac{L_{k-1}^1}{L_{k-1}^1 + L_{k-1}^2} c = c - c_k^2.$$

Then,

$$\begin{aligned} d_k^1 &= \min\{L_{k-1}^1, c_k^1\}, \\ L_k^1 &= L_{k-1}^1 + a_k^1 - d_k^1, \end{aligned}$$

and likewise for the other queue.

s.1.3.12. The rules below implement a strict priority rule for jobs of type A, i.e., jobs sent into queue A.

$$\begin{aligned} d_{A,k} &= \min\{L_{A,k-1}, c_k\}, \\ c_{B,k} &= c_k - d_{A,k}, \\ d_{B,k} &= \min\{L_{B,k-1}, c_{B,k}\}, \\ L_{A,k} &= L_{A,k-1} + a_{A,k} - d_{A,k}, \\ L_{B,k} &= L_{B,k-1} + a_{B,k} - d_{B,k}. \end{aligned}$$

As an aside, another interesting rule to distribute the capacity c_k over the queues could be based on the principle of *equal division of the contested sum*. This principle is based on game-theoretic ideas. Aumann and Maschler applied this principle to clarify certain division rules discussed in the Talmud to divide the legacy among a number of inheritors, each having a different claim size.

s.1.3.13. First determine how much capacity queue B minimally needs in period k:

$$c_{B,k} = \min\{L_{B,k-1}, r_B\}.$$

Observe that, since $c_k \geq r_A + r_B$, this rule ensures that queue A receives at least its reserved capacity r_A.

Since queue A is served with priority, we first give all capacity, except what queue B minimally needs, to queue A:

$$d_{A,k} = \min\{L_{A,k-1}, c_k - c_{B,k}\}.$$

And then we can give any leftover capacity to queue B, if needed.

$$d_{B,k} = \min\{L_{B,k-1}, c_k - d_{A,k}\}.$$

An example is the weekly capacity offered by a psychiatrist at a hospital. Part of the weekly capacity is reserved/allocated/assigned to serve certain patient groups. For instance, each week the psychiatrist does at most five intakes of new patients, provided there are any, and the rest of the capacity is used to treat other patients. The existing patients can also be divided in different groups, each receiving a minimal capacity. If there are fewer patients of some group, then the capacity can be planned/given to other patient groups.

s.1.3.14. Queue A can use all capacity, except what is reserved for queue B:

$$d_{A,k} = \min\{L_{A,k-1}, c_k - r_B\}.$$

Observe that, since $c_k \geq r_A + r_B$, this rule ensures that queue A receives at least its reserved capacity r_A.

Queue B cannot receive more than $c_k - r_A$, since r_A is allocated to queue A, and if queue A does not use all of r_A, then the surplus is lost. Also, queue B cannot get more than $c_k - d_{A,k}$ as this is what remains after serving queue A. Thus, letting $c_{B,k} = \min\{c_k - r_A, c_k - d_{A,k}\} = c_k - \max\{r_A, d_{A,k}\}$, we see that for queue B:

$$d_{B,k} = \min\{L_{B,k-1}, c_{B,k}\}.$$

An example can be the operation room of a hospital. There is a weekly capacity, part of the capacity is reserved for emergencies. It might not be possible to assign this reserved capacity to other patient groups, because it should be available at all times for emergency patients. A result of this is that unused capacity is lost.

In practice it may not be as extreme as in the model, but still part of the unused capacity is lost. 'Use it, or lose it', is what often, but not always, applies to service capacity.

s.1.3.15. Let a_k be the external arrivals at station A. Then:

$$\begin{aligned} d_k^A &= \min\{L_{k-1}^A, c_k^A\}, \\ L_k^A &= L_{k-1}^A - d_k^A + a_k. \end{aligned} \tag{1.3.3}$$

The departures of the first station during period k are the arrivals at station B at the end of period k, i.e., $a_k^B = d_k^A$. Thus,

$$\begin{aligned} a_k^B &= d_k^A, \\ d_k^B &= \min\{L_{k-1}^B, c_k^B\}, \\ L_k^B &= L_{k-1}^B - d_k^B + a_k^B. \end{aligned} \tag{1.3.4}$$

s.1.3.16.

$$\begin{aligned} d_k^A &= \min\{L_{k-1}^A, c_k^A, M - L_{k-1}^B\}, \\ L_k^A &= L_{k-1}^A - d_k^A + a_k, \\ a_k^B &= d_k^A, \text{ (ensures all jobs first pass station A and then station B)} \\ d_k^B &= \min\{L_{k-1}^B, c_k^B\}, \\ L_k^B &= L_{k-1}^B - d_k^B + a_k^B. \end{aligned} \tag{1.3.5}$$

This is a bit subtle: since there is room $M - L_{k-1}^B$ at the intermediate buffer and $d_k^A \leq M - L_{k-1}^B$, we know that in the worst case, i.e., when $c_k^B = 0$, still $L_k^B = L_{k-1}^B + d_k^A$. Thus, we are sure that the queue length of the intermediate queue will not exceed M.

There is still a small problem: What if for the first initial periods $M < L_{k-1}^B$. Then $M - L_{k-1}^B < 0$ and then by the specification above, $d_k^A < 0$. This is not what we want. Therefore,

$$d_k^A = \min\{L_{k-1}^A, c_k^A, \max\{M - L_{k-1}^B, 0\}\}.$$

s.1.3.17. Realize that Stations A and B have their own arrivals.

$$\begin{aligned}
d_k^A &= \min\{L_{k-1}^A, c_k^A\},\\
L_k^A &= L_{k-1}^A - d_k^A + a_k^A,\\
d_k^B &= \min\{L_{k-1}^B, c_k^B\},\\
L_k^B &= L_{k-1}^B - d_k^B + a_k^B,\\
a_k^C &= d_k^A + d_k^B,\\
d_k^C &= \min\{L_{k-1}^C, c_k^C\},\\
L_k^C &= L_{k-1}^C - d_k^C + a_k^C.
\end{aligned} \tag{1.3.6}$$

s.1.3.18. The behavior of the queue length process is easy:

$$\begin{aligned}
d_k &= \min\{L_{k-1} + a_k^a + a_k^b, c\},\\
L_k &= L_{k-1} + a_k^a + a_k^b - d_k.
\end{aligned}$$

To determine the waiting times, observe that any arrival in period k, independent of the stream, has to wait until all jobs at the start of the period in queue, i.e., $L_{k-1} - d_k$, are cleared; note that we assume here that the jobs served in period k depart at the start of the interval. Thus, the minimal waiting time is $W_{k,-} = \lfloor L_{k-1}/c \rfloor$. Similarly, the maximal waiting time is $W_{k,+} = \lfloor (L_{k-1} + a_k^a + a_k^b)/c \rfloor$.

The remaining problem is to make a model to 'distribute' the time between $W_{k,-}$ and $W_{k,+}$ over the two streams.

A simple model is to assume that the waiting time is averaged over the jobs. Then each job perceives a waiting time of

$$\frac{W_{k,-} + W_{k,+}}{2}.$$

Another way is to give priority to a customers, but only for the jobs that arrive in this period. (Hence, this is different from the priority queue. In that case priority customers can overtake customers that arrived earlier. In the present case this is not allowed, jobs of type a that arrive in period k cannot overtake b jobs that arrived prior to period k.) Making this completely explicit (so that the recursion can be fed to the computer) requires a bit of work however. It is important to understand the trick we will discuss now because we will use it to model queueing systems with batching. Observe that the first job of the a stream only has to wait for $W_{k,-}$, the second job must wait for $W_{k,-} + 1$, and so on. Thus, the waiting time W_k^a for the a_k^a items is such that

$$W_{k,-}^a := \lfloor L_{k-1}/c \rfloor \le W_k^a \le \lfloor (L_{k-1} + a_k^a)/c \rfloor =: W_{k,+}^a.$$

Similarly, for the b jobs must be

$$W_{k,-}^b := \lfloor (L_{k-1} + a_k^a)/c \rfloor \le W_k^b \le \lfloor (L_{k-1} + a_k^a + a_k^b)/c \rfloor =: W_{k,+}^b.$$

Note that $W_{k,+}^a = W_{k,-}^b$. It is then sensible to set

$$\begin{aligned}
W_k^a &= \frac{W_{k,-}^a + W_{k,+}^a}{2},\\
W_k^b &= \frac{W_{k,-}^b + W_{k,+}^b}{2}.
\end{aligned}$$

s.1.3.19. 1. Realize that the recursions of (1.3.1), applied to the queueing situation at the first machine, provide us with the total number of departures d_k during period k. However, it does not tell us about the type of these departures. Thus, to compute the queue in front of station A, we need to know the number of departures of type A, rather than the total number of departures of the first station.

2. It is reasonable that the number of jobs of type A in queue at the first station is equal to

$$L_k \frac{\lambda_A}{\lambda_A + \lambda_B}.$$

It is therefore reasonable to assume that the capacity c_k of the first station is also shared in this proportion to type A and B jobs. Thus, the number of departures to station A is

$$d_k(A) = \frac{\lambda_A}{\lambda_A + \lambda_B} \min\{L_{k-1}, c_k\}.$$

The rest of the recursions is very similar to what we did in earlier exercises.

s.1.3.20. We need an extra variable p_k that specifies which queue is being served. If $p_k = 0$, the server is moving from one queue to the other, if $p_k = 1$, the server is at queue 1, and if $p_k = 2$ it is at queue 2. Then, with

$$d_k^1 = \min\{L_{k-1}^1, c_k^1 \mathbb{1}_{p_k=1}\},$$
$$d_k^2 = \min\{L_{k-1}^2, c_k^2 \mathbb{1}_{p_k=2}\},$$

we can specify the evolution of the queue length processes. So it remains to deal with p. For this, we use a 'truth table' to compute p_{k+1} as a function of p_k and whether $L_k^1 > 0$ or not and $L_k^2 > 0$ or not.

$\mathbb{1}_{L_k^1>0}$	$\mathbb{1}_{L_k^2>0}$	p_k	p_{k+1}
0	0	0	0
0	0	1	1
0	0	2	2
1	0	0	1
1	0	1	1
1	0	2	0 (switch over time)
0	1	0	2
0	1	1	0 (switch over time)
0	1	2	2
1	1	0	1 (to break ties)
1	1	1	1
1	1	2	2

1.4 EXPONENTIAL DISTRIBUTION

Theory and Exercises

In Section 1.2 we introduced the Poisson process as a natural model of the (random) number of jobs arriving during intervals of time. As we will see in the sections to come, we can model a

single-server queueing system in continuous time if we specify the (probability) distribution of the inter-arrival times, i.e., the time between consecutive arrival epochs of jobs. A particularly fruitful model for the distribution of the inter-arrival times is the exponential distribution because, as it turns out, it is intimately related to the Poisson distribution. Besides explaining this relation, we derive many useful properties of the exponential distribution, in particular, that it is *memoryless*.

We say that X is an *exponentially distributed* random variable with mean $1/\lambda$ if

$$\mathsf{P}(X \le t) = 1 - e^{-\lambda t},$$

and then we write $X \sim \text{Exp}(\lambda)$.

The Poisson process N and exponentially distributed inter-arrival times are intimately related: A counting process $\{N(t)\}$ is a *Poisson process* with rate λ if and only if the inter-arrival times $\{X_i\}$ are *i.i.d.* (*independent and identically distributed*) and exponentially distributed with mean $1/\lambda$, in short,

$$X_i \sim \text{Exp}(\lambda) \Leftrightarrow N(t) \sim P(\lambda t).$$

We next provide further relations between the Poisson distribution and the exponential distribution.

1.4.1. *In the above expression for* $\mathsf{P}(X \le t)$ *we require that* $\lambda > 0$*. What would happen if you would allow* λ *to be zero or negative?*

1.4.2. *If the random variable* $X \sim \text{Exp}(\lambda)$*, show that its mean* $\mathsf{E}[X] = \frac{1}{\lambda}$.

1.4.3. *If* $X \sim \text{Exp}(\lambda)$*, show that its second moment* $\mathsf{E}\left[X^2\right] = \frac{2}{\lambda^2}$.

1.4.4. *If* $X \sim \text{Exp}(\lambda)$*, show that the* variance $\mathsf{V}[X] = \lambda^{-2}$.

1.4.5. *Suppose* $X \sim \text{Exp}(\lambda)$*. Show that, only when* $t < \lambda$*, the moment-generating function* $M_X(t)$ *is finite.*

1.4.6. *If* $X \sim \text{Exp}(\lambda)$*, show that its moment-generating function is equal to*

$$M_X(t) = \frac{\lambda}{\lambda - t}.$$

1.4.7. *Use the moment-generating function of* $X \sim \text{Exp}(\lambda)$ *to show that*

$$\mathsf{E}[X] = \frac{1}{\lambda}, \qquad \mathsf{E}\left[X^2\right] = \frac{2}{\lambda^2}.$$

1.4.8. *Show that, when* $X \sim \text{Exp}(\lambda)$*, the SCV of* X *is* 1.

We now provide a number of relations between the Poisson distribution and the exponential distribution to conclude that a process N_λ is a Poisson process with rate λ iff the inter-arrival times $\{X_i\}$ between individual jobs are i.i.d. $\sim \text{Exp}(\lambda)$.

1.4.9. *If* N_λ *is a Poisson process with rate* λ*, show that the time* X_1 *to the first arriving job is* $\text{Exp}(\lambda)$*.*

1.4.10. *Assume that inter-arrival times* $\{X_i\}$ *are i.i.d. and* $\sim \text{Exp}(\lambda)$*. Let the arrival time of the* i*th job be* $A_i = \sum_{k=1}^{i} X_k$*. Show that*

$$\mathsf{E}[A_i] = \frac{i}{\lambda}.$$

1.4.11. *Let A_i be the arrival time of customer i and set $A_0 = 0$. Assume that the inter-arrival times $\{X_i\}$ are i.i.d. with exponential distribution with mean $1/\lambda$ for some $\lambda > 0$. Prove that A_i has density*

$$f_{A_i}(t) = \lambda e^{-\lambda t} \frac{(\lambda t)^{i-1}}{(i-1)!}.$$

1.4.12. *Use the density f_{A_i} of 1.4.11 to show that $\mathsf{E}[A_i] = i/\lambda$.*

1.4.13. *If the inter-arrival times $\{X_i\}$ are i.i.d. $\sim \mathrm{Exp}(\lambda)$, prove that the number $N(t)$ of arrivals during the interval $[0,t]$ is Poisson distributed.*

We now introduce another fundamental concept. A random variable X is called *memoryless* when it satisfies

$$\mathsf{P}(X > t+h | X > t) = \mathsf{P}(X > h).$$

In words, the probability that X is larger than some time $t+h$, conditional on it being larger than a time t, is equal to the probability that X is larger than h.

1.4.14. *Show that $X \sim \mathrm{Exp}(\lambda)$ is memoryless.*

In fact, it can be shown that only exponential random variables have the memoryless property. The proof of this fact requires quite some work; we refer the reader to the literature if s/he wants to check this, see e.g. Yushkevich and Dynkin [1969, Appendix 3].

1.4.15. *If $X \sim \mathrm{Exp}(\lambda)$ and $S \sim \mathrm{Exp}(\mu)$, and X and S are independent, show that*

$$Z = \min\{X, S\} \sim \mathrm{Exp}(\lambda + \mu),$$

hence $\mathsf{E}[Z] = (\lambda+\mu)^{-1}$.

1.4.16. *If $X \sim \mathrm{Exp}(\lambda)$, $S \sim \mathrm{Exp}(\mu)$, and X and S are independent, show that*

$$\mathsf{P}(X \le S) = \frac{\lambda}{\lambda+\mu}.$$

Hints

h.1.4.1. The interpretation of the function $1 - e^{-\lambda t}$ is a probability. What are the consequences of this? What happens if $\lambda = 0$?

h.1.4.2.

$$\mathsf{E}[X] = \int_0^\infty t f(t)\,\mathrm{d}t = \int_0^\infty t\lambda e^{-\lambda t}\,\mathrm{d}t,$$

where $f(t) = \lambda e^{-\lambda t}$ is the density function of the distribution function F of X. Now solve the integral.

h.1.4.3.

$$\mathsf{E}\left[X^2\right] = \int_0^\infty t^2 \lambda e^{-\lambda t}\,\mathrm{d}t = \frac{2}{\lambda^2}.$$

h.1.4.4. Use (1.1.2e).

h.1.4.6.

$$M_X(t) = \mathsf{E}[\exp(tX)] = \int_0^\infty e^{tx} f(x)\,\mathrm{d}x.$$

h.1.4.7. Use (1.1.3c) and (1.1.3d).

h.1.4.8. Use (1.2.4)

h.1.4.9. What is the meaning of $\mathsf{P}(N(t)=0)$?

h.1.4.10. What is $\mathsf{E}[X+Y]$ for some general random variables (each with finite mean)?

h.1.4.11. Check the result for $i=1$ by filling in $i=1$ (just to be sure that you have read the formula right), and compare the result to the exponential density. Then write $A_i = \sum_{k=1}^{i} X_k$, and compute the moment-generating function for A_i and use that the inter-arrival times X_i are independent. Use the moment-generating function of X_i.

h.1.4.12. Use the standard integral $\int_0^\infty x^n e^{-\alpha x}\,\mathrm{d}x = \alpha^{-n-1}n!$. Another way would be to use that, once you have the moment-generating function of some random variable X, $\mathsf{E}[X] = \frac{\mathrm{d}}{\mathrm{d}t}M_X(t)|_{t=0}$.

h.1.4.13. Realize that $\mathsf{P}(N(t)=k) = \mathsf{P}(A_k \le t) - \mathsf{P}(A_{k+1} \le t)$.

h.1.4.14. Condition on the event $X > t$.

h.1.4.15. Use that if $Z = \min\{X,S\} > x$, it then must be that $X > x$ and $S > x$. Then use independence of X and S.

h.1.4.16. Define the joint distribution of X and S and carry out the computations, or use conditioning, or use the result of the previous exercise.

Solutions

s.1.4.1. If $\lambda < 0$, then $1-e^{-\lambda t}$ grows to $-\infty$ if $t \to \infty$. Just by itself this is not a problem. However, in our case $1-e^{-\lambda t}$ has the interpretation of a distribution function. Now, recall that a distribution function is bounded to values in the interval $[0,1]$.

Suppose $\lambda = 0$. Then $\mathsf{P}(X \le t) = 1 - e^0 = 0$. In words, this would mean that the probability of a finite inter-arrival time between any two customers is zero. So, no customers can arrive in this case.

s.1.4.2.

$$\begin{aligned}
\mathsf{E}[X] &= \int_0^\infty t\lambda e^{-\lambda t}\,\mathrm{d}t, \quad \text{density is } \lambda e^{-\lambda t} \\
&= \lambda^{-1}\int_0^\infty u e^{-u}\,\mathrm{d}u, \quad \text{by change of variable } u = \lambda t, \\
&= -\lambda^{-1}\, u e^{-u}\big|_0^\infty + \lambda^{-1}\int_0^\infty e^{-u}\,\mathrm{d}u \\
&= -\lambda^{-1}\, e^{-u}\big|_0^\infty = \frac{1}{\lambda}.
\end{aligned}$$

s.1.4.3.

$$\mathsf{E}\left[X^2\right] = \int_0^\infty t^2\lambda e^{-\lambda t}\,\mathrm{d}t$$

$$
\begin{aligned}
&= \lambda^{-2} \int_0^\infty u^2 e^{-u}\, \mathrm{d}u, \quad \text{by change of variable } u = \lambda t, \\
&= -\lambda^{-2}\, t^2 e^{-t}\big|_0^\infty + 2\lambda^{-2} \int_0^\infty t e^{-t}\, \mathrm{d}t \\
&= -2\lambda^{-2}\, t e^{-t}\big|_0^\infty + 2\lambda^{-2} \int_0^\infty e^{-t}\, \mathrm{d}t \\
&= -2\lambda^{-2}\, e^{-t}\big|_0^\infty \\
&= 2/\lambda^2.
\end{aligned}
$$

s.1.4.4. By the previous problems, $\mathsf{E}\left[X^2\right] = 2/\lambda^2$ and $\mathsf{E}[X] = 1/\lambda$. Hence the variance $\mathsf{V}[X] = \mathsf{E}\left[X^2\right] - (\mathsf{E}[X])^2 = \frac{2}{\lambda^2} - (\frac{1}{\lambda})^2 = \lambda^{-2}$.

s.1.4.5.

$$
\begin{aligned}
M_X(t) = \mathsf{E}\left[e^{tX}\right] &= \int_0^\infty e^{tx} f(x)\, \mathrm{d}x \\
&= \lambda \int_0^\infty e^{tx} e^{-\lambda x}\, \mathrm{d}x = \lambda \int_0^\infty e^{-(\lambda - t)x}\, \mathrm{d}x.
\end{aligned}
$$

This last integral only converges when $\lambda - t > 0$.

s.1.4.6.

$$
\begin{aligned}
M_X(t) = \mathsf{E}[\exp(tX)] &= \int_0^\infty e^{tx} f(x)\, \mathrm{d}x = \int_0^\infty e^{tx} \lambda e^{-\lambda x}\, \mathrm{d}x \\
&= \lambda \int_0^\infty e^{(t-\lambda)x}\, \mathrm{d}x = \frac{\lambda}{\lambda - t}.
\end{aligned}
$$

s.1.4.7. By 1.4.6, $M'_X(t) = \lambda/(\lambda - t)^2$. Hence, $M'_X(0) = 1/\lambda$. Next, $M''_X(t) = 2\lambda/(\lambda - t)^3$, hence $\mathsf{E}\left[X^2\right] = M''_X(0) = 2\lambda/\lambda^3 = 2\lambda^{-2}$.

s.1.4.8. By the previous problems, $\mathsf{V}[X] = 1/\lambda^2$ and $\mathsf{E}[X] = 1/\lambda$, hence

$$
C^2 = \frac{\mathsf{V}[X]}{(\mathsf{E}[X])^2} = \frac{1/\lambda^2}{1/\lambda^2} = 1.
$$

s.1.4.9. If there are no arrivals in some interval $[0, t]$, then it must be that $N(t) = 0$. Hence, for the first inter-arrival time X_1:

$$
\mathsf{P}(X_1 > t) = \mathsf{P}(N(t) = 0) = e^{-\lambda t} \frac{(\lambda t)^0}{0!} = e^{-\lambda t}.
$$

s.1.4.10. The simplest way to obtain the answer is to use that the expectation is a linear operator, i.e., $\mathsf{E}[X + Y] = \mathsf{E}[X] + \mathsf{E}[Y]$ for any r.v. X and Y. Then,

$$
\mathsf{E}[A_i] = \mathsf{E}\left[\sum_{k=1}^{i} X_k\right] = i\, \mathsf{E}[X] = \frac{i}{\lambda}.
$$

(Just as a reminder, $\mathsf{E}[XY] \neq \mathsf{E}[X]\,\mathsf{E}[Y]$ in general. Only when X and Y are uncorrelated (which is implied by independence), the product of the expectations is the expectation of the products.)

s.1.4.11. One way to find the distribution of A_i is by using the moment-generating function $M_{A_i}(t) = \mathsf{E}\left[e^{tA_i}\right]$ of A_i. Let X_i be the inter-arrival time between customers i and $i-1$, and $M_X(t)$ the associated moment-generating function. Using the i.i.d. property of the $\{X_i\}$,

$$\begin{aligned} M_{A_i}(t) &= \mathsf{E}\left[e^{tA_i}\right] = \mathsf{E}\left[\exp\left(t\sum_{k=1}^{i} X_k\right)\right] \\ &= \prod_{k=1}^{i} \mathsf{E}\left[e^{tX_k}\right] = \prod_{k=1}^{i} M_{X_k}(t) = \prod_{k=1}^{i} \frac{\lambda}{\lambda - t} = \left(\frac{\lambda}{\lambda - t}\right)^i. \end{aligned}$$

From a table of moment-generating functions it follows immediately that $A_i \sim \Gamma(i, \lambda)$, i.e., A_i is Gamma distributed.

s.1.4.12.

$$\mathsf{E}[A_i] = \int_0^\infty t f_{A_i}(t)\,\mathrm{d}t = \int_0^\infty t\lambda e^{-\lambda t}\frac{(\lambda t)^{i-1}}{(i-1)!}\,\mathrm{d}t.$$

Thus,

$$\mathsf{E}[A_i] = \frac{1}{(i-1)!}\int_0^\infty e^{-\lambda t}(\lambda t)^i\,\mathrm{d}t = \frac{i!}{(i-1)!\lambda} = \frac{i}{\lambda},$$

where we used the hint.

What if we would use the moment-generating function, as derived by the previous exercise?

$$\begin{aligned} \mathsf{E}[A_i] &= \frac{\mathrm{d}}{\mathrm{d}t} M_{A_i}(t)\bigg|_{t=0} \\ &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\lambda}{\lambda - t}\right)^i\bigg|_{t=0} \\ &= i\left(\frac{\lambda}{\lambda - t}\right)^{i-1}\frac{\lambda}{(\lambda - t)^2}\bigg|_{t=0} = \frac{i}{\lambda}. \end{aligned}$$

s.1.4.13. We want to show that

$$\mathsf{P}(N(t) = k) = e^{-\lambda t}\frac{(\lambda t)^k}{k!}.$$

Now observe that $\mathsf{P}(N(t) = k) = \mathsf{P}(A_k \le t) - \mathsf{P}(A_{k+1} \le t)$. Using the density of A_{k+1} and applying partial integration leads to

$$\begin{aligned} \mathsf{P}(A_{k+1} \le t) &= \lambda\int_0^t \frac{(\lambda s)^k}{k!}e^{-\lambda s}\,\mathrm{d}s \\ &= \lambda\frac{(\lambda s)^k}{k!}\frac{e^{-\lambda s}}{-\lambda}\bigg|_0^t + \lambda\int_0^t \frac{(\lambda s)^{k-1}}{(k-1)!}e^{-\lambda s}\,\mathrm{d}s \\ &= -\frac{(\lambda t)^k}{k!}e^{-\lambda t} + \mathsf{P}(A_k \le t). \end{aligned}$$

We are done.

s.1.4.14. By the definition of conditional probability

$$\mathsf{P}(X > t + h | X > t) = \frac{\mathsf{P}(X > t + h, X > t)}{\mathsf{P}(X > t)} = \frac{\mathsf{P}(X > t + h)}{\mathsf{P}(X > t)} = \frac{e^{-\lambda(t+h)}}{e^{-\lambda t}} = e^{-\lambda h} = \mathsf{P}(X > h).$$

Thus, no matter how long we have been waiting for the next arrival to occur, the probability that it will occur in the next h seconds remains the same.

This property seems to be also vindicated in practice: suppose that a patient with a broken arm just arrived at the emergency room of a hospital, what does that tell us about the time the next patient will be brought in? Not much, as most of us will agree.

s.1.4.15. Use that X and S are independent to get

$$\begin{aligned}\mathsf{P}(Z > x) &= \mathsf{P}(\min\{X,S\} > x) = \mathsf{P}(X > x \text{ and } S > x) = \mathsf{P}(X > x)\mathsf{P}(S > x)\\ &= e^{-\lambda x}e^{-\mu x} = e^{-(\lambda+\mu)x}.\end{aligned}$$

s.1.4.16. There is more than one way to show that $\mathsf{P}(X \le S) = \lambda/(\lambda+\mu)$.

Observe first that X and S, being exponentially distributed, have a density. Moreover, as they are independent, the joint density takes the form

$$f_{X,S}(x,y) = f_X(x)f_S(y) = \lambda\mu e^{-\lambda x}e^{-\mu y}.$$

With this,

$$\begin{aligned}\mathsf{P}(X \le S) &= \mathsf{E}[\,\mathbb{1}_{X\le S}]\\ &= \int_0^\infty \int_0^\infty \mathbb{1}_{x\le y} f_{X,S}(x,y)\,\mathrm{d}y\,\mathrm{d}x\\ &= \lambda\mu \int_0^\infty \int_0^\infty \mathbb{1}_{x\le y} e^{-\lambda x}e^{-\mu y}\,\mathrm{d}y\,\mathrm{d}x\\ &= \lambda\mu \int_0^\infty e^{-\mu y}\int_0^y e^{-\lambda x}\,\mathrm{d}x\,\mathrm{d}y\\ &= \mu \int_0^\infty e^{-\mu y}(1-e^{-\lambda y})\,\mathrm{d}y\\ &= \mu \int_0^\infty (e^{-\mu y}-e^{-(\lambda+\mu)y})\,\mathrm{d}y\\ &= 1-\frac{\mu}{\lambda+\mu}\\ &= \frac{\lambda}{\lambda+\mu}.\end{aligned}$$

1.5 CONSTRUCTION OF THE SINGLE-SERVER QUEUEING PROCESS IN CONTINUOUS TIME

Theory and Exercises

In Section 1.3 we modeled time as progressing in discrete 'chunks': minutes, hours, days, and so on. For given numbers of arrivals and capacity per period, we use the recursions (1.3.1) to compute the departures and queue length per period. However, we can also model time in a continuous way, so that jobs can arrive at any moment and have arbitrary service times. In this section, we consider a single-server FIFO queueing process in continuous time.

Assume we are given the *arrival process* $\{A(t); t \ge 0\}$, i.e., the number of jobs that arrived during $[0,t]$. Thus, $\{A(t); t \ge 0\}$ is a *counting process*.

From this arrival process, we can derive various other interesting concepts, such as the *arrival times* of individual jobs. Especially, if we know that $A(s) = k-1$ and $A(t) = k$, then the arrival time A_k of the kth job must lie somewhere in $(s,t]$. Thus, from $\{A(t)\}$, we can define[3]

$$A_k = \min\{t : A(t) \geq k\}, \tag{1.5.1}$$

and set $A_0 = 0$. Once we have the set of arrival times $\{A_k\}$, the set of *inter-arrival times* $\{X_k, k = 1,2,\ldots\}$ between consecutive customers can be constructed as

$$X_k = A_k - A_{k-1}. \tag{1.5.2}$$

However, often the basic data consists of the inter-arrival times $\{X_k; k = 1,2,\ldots\}$ rather than the arrival times $\{A_k\}$ or the arrival process $\{A(t)\}$. Then we construct the arrival times as

$$A_k = A_{k-1} + X_k,$$

with $A_0 = 0$. From the arrival times $\{A_k\}$ we can, in turn, construct the arrival process $\{A(t)\}$ as

$$A(t) = \max\{k : A_k \leq t\}. \tag{1.5.3}$$

Thus, from the inter-arrival times $\{X_k\}$ it is possible to construct $\{A_k\}$ and $\{A(t)\}$, and from $\{A(t)\}$ we can obtain $\{A_k\}$ and $\{X_k\}$.

1.5.1. *Show that we can also define $A(t)$ as*

$$A(t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \leq t}. \tag{1.5.4}$$

Thus, we just count all arrivals that occur up to and including time t.

1.5.2. *Is it true that the following mappings are correct:*

$$A_k : \mathbb{N} \to \mathbb{R}, \quad \textit{job id (integer) to arrival time (real number)},$$
$$A(t) : \mathbb{R} \to \mathbb{N}, \quad \textit{time (real number) to number of jobs (integer)}?$$

1.5.3. *What are the meanings of $A_{A(t)}$ and $A(A_n)$?*

1.5.4. *In view of the above, can $A(t)$ be defined as $\min\{k : A_k \geq t\}$ or as $\min\{k : A_k > t\}$?*

The *service times* of the jobs are given by the sequence $\{S_k\}$. Typically we assume that the service times are i.i.d, and also independent of the arrival process $\{A(t)\}$. With the arrival times and service times we can construct the *waiting time in queue* $\{W_{Q,k}\}$ as seen by the jobs at the moment they arrive. From Fig. 3 it is evident that

$$W_{Q,k} = [W_{Q,k-1} + S_{k-1} - X_k]^+. \tag{1.5.5}$$

With this it is easy to compute the waiting times: from the initial condition $W_{Q,0} = 0$ we can obtain $W_{Q,1}$, and then $W_{Q,2}$, and so on.

1.5.5. *Explain that $W_{Q,k} = [W_{Q,k-1} + S_{k-1} - X_k]^+$.*

[3] If we want to be mathematically precise, we must take inf rather than min. However, in this book we do not want to distinguish between subtleties.

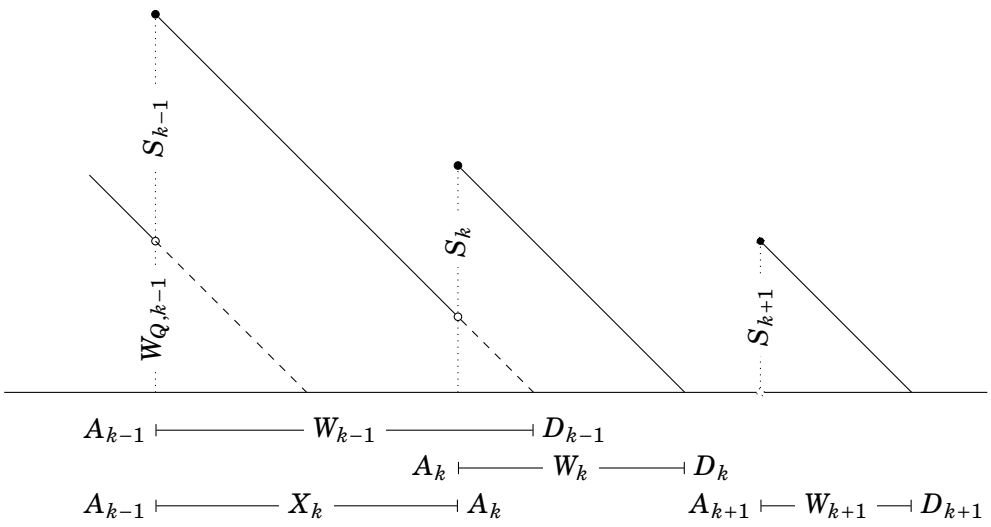


Figure 3: Construction of the single-server queue in continuous time. The sojourn time W_k of the kth job is the sum of the work in queue $W_{Q,k}$ at its arrival epoch A_k and its service time S_k; its departure time is then $D_k = A_k + W_k$. The waiting time of job k is clearly equal to $W_{k-1} - X_k$. We also see that job $k+1$ arrives at an empty system, hence its sojourn time $W_{k+1} = S_{k+1}$. Finally, the virtual waiting time process is shown by the lines with slope -1.

1.5.6. *If $S \sim U[0,7]$ and $X \sim U[0,10]$, where $U[I]$ stands for the uniform distribution concentrated on the interval I, compute $\mathsf{P}(S - X \le u)$, for S and X independent.*

The time job k *leaves the queue and moves to the server* is given by

$$\tilde{A}_k = A_k + W_{Q,k},$$

because a job can only move to the server after its arrival plus the time it needs to wait in queue. Note that we here explicitly use the FIFO assumption. Right after the job moves from the queue to the server, its service starts. Thus, $\tilde{A}_k$ is also the epoch at which the service of job k starts.

After completing its service, the job leaves the system. Hence, the *departure time of the system* is

$$D_k = \tilde{A}_k + S_k.$$

This in turn specifies the departure process $\{D(t)\}$ as

$$D(t) = \max\{k : D_k \le t\} = \sum_{k=1}^{\infty} \mathbb{1}_{D_k \le t}.$$

The *sojourn time*, or *waiting time in the system*, W_k, is the time a job spends in the entire system. With the above relations we see that

$$W_k = D_k - A_k = \tilde{A}_k + S_k - A_k = W_{Q,k} + S_k, \tag{1.5.6}$$

where each of these equations has its own interpretation.

1.5.7. *Explain that the following recursion for W_k is valid:*

$$W_{Q,k} = [W_{k-1} - X_k]^+, \qquad W_k = W_{Q,k} + S_k = [W_{k-1} - X_k]^+ + S_k, \tag{1.5.7}$$

from which follows a recursion for D_k:

$$D_k = A_k + W_k. \tag{1.5.8}$$

1.5.8. *Assume that $X_1 = 10$, $X_2 = 5$, $X_3 = 6$ and $S_1 = 17$, $S_2 = 20$ and $S_3 = 5$, compute the arrival times, waiting times in queue, the sojourn times and the departure times for these three customers.*

1.5.9. *Assume that $X_k = 10$ minutes and $S_k = 11$ minutes for all k, i.e., X_k and S_k are deterministic and constant. What are λ and μ? Compute A_k, W_k, D_k as functions of k. Then find expressions for $A(t)$ and $D(t)$.*

1.5.10. *Suppose that $X_k \in \{1,3\}$ such that $\mathsf{P}(X_k = 1) = \mathsf{P}(X_k = 3)$ and $S_k \in \{1,2\}$ with $\mathsf{P}(S_k = 1) = \mathsf{P}(S_k = 2)$. If $W_{Q,0} = 3$, what are the distributions of $W_{Q,1}$ and $W_{Q,2}$?*

1.5.11. *Explain the following recursions for a single server queue:*

$$\begin{aligned} A_k &= A_{k-1} + X_k, \\ D_k &= \max\{A_k, D_{k-1}\} + S_k, \\ W_k &= D_k - A_k. \end{aligned} \tag{1.5.9}$$

The *virtual waiting time process* $\{V(t)\}$ is the amount of waiting that an arrival would see if it would arrive at time t. To construct $\{V(t)\}$, we simply draw lines that start at points (A_k, W_k) and have slope -1, unless the line hits the x-axis, in which case the virtual waiting time remains zero until the next arrival occurs.

1.5.12. *Provide a specification of the virtual waiting time process $\{V(t)\}$ for all t.*

Once we have the arrival and departure processes it is easy to compute the *number of jobs in the system* at time t as, cf. Fig. 4,

$$L(t) = A(t) - D(t) + L(0), \tag{1.5.10}$$

where $L(0)$ is the number of jobs in the system at time $t = 0$; typically we assume that $L(0) = 0$. As in a queueing system, jobs can be in queue or in service, we distinguish between the number in the system $L(t)$, the number in queue $L_Q(t)$, and the number of jobs in service $L_s(t)$.

1.5.13. *In 1.5.9, find an expression for $L(A_k-)$.*

In summary, starting from a sequence of inter-arrival times $\{X_k\}$ and service times $\{S_k\}$ we can obtain a set of recursions by which we can simulate a queueing process in continuous time. A bit of experimentation with Python (or R perhaps) will reveal that this is easy.

1.5.14. *Explain that if we know $\tilde{A}(t)$, i.e. the number of jobs that departed from the queue up to time t, then*

$$\begin{aligned} L_Q(t) &= A(t) - \tilde{A}(t), \\ L_s(t) &= \tilde{A}(t) - D(t) = L(t) - L_Q(t). \end{aligned}$$

1.5.15. *Why don't we need a separate notation for $D_s(t)$, the number of jobs that departed from the server?*

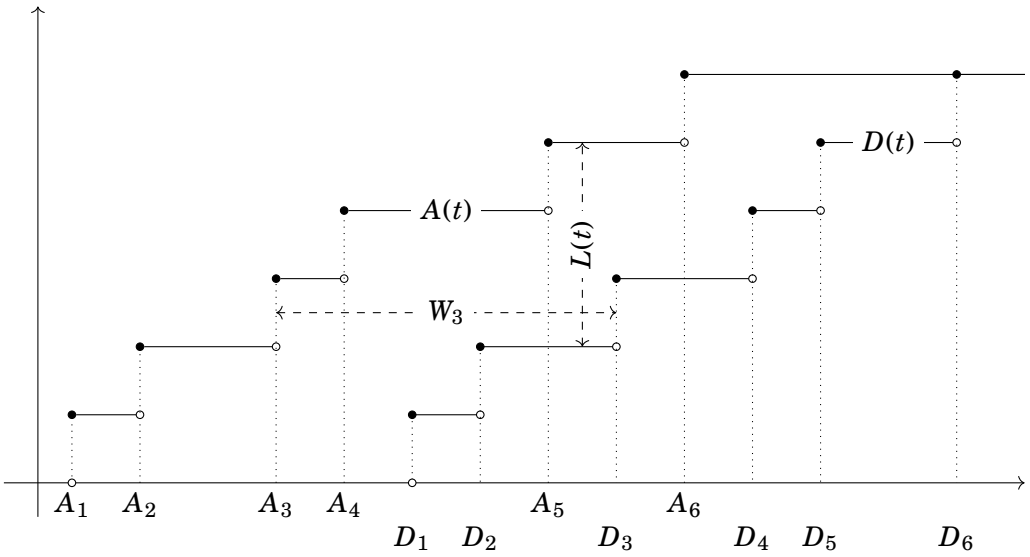


Figure 4: Relation between the arrival process $\{A(t)\}$, the departure process $\{D(t)\}$, the number in the system $\{L(t)\}$ and the waiting times $\{W_k\}$. In particular, $L(t)$ is the difference between the graphs of $A(t)$ and $D(t)$.

1.5.16. *Is $\tilde{A}(t) \le D(t)$ or $\tilde{A}(t) \ge D(t)$?*

1.5.17. *Consider a multi-server queue with m servers. Suppose that at some time t it happens that $\tilde{A}(t) - D(t) < m$ even though $A(t) - D(t) > m$. How can this occur?*

1.5.18. *Show that $L(t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \le t < D_k}$ when the system starts empty.*

1.5.19. *Show that $L(A_k) > 0 \implies A_k \le D_{k-1}$.*

Define the number of jobs in the system as seen by the kth arrival as

$$L(A_k-). \tag{1.5.11}$$

Observe that we write A_k-, and not A_k. As we are concerned with a process with jumps, we need to be quite particular about left and right limits at jump epochs.

1.5.20. *With the recursions* (1.5.7) *it is apparently easy to compute the waiting time (in queue), but it is less simple to compute the number of jobs in queue or in the system. In this exercise we develop an algorithm to compute the number of jobs in the system as seen by arrivals. Explain why the following (algorithmic efficient) procedure works:*

$$L(A_k-) = L(A_{k-1}-) + 1 - \sum_{i=k-1-L(A_{k-1}-)}^{k-1} \mathbb{1}_{D_i < A_k}.$$

1.5.21. *In 1.5.20, why do we take $i = k-1-L(A_{k-1}-)$ in the sum, and not $i = k-2-L(A_{k-1}-)$?*

1.5.22. *[Not obligatory] Try to extend the recursions* (1.5.9) *to a situation in which one queue is served by two servers.*

Hints

h.1.5.1. What is $\mathbb{1}_{A_k \le t}$ if $A_k \le t$?

h.1.5.4. Compare this to the definition in (1.5.3).

h.1.5.6. This is elementary, hence it might appear trivial, but it's not... In fact, I had a hard time finding a simple way to get the answer. It is good practice to try yourself before looking at the answer. Check also the previous problem, and make a drawing of the region over which you have to integrate.

h.1.5.8. The intent of this exercise is to make you familiar with the notation.

BTW, such simple test cases are also very useful to test computer code. The numbers in the exercise are one such simple case. You can check the results by hand; if the results of the simulator are different, there is a problem.

h.1.5.10. Use (1.5.7).

h.1.5.12. Make a plot of the function $A_{A(t)} - t$. What is the meaning of $V(A_{A(t)})$? What is $V(A_{A(t)}) + A_{A(t)} - t$?

h.1.5.18. Use Boolean algebra. Write, for notational ease, $A = \mathbb{1}_{A_k \le t}$ and $\bar{A} = 1 - A = \mathbb{1}_{A_k > t}$, and define something similar for D. Then show that $A - D = A\bar{D} - \bar{A}D$, and show that $\bar{A}D = 0$. Finally sum over k.

h.1.5.19. Use that $L(A_k) > 0$ means that the system contains at least one job at the time of the kth arrival, and that $A_k \le D_{k-1}$ means that job k arrives before job $k-1$ departs.

h.1.5.22. The problem is that in a multi-server queueing systems, unlike for single-server queues, jobs can overtake each other: a small job that arrives after a very large job can still leave the system earlier. After trying for several hours, I obtained an inelegant method. A subsequent search on the web helped a lot. The solution below is a modification of N. Krivulin, 'Recursive equations based models of queueing systems'.

Solutions

s.1.5.1. For every $A_k \le t$, we have that $\mathbb{1}_{A_k \le t} = 1$, and else the indicator is 0. Hence, in the summation we count the number of times $A_k \le t$.

s.1.5.2. Yes, it is true.

s.1.5.3. $A(t)$ is the number of arrivals during $[0,t]$. Suppose that $A(t) = n$. This nth job arrived at time A_n. Thus, $A_{A(t)}$ is the arrival time of the last job that arrived before or at time t. In a similar vein, A_n is the arrival time of the nth job. Thus, the number of arrivals up to time A_n, i.e., $A(A_n)$, must be n.

s.1.5.4. Suppose $A_3 = 10$ and $A_4 = 20$. Take $t = 15$. Then $\min\{k : A_k \ge 15\} = 4$ since $A_3 < t = 15 < A_4$. On the other hand $\max\{k : A_k \le t\} = 3$. And, indeed, at time $t = 15$, 3 jobs arrived, not 4. So defining $A(t)$ as $\min\{k : A_k \ge t\}$ is not OK. This example also shows that in general $A(t) \ne \min\{k : A_k > t\}$. So, neither definition is correct.

s.1.5.5. The waiting time of the kth arrival must be equal to the waiting time of the $k-1$th customer plus the amount of *service time* required by job $k-1$ minus the time that elapses between the arrival of job $k-1$ and job k, unless the server becomes idle between jobs $k-1$ and k.

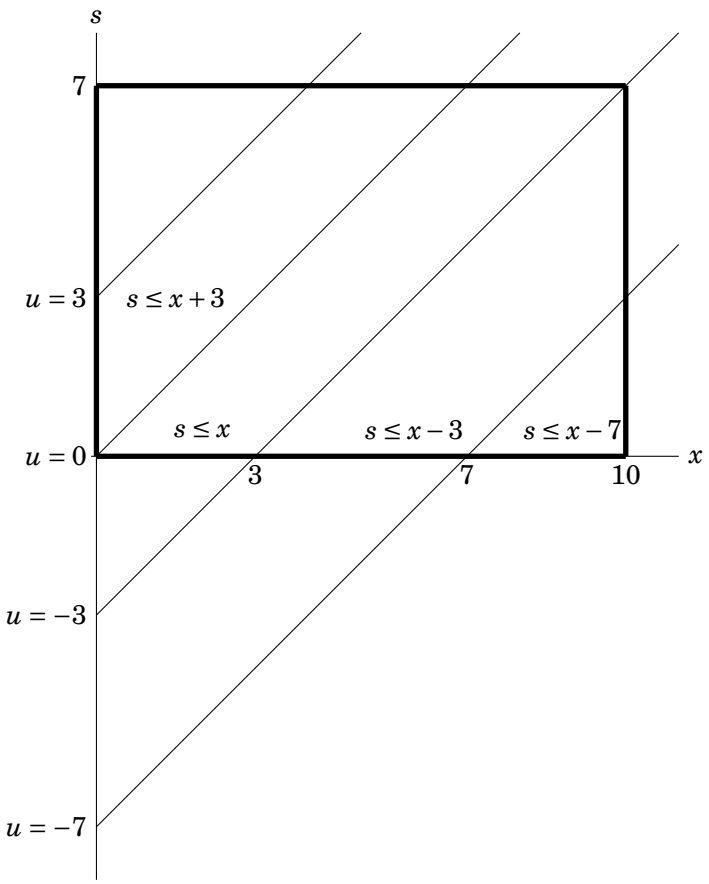
s.1.5.6. The joint density of S and X is given by

$$f_{XS}(x,s) = f_X(x) \cdot f_S(s) = \frac{1}{10} \mathbb{1}_{0\le x\le 10} \cdot \frac{1}{7} \mathbb{1}_{0\le s\le 7},$$

since X and S are independent. Thus,

$$\begin{aligned} \mathsf{P}(S - X \le u) = \mathsf{E}[\mathbb{1}_{S-X\le u}] &= \frac{1}{70} \int_0^{10} \int_0^7 \mathbb{1}_{s-x\le u} \, \mathrm{d}s \, \mathrm{d}x \\ &= \frac{1}{70} \int_0^{10} \int_0^7 \mathbb{1}_{s\le x+u} \, \mathrm{d}s \, \mathrm{d}x. \end{aligned}$$

Now we need to chop up the domain of $\mathsf{P}(S - X \le u)$, for which we use the figure below.



It is clear that the indicated rectangle has no overlap with the set of points (x,s) such that $s \le u + x$ for $u < -10$. (To see this, draw the line $s = x - 10$ in the figure.) At $u = -10$, the overlap is a single point, at $(10,0)$. Thus,

$$\mathsf{P}(S - X \le u) = 0, \quad \text{for } u \le -10.$$

When $u \in [-10,-3]$ we need to integrate over the triangle that results from cutting the line $s = x+u$ with the rectangle. The area is

$$70\,\mathsf{P}(S-X \le u) = \frac{(10+u)^2}{2}, \quad \text{for } -10 \le u \le -3,$$

where we multiply with 70 to get the normalization right.

When $u \in [-3,0]$, we integrate over a parallelogram with base $3+u$ and height 7 plus the triangle below the line $s = x-3$. The area is

$$70\,\mathsf{P}(S-X \le u) = (3+u)7 + \frac{(10-3)^2}{2} = 7u + \frac{91}{2}, \quad \text{for } -3 \le u \le 0.$$

For $u \in [0,7]$, we integrate over the trapezoid that results from intersecting the set $\{(x,s) : x \le s \le s+u\}$ and the rectangle plus the parallelogram plus the triangle below the line $s = x-3$. The area is

$$70\,\mathsf{P}(S-X \le u) = \frac{7^2}{2} - \frac{(7-u)^2}{2} + 3\cdot 7 + \frac{49}{2} = 7u - \frac{u^2}{2} + \frac{91}{2}, \quad \text{for } 0 \le u \le 7.$$

Finally, for $u \ge 7$, the set $s \le x+u$ covers the entire rectangle. Hence,

$$70\,\mathsf{P}(S-X \le u) = 70, \quad \text{for } 7 \le u.$$

Given the amount of effort I had to put into getting this answer, I wanted to check it. So I went to Wolfram Alpha (which is a great site for symbolic computations), and typed this:

```
\int_{0}^{10} \int_0^7 Boole[s<= x + u] ds dx,
```

so, once you know LaTeX you can use Wolfram Alpha. Wolfram Alpha turned it to

```
Integrate[Boole[s <= u + x], {x, 0, 10}, {s, 0, 7}]
```

If you fill this in at Wolfram, you'll get the results that we obtained above in seconds, rather than in one hour or so (depending on your proficiency with carrying out integrals).

s.1.5.7. Suppose k arrives right after job $k-1$. Then job k has to wait until job $k-1$ leaves, in other words, the service of job k can start only after job $k-1$ leaves. Consequently, in this case, $W_{Q,k} = W_{k-1}$. If there is time X_k between jobs k and $k-1$, then job k has to spend X_k less in queue. This explains the first formula. The rest follows right away.

s.1.5.8. Let's feed it to the computer. Mind that in Python (just like in C, and so on), arrays start at index 0, not at index 1.

```
>>> X = [0, 10, 5, 6]
>>> S = [0, 17, 20, 5]
>>> A = [0, 0, 0, 0]
>>> for i in range(1, len(X)):
...  A[i] = A[i-1] + X[i]
...
>>> A
[0, 10, 15, 21]
```

```
>>> WQ = [0, 0, 0, 0]
>>> for i in range(1, len(X)):
...  WQ[i] = max(WQ[i-1] + S[i-1] - X[i], 0)
...
>>> WQ
[0, 0, 12, 26]

>>> ST = [0,0,0,0]
>>> for i in range(1, len(X)):
...  ST[i] = WQ[i] + S[i]
...
>>> ST
[0, 17, 32, 31]

>>> D = [0, 0, 0, 0]
>>> for i in range(1, len(X)):
...  D[i] = A[i] + WQ[i] + S[i]
...
>>> D
[0, 27, 47, 52]
```

s.1.5.9. $\lambda = 6$ per hour, and $\mu = 60/11$ per hour. Note that $\mu < \lambda$. $A_0 = 0$, $A_1 = 10$, $A_2 = 20$, and so on. Hence, $A_k = 10k$. $W_{Q,0} = 0$, $W_{Q,1} = \max\{0+0-10, 0\} = 0$. $W_{Q,2} = \max\{0+11-10, 0\} = 1$. $W_{Q,3} = \max\{1+11-10, 0\} = 2$. Hence, $W_{Q,k} = k-1$ for $k \geq 1$. Thus, $W_k = k-1+11 = k+10$ for $k \geq 1$, and $D_k = 10k + k + 10 = 11k + 10$. Note that W_k increases linearly as a function of k. All in all, $A(t) = \lfloor t/10 \rfloor$, and $D(t) = \lfloor (t-10)/11 \rfloor$.

s.1.5.10. First find the distribution of $Y_k := S_{k-1} - X_k$ so that we can write $W_{Q,k} = [W_{Q,k-1} + Y_k]^+$. Use independence of $\{S_k\}$ and $\{X_k\}$:

$$\mathsf{P}(Y_k = -2) = \mathsf{P}(S_{k-1} - X_k = -2) = \mathsf{P}(S_{k-1} = 1, X_k = 3) = \mathsf{P}(S_{k-1} = 1)\mathsf{P}(X_k = 3) = \frac{1}{4}.$$

Dropping the dependence on k for ease, we get

$$\begin{aligned}
\mathsf{P}(Y = -2) = \mathsf{P}(S - X = -2) = \mathsf{P}(S = 1, X = 3) = \mathsf{P}(S = 1)\mathsf{P}(X = 3) &= \frac{1}{4},\\
\mathsf{P}(Y = -1) = \mathsf{P}(S = 2)\mathsf{P}(X = 3) &= \frac{1}{4},\\
\mathsf{P}(Y = 0) = \mathsf{P}(S = 1)\mathsf{P}(X = 1) &= \frac{1}{4},\\
\mathsf{P}(Y = 1) = \mathsf{P}(S = 2)\mathsf{P}(X = 1) &= \frac{1}{4}.
\end{aligned}$$

With this

$$\begin{aligned}
\mathsf{P}\left(W_{Q,1} = 1\right) = \mathsf{P}\left(W_{Q,0} + Y = 1\right) = \mathsf{P}(3 + Y = 1) = \mathsf{P}(Y = -2) &= \frac{1}{4},\\
\mathsf{P}\left(W_{Q,1} = 2\right) = \mathsf{P}(3 + Y = 2) = \mathsf{P}(Y = -1) &= \frac{1}{4},\\
\mathsf{P}\left(W_{Q,1} = 3\right) = \mathsf{P}(3 + Y = 3) = \mathsf{P}(Y = 0) &= \frac{1}{4},
\end{aligned}$$

$$\mathsf{P}\left(W_{Q,1}=4\right)=\mathsf{P}(3+Y=4)=\mathsf{P}(Y=1)=\frac{1}{4}.$$

And, then

$$\begin{aligned}\mathsf{P}\left(W_{Q,2}=1\right)&=\mathsf{P}\left(W_{Q,1}+Y=1\right)=\sum_{i=1}^{4}\mathsf{P}\left(W_{Q,1}+Y=1\,\middle|\,W_{Q,1}=i\right)\mathsf{P}\left(W_{Q,1}=i\right)\\&=\sum_{i=1}^{4}\mathsf{P}\left(i+Y=1\,\middle|\,W_{Q,1}=i\right)\frac{1}{4}=\sum_{i=1}^{4}\mathsf{P}\left(Y=1-i\,\middle|\,W_{Q,1}=i\right)\frac{1}{4}\\&=\frac{1}{4}\sum_{i=1}^{4}\mathsf{P}(Y=1-i)=\frac{1}{4}(\mathsf{P}(Y=0)+\mathsf{P}(Y=-1)+\mathsf{P}(Y=-2))=\frac{3}{16}.\end{aligned}$$

s.1.5.11. Of course, the service of job k cannot start before it arrives. Hence, it cannot leave before A_k+S_k. Therefore it must be that $D_k\geq A_k+S_k$. But the service of job k can also not start before the previous job, i.e. job $k-1$, left the server. Thus job k cannot start before D_{k-1}. To clarify it somewhat further, define S'_k as the earliest start of job k. Then it must be that $S'_k=\max\{A_k,D_{k-1}\}$—don't confuse the earliest start S'_k and the service time S_k—and $D_k=S'_k+S_k$.

s.1.5.12. There is a funny way to do this. Recall from a previous exercise that if $A(t)=n$, then A_n is the arrival time of the nth job. Thus, the function $A_{A(t)}$ provides us with arrival times as a function of t. When $t=A_{A(t)}$, i.e., when t is the arrival time of the $A(t)$th job, we set $V(t)=V(A_{A(t)})=W_{A(t)}$, i.e., the virtual waiting time at the arrival time $t=A_{A(t)}$ is equal to the waiting time of the $A(t)$th job. Between arrival moments, the virtual waiting time decreases with slope 1, until it hits 0. Thus,

$$V(t)=[V(A_{A(t)})-(t-A_{A(t)})]^+=[W_{A(t)}+(A_{A(t)}-t)]^+.$$

The notation may be a bit confusing, but it is in fact very simple. Take some t, look back at the last arrival time before time t, which is written as $A_{A(t)}$. (In computer code these times are easy to find.) Then draw a line with slope -1 from the waiting time that the last arrival saw.

s.1.5.13. Recall that $A(t)=\lfloor t/10\rfloor$, and $D(t)=\lfloor(t-10)/11\rfloor$. Hence,

$$L(A_k-)=k-1-D(A_k-)=k-1-D(10k-)=k-1-\left\lfloor\frac{(10k-)-10}{11}\right\rfloor.$$

The computation is a bit tricky since sometimes arrivals and departures coincide. (Consider for instance $t=120$.)

From this example you can infer that it is necessary that the service rate is greater than the arrival rate, i.e., $\mu>\lambda$, for otherwise, the queue length keeps on increasing (in the long run).

s.1.5.14. We defined $\tilde{A}(t)$ as the amount of people up to time t that left the queue and moved to the server. It is then clear to see that the number of jobs in queue $L_Q(t)$ is equal to the number amount of jobs that have arrived, i.e., $A(t)$, minus the number of jobs that left the queue, i.e., $\tilde{A}(t)$. Using the same reasoning for $L_s(t)$, the second line also follows.

s.1.5.15. Because $D_s(t)=D(t)$. Once customers leave the server, their service is completed, and they leave the queueing system.

s.1.5.16. All customers that left the system must have left the queue. Thus, $\tilde{A}(t) \geq D(t)$.

s.1.5.17. In this case, there are servers idling while there are still customers in queue. If such events occur, we say that the server is not work-conservative.

s.1.5.18.

$$\begin{aligned} L(t) &= A(t) - D(t) \\ &= \sum_{k=1}^{\infty} \mathbb{1}_{A_k \leq t} - \sum_{k=1}^{\infty} \mathbb{1}_{D_k \leq t} \\ &= \sum_{k=1}^{\infty} [\mathbb{1}_{A_k \leq t} - \mathbb{1}_{D_k \leq t}]. \end{aligned}$$

Write for the moment $A = \mathbb{1}_{A_k \leq t}$ and $\bar{A} = 1 - A = \mathbb{1}_{A_k > t}$, and likewise for D. Now we can use Boolean algebra to see that $\mathbb{1}_{A_k \leq t} - \mathbb{1}_{D_k \leq t} = A - D = A(D + \bar{D}) - D = AD + A\bar{D} - D = A\bar{D} - D(1 - A) = A\bar{D} - D\bar{A}$. But $D\bar{A} = 0$ since $D\bar{A} = \mathbb{1}_{D_k \leq t}\mathbb{1}_{A_k > t} = \mathbb{1}_{D_k \leq t < A_k}$ which would mean that the arrival time A_k of the kth job would be larger than its departure time D_k. As $A\bar{D} = \mathbb{1}_{A_k \leq t < D_k}$

$$\begin{aligned} L(t) &= \sum_{k=1}^{\infty} [\mathbb{1}_{A_k \leq t} - \mathbb{1}_{D_k \leq t}] \\ &= \sum_{k=1}^{\infty} \mathbb{1}_{A_k \leq t < D_k}. \end{aligned}$$

Boolean algebra is actually a really nice way to solve logical puzzles. If you are interested, you can find some examples on my homepage.

s.1.5.19. In a sense, the claim is evident, for, if the system contains a job when job k arrives, it cannot be empty. But if it is not empty, then at least the last job that arrived before job k, i.e., job $k-1$, must still be in the system. That is, $D_{k-1} \geq A_k$. A more formal proof proceeds along the following lines. Using that $A(A_k) = k$ and $D(D_{k-1}) = k - 1$,

$$\begin{aligned} L(A_k) > 0 \Leftrightarrow A(A_k) - D(A_k) > 0 \Leftrightarrow k - D(A_k) > 0 \Leftrightarrow k > D(A_k) \\ \Leftrightarrow k - 1 \geq D(A_k) \Leftrightarrow D(D_{k-1}) \geq D(A_k) \Leftrightarrow D_{k-1} \geq A_k, \end{aligned}$$

where the last relation follows from the fact that $D(t)$ is a counting process, hence monotone non-decreasing.

s.1.5.20. Let $L(A_k-)$, i.e., the number of jobs in the system as 'seen by' job k. It must be that $L(A_k-) = k - 1 - D(A_k)$. To see this, assume first that no job has departed when job k arrives. Then job k must see $k-1$ jobs in the system. In general, if at time A_k the number of departures is $D(A_k)$, then the above relation for $L(A_k-)$ must hold. Applying this to job $k-1$ we get that $L(A_{k-1}-) = k - 2 - D(A_{k-1})$.

For the computation of $L(A_k-)$ we do not have to take the departures before A_{k-1} into account as these have already been 'incorporated in' $L(A_{k-1}-)$. Therefore,

$$L(A_k-) = L(A_{k-1}-) + 1 - \sum_{i=k-1-L(A_{k-1}-)}^{k-1} \mathbb{1}_{D_i < A_k}.$$

Suppose $L(A_{k-1}) = 0$, i.e., job $k-1$ finds an empty system at its arrival and $D_{k-1} > A_k$, i.e., job $k-1$ is still in the system when job k arrives. In this case, $L(A_k-) = 1$, which checks with the formula. Also, if $L(A_{k-1}-) = 0$ and $D_{k-1} < A_k$ then $L(A_k-) = 0$. This also checks with the formula.

s.1.5.21. The reason to start at $k-1-L(A_{k-1}-)$ is that the number in the system as seen by job k is $k-1-D(A_k)$ (not $k-2-D(A_k)$). Hence, the jobs with index from $k-1-L(A_{k-1}-), k-L(A_{k-1}-), \ldots, k-1$, could have left the system between the arrival of job $k-1$ and job k.

s.1.5.22. The recursions for the two-server system are this:

$$\begin{aligned} A_k &= A_{k-1} + X_k, \\ C_k &= \max\{A_k, D_{k-2}\} + S_k, \\ M_k &= \max\{M_{k-1}, C_k\}, \\ D_{k-1} &= \min\{M_{k-1}, C_k\}. \end{aligned}$$

Here, C_k is the completion time of job k, and $\{D_k\}$ is a sorted list of departure times. Thus, D_k is the kth departure time; recall this is not necessarily equal to the completion time C_k of the kth job (as jobs may overtake each other). To understand the other equations, we reason like this. By construction, $C_k > D_{k-m}$ (as $S_k > 0$). Therefore, when we arrived at time C_k, $(k-m)$ jobs must have departed. Moreover, by construction, M_k tracks the latest completion time of all k jobs, hence, M_{k-m+1} is the latest completion time of the first $k-m+1$ jobs. Therefore, if $C_k > M_{k-1}$, job k must leave later than the latest of the jobs in $\{1,2,\ldots,k-1\}$. Hence, the latest departure time of the jobs in $\{1,2,\ldots,k-1\}$ jobs must be M_{k-1}. If however, $C_k < M_{k-1}$, then job k leaves earlier than the latest of the jobs in $\{1,2,\ldots,k-1\}$. As $C_k > D_{k-2}$, it must be that $C_k > M_{k-2}$, because D_{k-2} is the latest departure of the jobs in $\{1,2,\ldots,k-2\}$, and this is also equal to M_{k-2}. As a consequence, if $C_k < M_{k-1}$, job k is also the first job that leaves after D_{k-2} (provided of course that $C_{k+1} < C_k$). Thus, all in all $D_{k-1} = \min\{M_{k-1}, C_k\}$.

In an attempt to extend the above to $m > 2$ servers, I came up with this scheme:

$$\begin{aligned} A_k &= A_{k-1} + X_k, \\ C_k &= \max\{A_k, D_{k-m}\} + S_k, \\ M_k &= \max\{M_{k-m+1}, C_k\}, \\ D_{k-m+1} &= \min\{M_{k-m+1}, C_k\}, \end{aligned}$$

but it is not correct. Can you find a counterexample?

1.6 KENDALL'S NOTATION

Theory and Exercises

As became apparent in Sections 1.3 and 1.5, the construction of any single-station queueing process involves three main elements: the distribution of the inter-arrival times between consecutive jobs, the distribution of the service times of the individual jobs, and the number of servers present to process jobs. In this characterization, it is implicit that the inter-arrival times form a set of i.i.d. (independent and identically distributed) random variables, the service times are also i.i.d., and finally, the inter-arrival times and service times are mutually independent.

To characterize the type of queueing process it is common to use *Kendall's abbreviation* $A/B/c/K$, where A is the distribution of the inter-arrival times, B the distribution of the service times, c the number of servers, and K the system size, i.e., the total number of customers that

can be simultaneously present, whether in queue or in service.[4] In this notation it is assumed that jobs are served in first-in-first-out (FIFO) order; FIFO scheduling is also often called first-come-first-served (FCFS).

When at an arrival a number of jobs arrive simultaneously (like a bus at a restaurant), we say that a batch arrives. Likewise, the server can work in batches, for instance, when an oven processes multiple jobs at the same time. We write $A^X/B^Y/c$ to denote that X is the distribution of the arrival batch size and Y is the distribution of the service batch sizes. When $X \equiv Y \equiv 1$, i.e., single batch arrivals and single batch services, we suppress the X and Y in the queueing formula.

Two inter-arrival and service distributions are the most important in queueing theory: the exponential distribution denoted with the shorthand M, as it is memoryless, and a general distribution (with the implicit assumption that its first moment is finite) denoted with G. We write D for a deterministic (constant) random variable.

Familiarize yourself with this notation as it is used continuously in the rest of the book. Here are some exercises to illustrate the notation.

1.6.1. *What is the meaning of $M/M/1$?*

1.6.2. *By how many parameters is the $M/M/1$ queue characterized?*

1.6.3. *What is the $D/D/1$ queue?*

1.6.4. *Is it true that the $M/M/c$ shorthand means that jobs arrive as a Poisson process, job service times are exponentially distributed, and there are c servers.*

1.6.5. *What is the meaning of $M/M/c/K$?*

1.6.6. *What is the meaning of $M/M/c/c$?*

1.6.7. *What is the meaning of $M(n)/M(n)/1$?*

1.6.8. *What is the meaning of $M^X/M/1$?*

1.6.9. *What is the meaning of $M/G/1$?*

1.6.10. *What is the meaning of $M/G/\infty$?*

1.6.11. *What is the meaning of $G/G/1$?*

1.6.12. *What is the meaning of $M/D/1-LIFO$?*

1.6.13. *Is the $M/D/1$ queue a specific type of $M/G/c$ queue?*

You should also understand the differences between different scheduling rules. The next exercise should help with this.

1.6.14. *What are some advantages and disadvantages of using the Shortest Processing Time First (SPTF) rule to serve jobs?*

When a customer finds a large queue in front of it, s/he can use the normal distribution to estimate the distribution of the time s/he will spend in queue. The next exercise shows how.

1.6.15. *Suppose for the $G/G/1$ that a job sees n jobs in the system upon arrival. Use the central limit theorem to estimate the distribution of the waiting time in queue for this job.*

[4] The meaning of K differs among authors. Sometimes it stands for the capacity of the queue, not the entire system. In this book K corresponds to the system's size.

Hints

h.1.6.14. Look up the relevant definitions on Wikipedia or Hall [1991].

h.1.6.15. Let $W_{Q,n} = \sum_{k=1}^{n} S_k$. Since $\{S_k\}$ are assumed to be i.i.d. for the $G/G/1$ queue, $W_{Q,n}$ has mean $\mu_n = n\,\mathsf{E}[S]$ and $\sigma_n^2 = n\,\mathsf{V}[S]$.

Solutions

s.1.6.1. $M/M/1$: The distribution of the inter-arrival times is memoryless, hence exponential, the service times are also memoryless, and there is 1 server. As K is unspecified, the system can contain any number of jobs.

s.1.6.2. The inter-arrival times are exponentially distributed with rate λ; the service times are also exponential, but with parameter μ. Thus, if we know λ and μ, we have fully characterized the parameters of both distributions. Since the number of servers is 1, only λ and μ remain.

s.1.6.3. $D/D/1$: A queueing process with deterministic inter-arrival times, deterministic service times and 1 server.

s.1.6.4. $M/M/c$: A *multi-server* queue with c servers in which all servers have the same service capacity. Jobs arrive according to a Poisson process and have exponentially distributed service times. Hence, it is true.

s.1.6.5. $M/M/c/K$: Inter-arrival times and process times are exponential, and the *system capacity* is K jobs. Thus, the queue can contain at most $K - c$ jobs.

s.1.6.6. $M/M/c/c$: In this system the number of servers is the same as the system capacity, thus the queue length is always zero. This queueing system is useful to determine the number of beds in a hospital; the beds act as servers.

s.1.6.7. $M(n)/M(n)/1$: The inter-arrival times are exponential, just as the service times, but the rates of the arrival and service processes may depend on the queue length n.

s.1.6.8. $M^X/M/1$: Customers arrive with exponentially distributed inter-arrival times. However, each customer brings in a number of jobs, known as a batch. The number of jobs in each batch is distributed as the random variable X. Thus, the arrival process of work is *compound Poisson*.

s.1.6.9. $M/G/1$: The inter-arrival times are exponentially distributed, the service times can have any general distribution (with finite mean), and there is 1 server.

s.1.6.10. $M/G/\infty$: Exponential inter-arrival times, service times can have any distribution, and there is an unlimited supply of servers. This is also known as an *ample server*. Observe that in this queueing process, jobs actually never have to wait in queue; upon arrival there is always a free server available.

s.1.6.11. $G/G/1$: Generally distributed inter-arrival and service times, 1 server.

s.1.6.12. $M/D/1-LIFO$: Now, job service times are deterministic, and the service sequence is last-in-first-out (LIFO).

s.1.6.13. Yes, take $G = D$ and $c = 1$.

s.1.6.14. Advantage: SPTF minimizes the number of jobs in queue. Thus, if you want to keep the shop floor free of jobs, then this is certainly a good rule. Disadvantage: large jobs get near to terrible waiting times, and the variance of the waiting time increases. Thus, the C_s^2 is larger than under FIFO. Also, SPTF does not take due dates into account, thus giving a reliable due date quotation to a customer is hard (near to impossible).

s.1.6.15. Under conditions you can find on the internet,

$$\frac{W_{Q,n} - \mu_n}{\sigma_n} \to \mathcal{N}(0,1), \quad \text{as } n \to \infty,$$

where $\mathcal{N}(0,1)$ is a normally distributed random variable with $\mu = 0$ and $\sigma^2 = 1$. But then

$$\begin{aligned}
&\frac{W_{Q,n} - \mu_n}{\sigma_n} \approx \mathcal{N}(0,1) \iff \\
&W_{Q,n} - \mu_n \approx \sigma_n \mathcal{N}(0,1) \iff \\
&W_{Q,n} - \mu_n \approx \mathcal{N}(0, \sigma_n^2) \iff \\
&\quad W_{Q,n} \approx \mu_n + \mathcal{N}(0, \sigma_n^2) \iff \\
&\quad W_{Q,n} \approx \mathcal{N}(\mu_n, \sigma_n^2) = \mathcal{N}(n\mathsf{E}[X], n\mathsf{V}[S]),
\end{aligned}$$

where we used in the last equation the fact that the variance of a (fixed) sum of i.i.d random variables is equal to the sum of the variances.

1.7 QUEUEING PROCESSES AS REGULATED RANDOM WALKS

Theory and Exercises

In the construction of queueing processes as set out in Section 1.3 we are given two sequences of i.i.d. random variables: the number of arrivals $\{a_k\}$ per period and the service capacities $\{c_k\}$, cf., (1.3.2). Observe that in (1.3.2) the process $\{L_k\}$ shares a resemblance to a random walk $\{Z_k, k = 0,1,\ldots\}$ with Z_k given by

$$Z_k = Z_{k-1} + a_k - c_k. \tag{1.7.1}$$

To see that $\{Z_k\}$ is indeed a random walk, observe that Z makes jumps of size $a_k - c_k, k = 1,\ldots$, and $\{a_k - c_k\}$ is a sequence of i.i.d. random variables since, by assumption, $\{a_k\}$ and $\{c_k\}$ are i.i.d. Clearly, $\{Z_k\}$ is 'free', i.e., it can take positive and negative values, but $\{L_k\}$ is restricted to the non-negative integers. In this section, we show how to build the queueing process $\{L_k\}$ from the random walk $\{Z_k\}$ using a device called a *reflection map*, which gives an elegant construction of a queueing process. Moreover, we can use the probabilistic tools that have been developed for the random walk to analyze queueing systems. One example is the distribution of the time until an especially large queue length is reached; these times can be formulated as *hitting times* of the random walk. Another example is the average time it takes to clear a large queue.

1.7.1. *Show that L_k satisfies the relation*

$$L_k = Z_k - \min_{1\le i\le k} Z_i \wedge 0, \tag{1.7.2}$$

where Z_k is defined by the above random walk and we write $a \wedge b$ for $\min\{a,b\}$.

This recursion for L_k leads to really interesting graphs. In Fig. 5 we take $a_k \sim B(0.3)$, i.e., a_k is Bernoulli-distributed with success parameter $p = 0.3$, i.e., $\mathsf{P}(a_k = 1) = 0.3 = 1 - \mathsf{P}(a_k = 0)$, and $c_k \sim B(0.4)$. In Fig. 6, $a_k \sim B(0.49)$ and the random walk is constructed as

$$Z_k = Z_{k-1} + 2a_k - 1. \tag{1.7.3}$$

Thus, if $a_k = 1$, the random walk increases by one step, while if $a_k = 0$, the random walk decreases by one step, so that $Z_k \neq Z_{k-1}$ always. Observe that this is slightly different from a random walk that satisfies (1.7.1); there, $Z_k = Z_{k-1}$, if $a_k = c_k$.

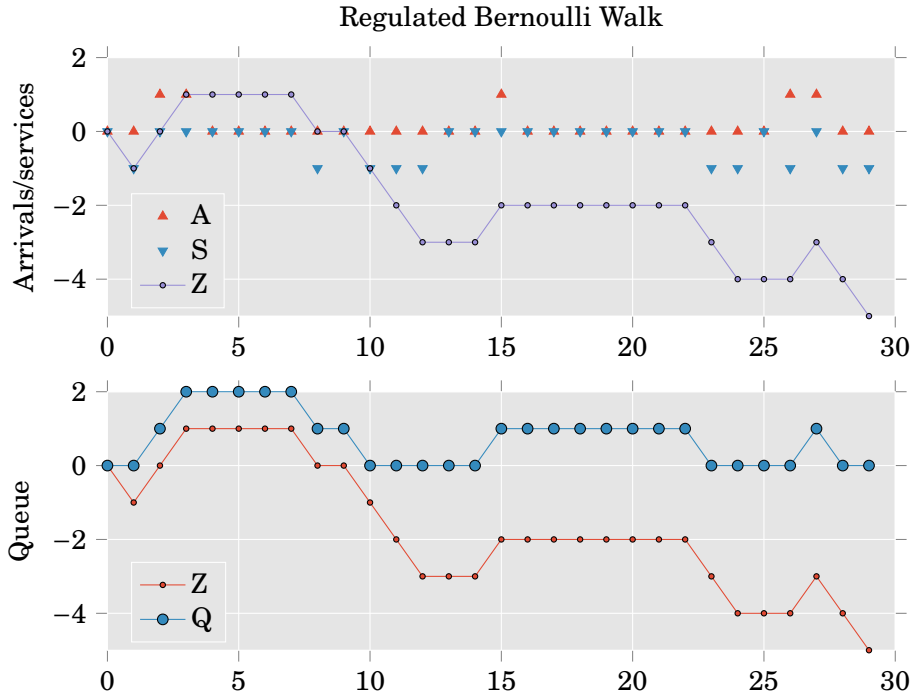


Figure 5: The upper panel shows a graph of the random walk Z. An upward pointing triangle corresponds to an arrival, a downward triangle to a potential service. The lower panel shows the queueing process $\{L_k\}$ as a random walk with reflection.

With (1.7.2), we see that a random walk $\{Z_k\}$ can be converted into a queueing process $\{L_k\}$, and we might try to understand the transient behavior of the latter by investigating the transient behavior of the former. Suppose that $a_k \sim P(\lambda)$ and $c_k \sim P(\mu)$.

1.7.2. *Show that if $\{a_k\}$ forms an i.i.d. sequence of random variables all Poisson distributed $P(\lambda)$ then, $\sum_{j=1}^{k} a_j = P(\lambda k)$.*

With the above exercise,

$$Z_k = Z_0 + N_{\lambda k} - N_{\mu k},$$

and we call $Z = \{Z_k\}$ the *free* (discrete-time) $M/M/1$ queue as, contrary to the real $M/M/1$ queue, Z can take negative values.

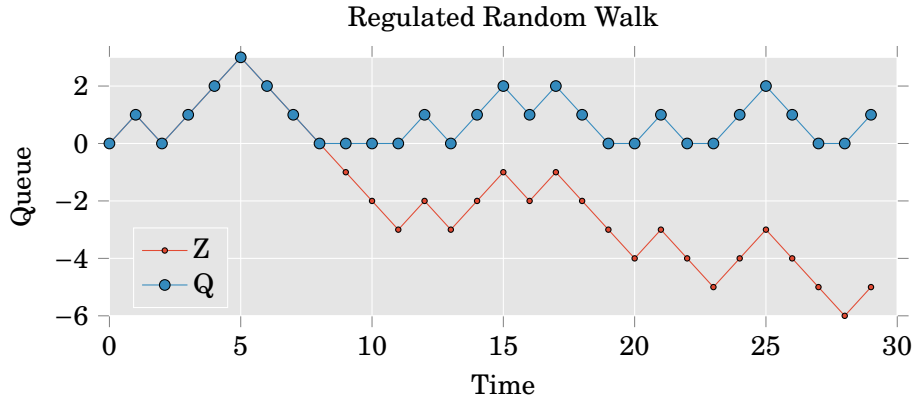


Figure 6: Another example of a reflected random walk.

1.7.3. *Show that when $n > m$ and $Z_0 = m$,*

$$\mathsf{P}(Z_k = n) = e^{-(\lambda+\mu)k}(\lambda k)^{n-m} \sum_{j=0}^{\infty} \frac{(\lambda\mu k^2)^j}{j!(n-m+j)!}.$$

The solution of the above exercise shows that there is no simple function by which we can compute the transient distribution of this simple random walk Z. Since a queueing process is typically a more complicated object (as we need to obtain L from Z via (1.7.2)), our hopes of finding anything simple for the transient analysis of the $M/M/1$ queue should not be too high. But the $M/M/1$ queue is about the simplest queueing system; other queueing systems are (much) more complicated. We therefore give up the analysis of the transient behavior of queueing systems and henceforth contend ourselves with the analysis of queueing systems in the limit as $t \to \infty$. The limiting random variable L is known as the *steady-state limit* of the sequence of random variables $\{L_k\}$, and the distribution of L is known as the *limiting distribution* or *stationary distribution* of $\{L_k\}$. Taking these limits warrants two questions: what type of limit is actually meant here, and what is the rate of convergence to this limiting situation? Here we sidestep all such fundamental issues, as the details require measure theory and more advanced probability theory than we can deal with here.

1.7.4. *[Not obligatory] Sketch a mathematical framework to answer these questions.*

To provide some intuition about the rate of convergence we consider now an example. Specifically, we consider the sequence of waiting times $\{W_{Q,k}\}$ to a limiting random variable W_Q, where $W_{Q,k}$ is constructed according to the recursion (1.5.5). Suppose that $X_k \sim U\{1,2,4\}$ and $S_k \sim U\{1,2,3\}$. Starting with $W_{Q,0} = 5$ we use (1.5.5) to compute the *exact* distribution of $W_{Q,k}$ for $k = 1,2,\ldots,20$, cf., the left panel in Fig. 7. We see that when $k = 5$, the 'hump' of $\mathsf{P}(W_{Q,5} = x)$ around $x = 5$ is due the starting value of $W_{Q,0} = 5$. However, for $k > 10$ the distribution of $W_{Q,k}$ hardly changes, at least not visually. Apparently, the convergence of the sequence of distributions of $W_{Q,k}$ is rather fast. In the middle panel we show the results of a set of *simulations* for increasing simulation length, up to $N = 1000$ samples. Here the *empirical distribution* for the simulation is defined as

$$\mathsf{P}(W_Q \le x) = \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{W_{Q,k} \le x},$$

where $W_{Q,k}$ is obtained by simulation. As should be clear from the figure, the simulated distribution also converges quite fast to some limiting function. Finally, in the right panel we

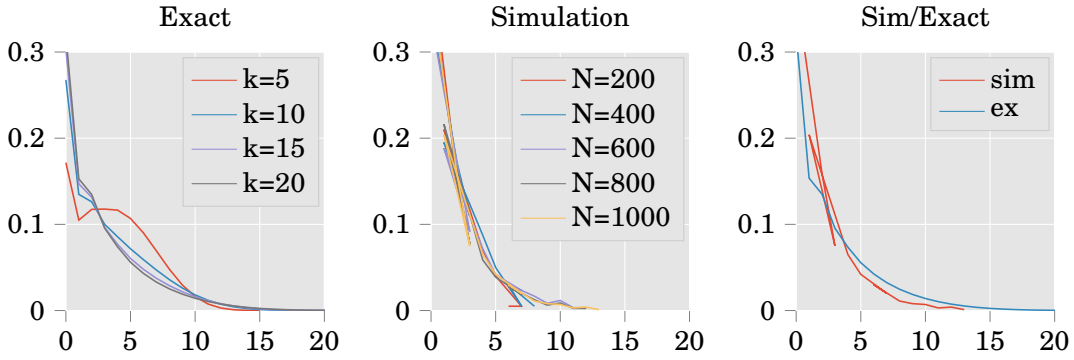


Figure 7: The density of $W_{Q,k}$ for $k = 5, 10, 15, 20$ computed by an exact method as compared the density obtained by simulation of different run lengths $N = 200, 400, \ldots, 1000$. The right panel compares the exact density of $W_{Q,20}$ to the density obtained by simulation for $N = 1000$.

compare the densities as obtained by the exact method and simulation with $n = 1000$. Clearly, for all practical purposes, these densities can be treated as the same.

The combination of the fast convergence to the steady-state situation and the difficulties with the transient analysis validates, to some extent, that most queueing theory is concerned with the analysis of the system in *stationarity*. The study of queueing systems in stationary state will occupy us for the rest of the book.

1.7.5. *Suppose that* $X_k \in \{1,3\}$ *such that* $\mathsf{P}(X_k = 1) = \mathsf{P}(X_k = 3)$ *and* $S_k \in \{1,2\}$ *with* $\mathsf{P}(S_k = 1) = \mathsf{P}(S_k = 2)$. *Write a computer program to see how fast the distributions of* $W_{Q,k}$ *converge to a limiting distribution function.*

1.7.6. *Validate the results of Fig. 7 with simulation.*

Hints

h.1.7.1. Note first that from the expression for Z_k, $a_k - c_k = Z_k - Z_{k-1}$. Use this to get $L_k = [L_{k-1} + Z_k - Z_{k-1}]^+$. Subtract Z_k from both sides, use recursion and use subsequently,

$$\begin{aligned} &\max\{\max\{a,b\},c\} = \max\{a,b,c\}, \\ &L_0 = Z_0, \\ &\max\{-a,-b\} = -\min\{a,b\}. \end{aligned}$$

h.1.7.2. Use 1.2.17.

Solutions

s.1.7.1. Note first that from the expression for Z_k, $a_k - c_k = Z_k - Z_{k-1}$. Using this in the recursion for L_k, we get

$$L_k = [L_{k-1} + Z_k - Z_{k-1}]^+,$$

thus,

$$L_k - Z_k = \max\{L_{k-1} - Z_{k-1}, -Z_k\}.$$

From this, using recursion and the hints, we see that

$$\begin{aligned} L_k - Z_k &= \max\{\max\{L_{k-2} - Z_{k-2}, -Z_{k-1}\}, -Z_k\} \\ &= \max\{L_{k-2} - Z_{k-2}, -Z_{k-1}, -Z_k\} \\ &= \max\{L_0 - Z_0, -Z_1, \ldots, -Z_k\} \\ &= \max\{0, -Z_1, \ldots, -Z_k\} \\ &= -\min\{0, Z_1, \ldots, Z_k\}. \end{aligned}$$

For further discussion, if you are interested, see Baccelli and Massey [1988].

s.1.7.2. Write a for the common random variable. Then

$$M_a(s) = \mathsf{E}\left[e^{sa}\right] = \sum_{j=0}^{\infty} e^{sj} e^{-\lambda} \lambda^j / j! = e^{\lambda(e^s - 1)}.$$

Then with $X = \sum_{j=1}^{k} a_j$, and by independence,

$$M_X(s) = (M_a(s))^k = e^{\lambda k(e^s - 1)}.$$

Hence, $X \sim P(\lambda k)$.

s.1.7.3.

$$\begin{aligned} \mathsf{P}(Z_k = n) &= \mathsf{P}\left(m + N_{\lambda k} - N_{\mu k} = n\right) \\ &= \mathsf{P}\left(N_{\lambda k} - N_{\mu k} = n - m\right) \\ &= \sum_{j=0}^{\infty} \mathsf{P}\left(N_{\lambda k} = n - m + j, N_{\mu k} = j\right) \\ &= \sum_{j=0}^{\infty} e^{-\lambda k} \frac{(\lambda k)^{n-m+j}}{(n-m+j)!} e^{-\mu k} \frac{(\mu k)^j}{j!} \\ &= e^{-(\lambda+\mu)k} (\lambda k)^{n-m} \sum_{j=0}^{\infty} \frac{(\lambda \mu k^2)^j}{j!(n-m+j)!}. \end{aligned} \tag{1.7.4}$$

s.1.7.4. The *long-run limiting behavior* of a queueing system (i.e., the first question) is an important topic by itself. The underlying question is what happens if we simulate the system for a long time. For instance, does there exist a random variable L such that $L_k \to L$ in some sense? The answer to this question is in the affirmative, provided some simple stability conditions are satisfied, see Section 2.1. However, it requires a considerable amount of mathematics to make this procedure precise. To sketch what has to be done, first, we need to define $\{L_k\}$ as random variables in their own right. Note that up to now we just considered each L_k as a *number*, i.e., a measurement or simulation of the queue length time of the kth period. Defining L_k as a random variable is not as simple as the definition of, for instance, the number of arrivals $\{a_k\}$; these random variables can be safely *assumed* to be i.i.d. However, the queue lengths $\{L_k\}$ are certainly not i.i.d., but, as should be apparent from (1.5.7), they are *constructed* in terms of recursions. Next, based on these recursions, we need to show that the sequence of distribution functions $\{G_k\}$ associated with the random variables $\{L_k\}$ converges to some limiting distribution function G, say. Finally, it is necessary to show that it is possible to construct a random variable L that has G as its distribution function. In this sense, then, we can say that $L_k \to L$.

s.1.7.5. Here is an example with Python. I compute the difference, i.e., the Kolmogorov-Smirnov statistic, between the distributions of $W_{Q,k-1}$ and $W_{Q,k}$,

$$\max_x\{|\mathsf{P}\left(W_{Q,k} \le x\right) - \mathsf{P}\left(W_{Q,k-1} \le x\right)|\},$$

for x in the support of $W_{Q,k}$.

The code can be found in the `exact` function in the file `waiting_time_simulation.py` at my github repo.

If you make a plot, you will see that after some 10 customers the distribution hardly changes any further.

s.1.7.6. The code is in the file `waiting_time_simulation.py` at my github repo.

1.8 OLD EXAM QUESTIONS

1.8.1 *Multiple-choice Questions*

The questions in this section are actually claims that are either true of false. It is up to you to decide which of the two alternatives is correct. A (in)correct answers (costs) earns you a point.

1.8.1. *[201703] If the random variable $X \sim \text{Exp}(\lambda)$, then*

$$\mathsf{E}\left[X^2\right] = \frac{1}{\lambda^2}.$$

1.8.2. *[201703] If $X \sim \text{Exp}(\lambda)$ and $Y \sim \text{Exp}(\mu)$ and X and Y are independent, then*

$$Z = \max\{X, Y\} \sim \text{Exp}(\lambda + \mu).$$

1.8.3. *[201703] If the interarrival times $\{X_k\}$ are i.i.d. and exponentially distributed with mean $1/\lambda$, and $A_k = \sum_{i=1}^{k} X_i$, then*

$$\mathsf{P}(A_{k+1} \le t) = -\frac{(\lambda t)^k}{k!} e^{-\lambda t} + \mathsf{P}(A_k \le t).$$

1.8.4. *[201703] Assume that $N_a(t) \sim P(\lambda t)$, $N_s(t) \sim P(\mu t)$ and independent. Then,*

$$\mathsf{P}(N_a(t) + N_s(t) = n) = e^{-(\mu+\lambda)t} \sum_{i=0}^{n} \frac{(\mu t)^{n-i}}{(n-i)!} \frac{(\lambda t)^i}{i!}.$$

1.8.5. *[201703] If $N(t) \sim P(\lambda t)$, then $\mathsf{E}\left[N^2\right] = (\lambda t)^2$.*

1.8.6. *[201703] For the $M/D/1$ queue, job service times are exponentially distributed and the interarrival times are deterministic.*

1.8.7. *[201703] A machine can switch on and off. If the queue length hits N, the machine switches on, and if the system becomes empty, the machine switches off. Let $I_k = 1$ if the machine is on in period k and $I_k = 0$ if it is off, let L_k be the number of items in the system at the end of period k, then,*

$$I_{k+1} = \begin{cases} 1 & \text{if } L_k \ge N, \\ I_k & \text{if } 0 < L_k < N, \\ 0 & \text{if } L_k = 0, \end{cases}$$

$$\begin{aligned} I_{k+1} &= \mathbb{1}_{L_k \geq N} + I_k \mathbb{1}_{0<L_k<N}, \\ d_k &= \min\{L_{k-1}, c_k\}, \\ L_k &= L_{k-1} - (1-I_k)d_k + a_k. \end{aligned}$$

Assume that $I_0 = 0$ at time $k = 0$.

It is true that the above recursions model this queueing system.

1.8.8. *[201703] For the G/G/1 queue the following recursion is true:*

$$\begin{aligned} W_k &= [W_{k-1} - X_k]^+ + S_k, \\ D_k &= A_k + W_k. \end{aligned}$$

1.8.9. *[201703] For the G/G/1 queue, suppose that the interarrival times $X_k \in \{1,3\}$ such that $\mathsf{P}(X_k = 1) = 1/5$ (hence, $\mathsf{P}(X_k = 3) = 4/5$) and the service times $S_k \in \{1,2\}$ with $\mathsf{P}(S_k = 1) = 1/3$ (hence, $\mathsf{P}(S_k = 2) = 2/3$). If $W_{L,0} = 3$,*

$$\mathsf{P}\left(W_{L,1} = 1\right) = \frac{4}{15}.$$

1.8.10. *[201703] Consider the random walk*

$$Z(t) = Z(0) + N_\lambda(t) - N_\mu(t),$$

where the arrival process is a Poisson process $N_\lambda(t)$ and the departure process is a Poisson process $N_\mu(t)$.

$$\mathsf{P}(m)Z(t) = n = \sum_{k=0}^{\infty} e^{-\mu t} \frac{(\mu t)^{k-n+m}}{(k-n+m)!} e^{-\lambda t} \frac{(\lambda t)^k}{k!},$$

where $\mathsf{P}(m)\cdot$ means that $Z(0) = m$.

1.8.11. *[201704] If $X \sim \mathrm{Exp}(\lambda)$, $S \sim \mathrm{Exp}(\mu)$ and independent, then*

$$\begin{aligned} \mathsf{P}(X \leq S) &= \mathsf{E}[\mathbb{1}_{X \leq S}] \\ &= \int_0^\infty \int_0^\infty \mathbb{1}_{x \leq y} f_{X,S}(x,y)\, dy\, dx \\ &= \lambda\mu \int_0^\infty \int_0^\infty \mathbb{1}_{x \leq y} e^{-\lambda x} e^{-\mu y}\, dy\, dx \\ &= \lambda\mu \int_0^\infty e^{-\mu y} \int_0^y e^{-\lambda x}\, dx\, dy \\ &= \mu \int_0^\infty e^{-\mu y} e^{-\lambda y}\, dy \\ &= \mu \int_0^\infty e^{-(\lambda+\mu)y}\, dy \\ &= \frac{\mu}{\lambda+\mu}. \end{aligned}$$

1.8.12. *[201704] For the G/G/1 it is true that*

$$W_k = [W_{k-1} - X_k + S_k]^+.$$

1.8.13. *[201704]For the G/G/1 queue the virtual waiting time process $\{V(t), t \geq 0\}$ satisfies*

$$V(t) = [V(A_{A(t)}) + (A_{A(t)} - t)]^+.$$

1.8.14. *[201704]For the G/G/1 queue, if $\mathsf{E}[B]$ is the expected busy time and $\mathsf{E}[I]$ is the expected idle time, then*

$$\mathsf{E}[B] = \frac{\rho}{1-\rho}\mathsf{E}[I].$$

1.8.15. *[201704]Consider a paint factory which contains a paint mixing machine that serves two classes of jobs, A and B. The processing times of jobs of types A and B are constant and require t_A and t_B hours. The job arrival rate is λ_A for type A and λ_B for type B jobs. It takes a setup time of S_{ij} hours to clean the mixing station when changing from paint type i to type j.*

The linear program below can be used to determine the minimal batch sizes. To keep the system (rate) stable,

$$\textit{minimize } T$$

such that

$$\begin{aligned} T &= k_A t_A + S_{AB} + k_B t_B + S_{BA}, \\ \lambda_A T &< k_A, \\ \lambda_B T &< k_B. \end{aligned}$$

1.8.16. *[201704]For the G/G/1 queue the stationary distribution of the waiting times at arrival times is equal to the empirical distribution*

$$\mathsf{P}(W \leq x) = \frac{1}{n}\sum_{k=1}^{n} \mathbb{1}_{W_k \leq x}.$$

1.8.17. *[201802] If $N(t)$ is Poisson distributed with parameter λt, i.e., $N(t) \sim P(\lambda t)$, the variance $\mathsf{V}[N(t)] = \lambda t$.*

1.8.18. *[201802]*

$$\sum_{n=0}^{\infty} n\frac{\lambda^n}{n!} = \lambda e^{\lambda}.$$

1.8.19. *[201802] Let N be a Poisson process with rate λ. Then,*

$$\{N(0,s] + N(s,t] = 1\} \cap \{N(0,t] = 1\} = \{N(0,s] = 1\}.$$

1.8.20. *[201802] If $N \sim P(\lambda)$,*

$$M_N(s) = \mathsf{E}\left[e^{sN}\right] = \exp(\lambda(s-1)).$$

1.8.21. *[201802] Suppose $N \sim P(\lambda)$ and N_1 is a random variable obtained by 'thinning' N with Bernoulli random variables with success probability p. The following reasoning is correct:*

$$\begin{aligned}
\mathsf{P}(N_1 = k) &= \sum_{n=k}^{\infty} \mathsf{P}(N_1 = k, N = n) = \sum_{n=k}^{\infty} \mathsf{P}(N_1 = k \,|\, N = n)\mathsf{P}(N = n) \\
&= \sum_{n=k}^{\infty} \mathsf{P}(N_1 = k \,|\, N = n) e^{-\lambda} \frac{\lambda^n}{n!} \\
&= \sum_{n=k}^{\infty} \binom{n}{k} p^k (1-p)^{n-k} e^{-\lambda} \frac{\lambda^n}{n!} \\
&= e^{-\lambda} \sum_{n=k}^{\infty} \frac{p^k (1-p)^{n-k}}{k!(n-k)!} \lambda^n = e^{-\lambda} \frac{(\lambda p)^k}{k!} \sum_{n=k}^{\infty} \frac{(\lambda(1-p))^{n-k}}{(n-k)!} \\
&= e^{-\lambda} \frac{(\lambda p)^k}{k!} \sum_{n=0}^{\infty} \frac{(\lambda(1-p))^n}{n!} = e^{-\lambda} \frac{(\lambda p)^k}{k!} e^{\lambda(1-p)} \\
&= e^{-\lambda p} \frac{(\lambda p)^k}{k!}.
\end{aligned}$$

1.8.22. *[201802] When many unrelated jobs arrive at a queueing system, like patients at a hospital, or customers at a shop, it is reasonable to model the interarrival times as exponentially distributed with a mean that is constant during short periods, (e.g. 10 minutes or 30 minutes, depending on the relevant context).*

1.8.23. *[201802] If X is exponentially distributed with rate λ,*

$$\mathsf{E}[X] = \int_0^{\infty} \lambda e^{-\lambda t}\, dt.$$

1.8.24. *[201802] Given a non-negative random variable B taking values in $\mathbb{N}$ and with $F(i) = \mathsf{P}(B \le i)$, then, $\mathsf{E}[B] = \sum_{i=0}^{\infty} iF(i)$.*

1.8.25. *[201802] In the $D/M/1$ jobs have deterministic service times.*

1.8.26. *[201802] In a discrete-time queueing system, when job arrivals in period k cannot be served in period k, then $d_k = \min\{Q_{k-1}, c_k\}$, $Q_k = Q_{k-1} - d_k + a_k$.*

1.8.27. *[201802] For a continuous-time queueing system, if S_2 is the service time of job 2 and X_3 is the time between jobs 2 and 3, and S_2 and X_3 are independent, then*

$$\mathsf{P}(S_2 = 1, X_3 = 3) = \mathsf{P}(S_2 = 1)\mathsf{P}(X_3 = 3).$$

1.8.28. *[201802] For a single-server queueing system that starts empty, the number $L(t)$ of jobs in the system at time t satisfies*

$$L(t) = \sum_{k=1}^{\infty} \left[\mathbb{1}_{A_k \le t} - \mathbb{1}_{D_k \le t} \right].$$

1.8.29. *[201802] For three numbers a, b, c, $\max\{a, \max\{b, c\}\} = \max\{a, b, c\}$.*

1.8.30. *[201802] If a r.v. $X \sim \mathrm{Exp}(\lambda)$, i.e., exponentially distributed with mean λ^{-1}, then the following shows that X has the memoryless property:*

$$\mathsf{P}(X > t + h | X > t) = \frac{\mathsf{P}(X > t + h, X > t)}{\mathsf{P}(X > t)} = \frac{\mathsf{P}(X > t + h)}{\mathsf{P}(X > t)} = \frac{e^{-\lambda(t+h)}}{e^{-\lambda t}} = e^{-\lambda h} = \mathsf{P}(X > h).$$

1.8.31. *[201803] Consider the G/G/1 queue. Under the 'shortest processing time first' scheduling rule predictions of the finish times (i.e., quoting due dates) are more accurate than under the 'first-in-first-out' rule.*

1.8.32. *[201803] Consider a single-server that serves two parallel queues A and B. Each queue receives a minimal service capacity every period. Reserved capacity unused for one queue can be used to serve the other queue. Any extra capacity beyond the reserved capacity is given to queue A with priority. The following set of recursions suffices to simulate this situation,*

$$\begin{aligned} c_{k,A} &= \min\{Q_{k-1,A}, r_A\}, \\ d_{k,A} &= \min\{Q_{k-1,A}, c_k - c_{k,A}\}, \\ Q_{k,A} &= Q_{k-1,A} - d_{k,A} + a_{k,A}, \\ d_{k,B} &= \min\{Q_{k-1,B}, c_k - d_{k,A}\}, \\ Q_{k,B} &= Q_{k-1,B} - d_{k,B} + a_{k,B}, \end{aligned}$$

where r_A, r_B are the reserved capacities for each queue, and c_k the total service capacity available in time k and such that $c_k \geq r_A + r_B$.

1.8.33. *[201803] The Kolmogorov-Smirnov statistic between the distributions of the random variables $W_{Q,k}$ and $W_{Q,k-1}$ is given by*

$$\max_x \{\mathsf{P}\left(W_{Q,k} \leq x\right) - \mathsf{P}\left(W_{Q,k-1} \leq x\right)\}.$$

1.8.34. *[201804] Assume that the interarrival times $\{X_i\}$ are i.i.d. and $X_i \sim \text{Exp}(\lambda)$. Let $A_i = X_1 + X_2 + \cdots + X_i = \sum_{k=1}^{i} X_k$ with $i \geq 1$. Then,*

$$\mathsf{E}[A_i] = \frac{i}{\lambda}.$$

1.8.35. *[201804] We have a discrete-time queueing system with a server with capacity c per period. The server can serve 2 more jobs when the queue length is 24 or longer, and 1 less when the queue length is less than 12. We can use the following construction to simulate this queueing process:*

$$\begin{aligned} c_n &= c + 2\,\mathbb{1}_{Q_{n-1} \geq 24} - \mathbb{1}_{Q_{n-1} \leq 12}, \\ d_n &= \min\{Q_{n-1}, c_n\}, \\ Q_n &= Q_{n-1} + a_n - d_n, \end{aligned}$$

where the notation is the same as in the book.

1.8.36. *[201804] We have a discrete-time queueing system with a server with capacity c per period. The server can serve 2 more jobs when the queue length is 24 or longer, and 1 less when the queue length is less than 12. The following formula computes the number of periods in which the queue exceeds 30 for a simulation that starts at period 1 and stops at period N,*

$$N - \sum_{n=1}^{N} \mathbb{1}_{Q_n \leq 30}.$$

1.8.37. *[201804] The following is correct:*

$$(\lambda t)^k (\mu t)^{k+m-n} = (\lambda/\mu)^{(n-m)/2} (t\sqrt{\lambda\mu})^{2k+m-n}.$$

1.8.38. *[201804] Consider the G/G/1 queue. The total amount of service that arrived during the arrival times of the first and the $n+1$th job, i.e., between $[A_1, A_{n+1})$, is $\sum_{i=1}^{n} S_i$. Thus, the fraction of time that the server has been busy during $[A_1, A_{n+1})$ is*

$$\frac{\sum_{i=1}^{n} S_i}{A_{n+1} - A_1} = \frac{\sum_{i=1}^{n} S_i}{\sum_{i=1}^{n+1} X_i - X_1} = \frac{\sum_{i=1}^{n} S_i}{\sum_{i=2}^{n+1} X_i}.$$

1.8.39. *[201807] Let $\{N(t)\}$ denote the Poisson process with rate λ, and write $N(s,t]$ for the number of arrivals in the interval $(s,t]$. Claim: the following holds for $h \ll 1$,*

$$\begin{aligned}
\mathsf{P}(N(t+h) = n | N(t) = n) &= \mathsf{P}(N(t+h) = n, N(t) = n) / \mathsf{P}(N(t) = n) \\
&= \mathsf{P}(N(t, t+h] = 0, N(t) = n) / \mathsf{P}(N(t) = n) \\
&= \mathsf{P}(N(t, t+h] = 0) \mathsf{P}(N(t) = n) / \mathsf{P}(N(t) = n), \\
&= \mathsf{P}(N(t, t+h] = 0) \\
&= \mathsf{P}(N(0, h] = 0) \\
&= e^{-\lambda h} (\lambda h)^0 / 0! \\
&= e^{-\lambda h} = 1 - \lambda h + o(h).
\end{aligned}$$

1.8.40. *[201807] Assume that the arrival rate of customers is constant in a given half hour. Claim: it is reasonable to model the interarrival times of customers at call centers or hospitals as exponentially distributed.*

1.8.41. *[201807] If X is an exponentially distributed random variable with mean $1/\lambda$ and Y exp. dist. with mean $1/\mu$, and X and Y are independent. Claim:*

$$\mathsf{P}(X \le Y) = 1 - \frac{\lambda}{\lambda + \mu}.$$

1.8.42. *[201807] Let S be a continuous random variable with survivor function G, density f, and finite second moment. Claim:*

$$\begin{aligned}
\int_0^\infty y G(y)\, dy &= \int_0^\infty y \int_y^\infty f(x)\, dx\, dy = \int_0^\infty y \int_0^\infty 1\{y \le x\} f(x)\, dx\, dy \\
&= \int_0^\infty f(x) \int_0^\infty y 1\{y \le x\}\, dy\, dx = \int_0^\infty f(x) \frac{x^2}{2}\, dx = \frac{\mathsf{E}[S^2]}{2}.
\end{aligned}$$

1.8.43. *[201902] Let $N \sim P(\lambda)$. Then its variance is λ.*

1.8.44. *[201902] Let $N \sim P(\lambda)$. Then its SCV is $1/\lambda^2$.*

1.8.45. *[201902] Let $f(x) = 10^6 x^2$. Then $f(x) = o(x)$.*

1.8.46. *[201902] Let N be a random variable taking values in $\mathbb{N}$. Let $\{Z_i\}$ be a set of i.i.d. Bernoulli random variables, and independent of N. Let $Y = \sum_{i=1}^{N} Z_i$. Then, for $s > 0$,*

$$\mathsf{E}\left[e^{sY}\right] = \sum_{n=0}^{\infty} \mathsf{E}\left[e^{s\sum_{i=1}^{n} Z_i}\right] \mathsf{E}[\mathbb{1}_{N \le n}].$$

1.8.47. *[201902] Let Y be a random variable, exponentially distributed with rate $\lambda > 0$. Then $\mathsf{V}[Y] = 1/\lambda^2$.*

1.8.48. *[201902] Let A_i be the arrival time of customer i and set $A_0 = 0$. Assume that the inter-arrival times $\{X_i\}$ are i.i.d. with exponential distribution with mean $1/\lambda$ for some $\lambda > 0$. It it true that for the moment-generating function $M_{A_i}(t)$*

$$M_{A_i}(t) = \mathsf{E}\left[e^{tA_i}\right] = \mathsf{E}\left[\exp\left(t\sum_{k=1}^{i-1} X_k\right)\right].$$

1.8.49. *[201902] Let $S \sim U[0,7]$ and $X \sim U[0,10]$, where $U[I]$ stands for the uniform distribution concentrated on the interval I, and S and X independent. Then the joint density function is equal to*

$$f_{XS}(x,s) = \mathbb{1}_{0\le x\le 10, 0\le s\le 7}.$$

1.8.50. *[201902] Consider a check-in desk at an airport. There is one desk that is dedicated to business customers. However, when it is idle (i.e., no business customer in service or in queue), this desk also serves economy class customers. The other c desks are reserved for economy class customers. The queueing process as perceived by the economy class customers can be modeled as an $M/M/(c+1)$ queue.*

1.8.51. *[201902] In the $M/G/\infty$ queue jobs never spend time in queue.*

1.8.52. *[201902] The following Python code to simulate a queueing system in discrete time will work as is:*

```
import numpy as np

Q = np.zeros_like(a)
d = np.zeros_like(a)
Q[0] = 10 # initial queue length

for k in range(1, len(a)):
 d[k] = min(Q[k - 1], c[k])
 Q[k] = Q[k - 1] - d[k] + a[k]
```

1.8.53. *[201902] We have a queueing system in discrete time. Take $c_k = c\,\mathbb{1}_{Q_{k-1}>t}$ as service capacity in period k. If $c < t$, and $Q_0 > 0$, then $Q_k > 0$ for all k.*

1.8.54. *[201902] With the recursion below we can simulate a queueing system in discrete time such that the arrivals in period k can also be served in period k. (Note: the question is not whether this code will run as is; it will not.)*

```
for k in range(1, len(a)):
 Q[k] = max(Q[k - 1] - c[k] + a[k], 0)
 d[k] = Q[k - 1] + a[k] - Q[k]
```

1.8.55. *[201902] Take $d_k = \min\{Q_{k-1} + a_k, c_k\}$, and assume that jobs are served in FIFO sequence. The largest possible waiting time $W_{+,k}$ for a job arriving in period k is given by*

$$W_{+,k} := \min\left\{m : \sum_{i=k}^{k+m} c_i \ge Q_{k-1} + a_k\right\}.$$

1.8.56. *[201902] Consider a random walk $Z_k = Z_{k-1} + X_k$ with $\{X_k\}$ a process of i.i.d. random variables such that $\mathsf{P}(X_k = 1) = 1 - \mathsf{P}(X_k = -1) = p \in (0,1)$. Then, for large n and $\alpha = 2$*

$$\mathsf{P}\left(Z_n > np + \alpha\sqrt{np(1-p)}\right) \approx 2.5\%.$$

1.8.57. *[201903] Let $\{X_k, k = 1,2,\ldots\}$ be a set of i.i.d. exponentially distributed random variables with mean $1/\lambda$, and $\{N(t), t \geq 0\}$ with $N(0) = 0$ the associated counting process. Then $\mathsf{P}(X_1 \leq s, X_2 \leq t) = \mathsf{P}(N(s+t) = 2)$.*

1.8.58. *[201903] Let $X \sim \text{Exp}(\lambda)$ and $M_X(s)$ the associated moment-generating function. Then $M_X(s) < \infty$ for all $s \in \mathbb{R}$.*

1.8.59. *[201903] We consider a discrete-time queueing system with a_k the number of arrivals in period k, and c_k the service capacity. Let*

$$d_k = \min\{Q_{k-1}, c_k\}, \qquad Q_k = Q_{k-1} - d_k + a_k. \tag{1.8.1}$$

Take a_k Poisson distributed with $\lambda = 2$, i.e., $a_k \sim P(2)$, and $c_k \sim P(1)$ and initialize $Q_0 = 0$. Then $\mathsf{P}(Q_{10000} \geq 100) \leq 1/2$.

1.8.60. *[201903] A single-server queueing station processes customers. At the start of a period the server capacity is chosen, so that for period k the capacity is c_k. Demand that arrives in a period can be served in that period. It costs β per unit time per unit processing capacity to operate the machine, i.e., to have it switched on. There is also a cost h per unit time per job in the system. The total cost up to some time T is given by $\beta\sum_{k=1}^{T} c_k$.*

1.8.61. *[201903] In python you need to set the seed of the random number generator to a fixed value in order to obtain the same random numbers for various simulation runs.*

1.8.62. *[201903] Consider the G/G/1 queue in continuous time with inter-arrival times X_k, service times S_k, arrival times A_k and departure times D_k. We can compute the waiting times and sojourn times with the following recursion:*

$$W_{Q,k} = [W_{k-1} - X_k]^+, \qquad W_k = W_{Q,k} + S_k = [W_{k-1} - X_k]^+ + S_k. \tag{1.8.2}$$

1.8.63. *[201904] One server serves two queues and has capacity c per period available. We divide the capacity c over the queues in proportion to the queue lengths L^i_{k-1}, $i = 1,2$. The following implements this rule:*

$$c^1_k = \left\lfloor \frac{L^1_{k-1}}{L^1_{k-1} + L^2_{k-1}} c + \frac{1}{2} \right\rfloor, \qquad c^2_k = c - c^1_k,$$

where c^i_k be the capacity allocated to queue i in period k and we include the rounding to prevent the loss of capacity.

1.8.64. *[201904] When $X \sim \text{Exp}(\lambda)$ its SCV is larger than 1.*

1.8.65. *[201904] For the G/G/1 queue the waiting time satisfies the recursion*

$$W_{Q,k} = \max\{W_{Q,k-1} + S_k - X_k, 0\}.$$

1.8.66. *[201907] $e^x = 1 + x^2 + o(x^2)$.*

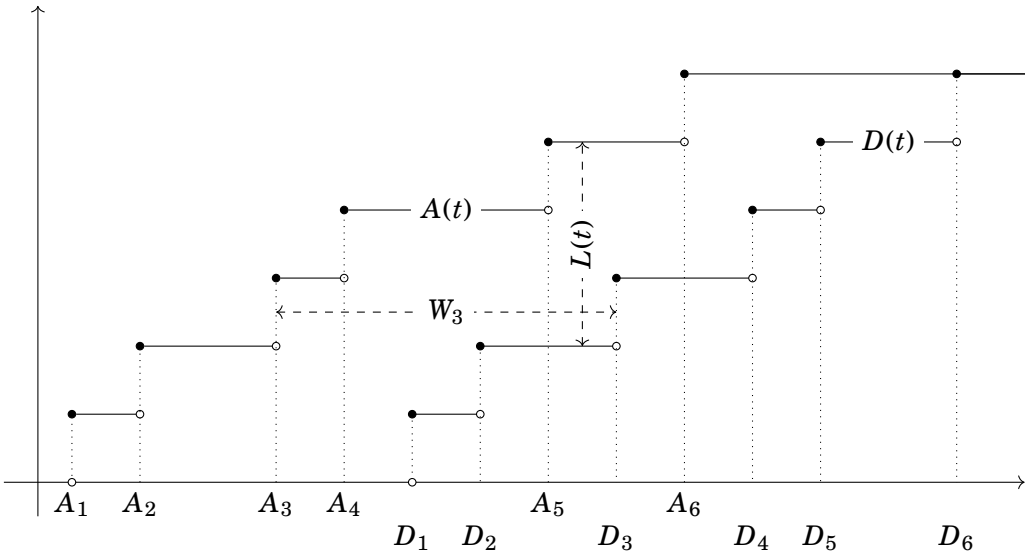
1.8.67. *[201907] A machine produces items to serve customer demand. A fraction p_k of the items produced in period k turns out to be faulty, and has to be made anew. The following set of recursions models the queue of customers waiting to be served from the machine:*

$$d_k = \min\{L_{k-1}, c_k\},$$
$$L_k = L_{k-1} - d_k(1-p_k) + a_k.$$

1.8.68. *[201907] The departure process $\{D(t)\}$ can be computed from the set $\{D_k\}$ of departure times according to:*

$$D(t) = \sum_{k=1}^{\infty} \mathbb{1}_{D_k \le t}.$$

1.8.69. *[201907] The waiting time of the third job is correctly represented in the figure below.*



1.8.2 *Open Questions*

1.8.70. *[201704] One server serves n queues in parallel. The server has capacity $c_k \in \mathbb{R}_+$ for day k. The amount of work that arrives on day k for queue i is given by $a_k^i \in \mathbb{R}_+$. Jobs arriving on day k cannot be served on day k. Queue i receives service capacity in proportion to its queue length. Derive a set of recursions to compute the queue lengths $Q_k^i \in \mathbb{R}_+$ for days $k = 1, 2, \ldots$.*

1.8.71. *[201704] Provide a real-world example for this queueing model.*

1.8.72. *[201704] For the situation of the previous question, find a recursion to compute a tight upper bound on the time the arriving work a_k^i spends in the system.*

1.8.73. *[201807] Consider a network of two stations in tandem. Jobs only arrive at station 1. Jobs arriving in period k are only accepted when the total number of jobs in the system is less than K at the start of the period. Otherwise, all arriving jobs are rejected. Provide a set of recursions to simulate the queue length process at both stations in discrete time.*

1.8.74. *[201807] We have a queueing system that operates under the following set of recursions:*

$$c_n = 3\,\mathbb{1}_{Q_n \in [5,10]} + \mathbb{1}_{Q_n \notin [5,10]},$$

$$d_n = \min\{Q_{n-1} + a_n, c_n\}$$
$$Q_n = Q_{n-1} + a_n - d_n,$$

where the arrivals a_n are i.i.d. and have Poisson distribution with parameter $\lambda = 2$, c_n is the service capacity in period n and d_n corresponds to the number of departures. Why is this queueing system unstable?

1.8.75. *[201807] What is a disadvantage of using simulation to analyze queueing systems?*

1.8.76. *[201904] A machine produces items, but a fraction p of the items does not meet the quality requirements after the first pass at the server, but it requires a second pass. Assume that the repair of a faulty item requires half of the work of a new job, and that the faulty jobs are processed with priority over the new jobs. Also assume that faulty items do not need more than one repair (hence, faulty items that are repaired cannot be faulty anymore). Make a set of discrete-time recursions to analyze this case.*

1.8.77. *[201907] Why did we discuss the transient behavior of the random walk in a course on queueing theory?*

Solutions

s.1.8.1. Answer = B.

$$\mathsf{E}\left[X^2\right] = \frac{2}{\lambda^2}.$$

s.1.8.2. Answer = B.
$Z = \min\{X, Y\} \sim \text{Exp}(\lambda + \mu)$.

s.1.8.3. Answer = A.

I decided to withdraw this question. The initial wording of the question was like this: 'If the interarrival times $\{A_k\}$ are i.i.d. and exponentially distributed with mean $1/\lambda$, then…'. However, in the book we use the notation A_k for arrival times, not interarrival times. During the exam I changed the word 'interarrival' by 'arrival', hoping that it would clarify the meaning of the A_k, but then there is another problem, namely the arrival times are not i.i.d. Hence, there was no way in which the problem could be interpreted during the exam in an unambiguous way.

s.1.8.4. Answer = A.

s.1.8.5. Answer = B.

Set $t = 1$ for notational simplicity. Then

$$M(s) = \mathsf{E}\left[e^{sN}\right] = e^{-\lambda} \sum_{n=0}^{\infty} \frac{\lambda^n}{n!} = \exp\{\lambda(e^s - 1)\}.$$

Then,

$$M'(s) = \frac{\mathrm{d}}{\mathrm{d}t} M(s) = \exp\{\lambda(e^s - 1)\}\lambda e^s,$$

and

$$M''(s) = \exp\{\lambda(e^s - 1)\}(\lambda^2 e^{2s} + \lambda e^s).$$

Finally, $\mathsf{E}\left[N^2\right] = M''(0) = \lambda^2 + \lambda$.

s.1.8.6. Answer = B.

The interarrival times are exponentially distributed and the service times are deterministic.

s.1.8.7. Answer = B. It should be this: $d_k = I_k \min\{L_{k-1}, c_k\}$, $L_k = L_{k-1} - d_k + a_k$.

s.1.8.8. Answer = A.

s.1.8.9. Answer = A.

$$\mathsf{P}\left(W_{L,1} = 1\right) = \mathsf{P}(S_0 = 1, X_1 = 3) = \frac{1}{3}\frac{4}{5} = \frac{4}{15}.$$

I asked < 1/2 in case you would use your calculator which might have resulted in rounding errors. Like this, I was on the safe side.

s.1.8.10. Answer = A.

s.1.8.11. Answer = B.

$$\begin{aligned}
\mathsf{P}(X \le S) &= \mathsf{E}[\mathbb{1}_{X \le S}] \\
&= \int_0^\infty \int_0^\infty \mathbb{1}_{x \le y} f_{X,S}(x,y)\,\mathrm{d}y\,\mathrm{d}x \\
&= \lambda\mu \int_0^\infty \int_0^\infty \mathbb{1}_{x \le y} e^{-\lambda x} e^{-\mu y}\,\mathrm{d}y\,\mathrm{d}x \\
&= \lambda\mu \int_0^\infty e^{-\mu y} \int_0^y e^{-\lambda x}\,\mathrm{d}x\,\mathrm{d}y \\
&= \mu \int_0^\infty e^{-\mu y}(1 - e^{-\lambda y})\,\mathrm{d}y \\
&= \mu \int_0^\infty (e^{-\mu y} - e^{-(\lambda+\mu)y})\,\mathrm{d}y \\
&= \mu \int_0^\infty (e^{-\mu y} - e^{-(\lambda+\mu)y})\,\mathrm{d}y \\
&= 1 - \frac{\mu}{\lambda+\mu}.\text{‘}
\end{aligned}$$

s.1.8.12. Answer = B.

$$\begin{aligned}
W_{Q,k} &= [W_{k-1} - X_k]^+, \\
W_k &= W_{Q,k} + S_k = [W_{k-1} - X_k]^+ + S_k.
\end{aligned} \tag{1.8.3}$$

s.1.8.13. Answer = A.

s.1.8.14. Answer = A.

s.1.8.15. Answer = A.

s.1.8.16. Answer = B.

$$\mathsf{P}(W \le x) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{W_k \le x}. \tag{1.8.4}$$

s.1.8.17. Answer = A.

s.1.8.18. Answer = A.

s.1.8.19. Answer = B.

s.1.8.20. Answer = B.

s.1.8.21. Answer = A.

s.1.8.22. Answer = A.

s.1.8.23. Answer = B.

s.1.8.24. Answer = B.

s.1.8.25. Answer = B.

s.1.8.26. Answer = A.

s.1.8.27. Answer = A.

s.1.8.28. Answer = A.

s.1.8.29. Answer = A.

s.1.8.30. Answer = A.

s.1.8.31. Answer = B.

s.1.8.32. Answer = B. The following is correct:

$$\begin{aligned} c_{k,B} &= \min\{Q_{k-1,B}, r_B\}, \\ d_{k,A} &= \min\{Q_{k-1,A}, c_k - c_{k,B}\}, \\ Q_{k,A} &= Q_{k-1,A} - d_{k,A} + a_{k,A}, \\ d_{k,B} &= \min\{Q_{k-1,B}, c_k - d_{k,A}\}, \\ Q_{k,B} &= Q_{k-1,B} - d_{k,B} + a_{k,B}. \end{aligned}$$

s.1.8.33. Answer = B. It should be

$$\max_x \{|\mathsf{P}\left(W_{Q,k} \le x\right) - \mathsf{P}\left(W_{Q,k-1} \le x\right)|\}.$$

s.1.8.34. Answer = A.

s.1.8.35. Answer = B. It should be this:

$$\begin{aligned} c_n &= c + 2\,\mathbb{1}_{Q_{n-1} \ge 24} - \mathbb{1}_{Q_{n-1} < 12}, \\ d_n &= \min\{Q_{n-1}, c_n\}, \\ Q_n &= Q_{n-1} + a_n - d_n. \end{aligned}$$

Note the *less than*.

s.1.8.36. Answer = A.

$$N - \sum_{i=n}^{N} \mathbb{1}_{Q_n \le 30} = \sum_{i=n}^{N} (1 - \mathbb{1}_{Q_n \le 30}) = \sum_{i=n}^{N} \mathbb{1}_{Q_n > 30}.$$

s.1.8.37. Answer = A.

s.1.8.38. Answer = B. The point here is that it is not true in general. Only when $L(A_{n+1}) = 0$, it is true. Otherwise, if $A_{n+1} < D_n$, there is a queue, hence the total amount of work arrived is larger than what has been served. In fact, the quantity in the formula may exceed 1, in which case it cannot be a fraction of time.

s.1.8.39. Answer = A.

s.1.8.40. Answer = A.

s.1.8.41. Answer = B.

s.1.8.42. Answer = A.

s.1.8.43. Answer = A.

s.1.8.44. Answer = B. The SCV is $1/\lambda$.

s.1.8.45. Answer = A.

s.1.8.46. Answer = B.

$$\mathsf{E}\left[e^{sY}\right] = \sum_{n=0}^{\infty} \mathsf{E}\left[e^{s\sum_{i=1}^{n} Z_i}\right] \mathsf{E}[\mathbb{1}_{N=n}].$$

We discussed this and similar expressions in class, at least in two lectures.

s.1.8.47. Answer = A.

s.1.8.48. Answer = B. $A_i = A_{i-1} + X_i$. Hence,

$$M_{A_i}(t) = \mathsf{E}\left[e^{tA_i}\right] = \mathsf{E}\left[\exp\left(t \sum_{k=1}^{i} X_k\right)\right].$$

s.1.8.49. Answer = B.

$$f_{XS}(x,s) = f_X(x) \cdot f_S(s) = \frac{1}{10} \mathbb{1}_{0 \le x \le 10} \cdot \frac{1}{7} \mathbb{1}_{0 \le s \le 7}.$$

s.1.8.50. Answer = B.

s.1.8.51. Answer = A.

s.1.8.52. Answer = B. The vectors a and c are not given. See this:

```
>>> import numpy as np

>>> Q = np.zeros_like(a)
>>> d = np.zeros_like(a)
>>> Q[0] = 10 # initial queue length

>>> for k in range(1, len(a)):
...  d[k] = min(Q[k - 1], c[k])
...  Q[k] = Q[k - 1] - d[k] + a[k]
...
```

s.1.8.53. Answer = A.

s.1.8.54. Answer = A. This recursion is used to relate the queueing process to a random walk.

s.1.8.55. Answer = A.

The initial formulation of the question was like this: Take $d_k = \min\{Q_{k-1} + a_k, c_k\}$, and assume that jobs are served in FIFO sequence. The largest possible waiting time $W_{+,k}$ for the kth arriving job is given by

$$W_{+,k} := \min\left\{ m : \sum_{i=k}^{k+m} c_i \geq Q_{k-1} + a_k \right\}.$$

We removed this question from the exam as it was confusing. The problem was that the kth job need not arrive in period k.

The formulation in the question is adapted so that it can be used as an example.

s.1.8.56. Answer = B. Immediate consequence of the central limit theorem. It is in fact very interesting. It tells us that when $\mathsf{E}[X_k] < 0$ and Q_0 is very large, the relative variability of the queueing process decreases when n becomes large. To see this, observe that $\mathsf{E}[X_k] = p - (1-p) = 2p-1$ and $\mathsf{V}[X_k] = 1-(2p-1)^2 = 2p(1-2p)$. Then consider $(Z_n - n\,\mathsf{E}[X_k])/\sqrt{n\,\mathsf{V}[X_k]}$. From the central limit theorem we know that this random variable is normally distributed with mean 0 and $\sigma = 1$.

The statement in the question uses random variables X_k such that $\mathsf{P}(X_k = 1) = p = 1 - \mathsf{P}(X_k = 0)$.

s.1.8.57. Answer = B.

When $X_1 \leq s$ and $X_2 \leq t$ then still $N(s+t) = 3$ is possible. For $N(s+t) = 2$ it is necessary that $X_1 + X_2 + X_3 > s + t$.

s.1.8.58. Answer = B.

If $s > \lambda$ the integral does not converge.

s.1.8.59. Answer = B.

s.1.8.60. Answer = B.

The total cost is

$$\sum_{k=1}^{T} \left(\beta c_k + h Q_k \right).$$

s.1.8.61. Answer = A.

s.1.8.62. Answer = A.

s.1.8.63. Answer = A, 1.3.11. We introduce rounding to prevent the service of 'partial' customers. For instance, if the first queue contains 1 job, and the second 2, then without rounding we would server 2/3 customer of the second type.

Note that the case with empty queues does not lead to a problem. When there are no jobs, the service distribution is irrelevant.

s.1.8.64. Answer = B, see 1.4.8.

s.1.8.65. Answer = B, (1.5.5).

s.1.8.66. Answer = B, see 1.1.2.

s.1.8.67. Answer = A, see1.3.6.

s.1.8.68. Answer = A.

s.1.8.69. Answer = A.

s.1.8.70. Let c_k^i be the capacity allocated to queue i in period k. The fair rule gives that

$$c_k^i = \frac{Q_{k-1}^i}{\sum_{i=1}^n Q_{k-1}^i} c_k.$$

Then,

$$d_k^i = \min\{Q_{k-1}^i, c_k^i\},$$
$$Q_k^i = Q_{k-1}^1 + a_k^i - d_k^i.$$

s.1.8.71. A machine serving work of different types of jobs. Jobs of the same type are put in the same queue. When the machine has to switch from one type of job to another, it might need to change a tool. For this reason it works on jobs of the same queue for some time. Then it changes to the next queue, and so on.

s.1.8.72. We need to compute the time it takes to clear Q_k^i. This is

$$\min\left\{r : \sum_{j=k+1}^{r} c_j^i \geq Q_k^i\right\}.$$

Some wrong answers:

- Computing $\mathsf{E}[W]$. The question is not to compute the expectation of waiting time...
- $W_k^i = (Q_{k-1}^i + a_k^i)/C_k$. This is partly ok, intuitively at least. However, the capacity c_k is the total capacity for all queues, hence the division should be by c_k^i. This is still not completely ok, because c_k^i is not constant as a function of k.
- $W_k = [W_{k-1} - X_k]^+ + S_k$. Here we are not dealing with a continuous-time queueing model.

s.1.8.73.

$$
\begin{aligned}
a'_{k,1} &= a_{k,1}\,\mathbb{1}_{Q_{k-1,1}+Q_{k-1,2}<K},\\
d_{k,1} &= \min\{c_{k,1}, Q_{k-1,1}\},\\
Q_{k,1} &= Q_{k-1,1} + a'_{k,1} - d_{k,1},\\
d_{k,2} &= \min\{c_{k,2}, Q_{k-1,2}\},\\
Q_{k,2} &= Q_{k-1,2} + d_{k,1} - d_{k,2}.
\end{aligned}
$$

Some students just model one queue, or do not mention that the arrivals at the second station are the departures of the first station.

s.1.8.74. This system is in fact unstable. Once $Q_n > 10$, there is a drift upwards with rate $2-1$. Eventually, the queue will drift to infinity.

Now even when $Qn > 10$, it may happen that some time later the queue length gets below 10. This happens when $a_i = 0$ for a sufficient number of consecutive periods. However, whenever the queue is less than 10, the state 10 will be hit with probability one. Eventually there will be time that it does not get back to state 10 again.

The process is a random walk with drift. The study of its probabilistic properties is interesting. Chapter 3 of Feller 1 is an interesting read.

s.1.8.75. It can take a long simulation time to obtain relevant answers. It is hard to get structural insights into systems. When you make a bug, it may be hard to find it out.

Some students say that the fact that a simulation is a model is itself a disadvantage. This is of course not a good answer. Anything we say is a model: the word 'apple' is not an apple itself, and all apples are slightly different...

Others say that with simulation it is only possible to model systems that operate in discrete time. This is nonsense, just check the book.

s.1.8.76. See 1.3.7.

s.1.8.77. The random walk acts as a queueing system without a boundary at $y = 0$; the random walk can be negative while the queueing system cannot be negative. Often a queueing system is called a reflected random walk. We analyzed the transient behavior of the random walk, and saw that that was already quite complicated. Including reflections makes the analysis of the transient behavior harder. Thus we decided to focus on the stationary behavior instead.

2

ANALYTICAL MODELS

In this chapter we focus on developing analytic models for various queueing systems in steady-state. In the analysis we use sample-path and level-crossing arguments to count how often certain events occur as a function of time. Then we define probabilities in terms of limits of fractions of these counting processes. Like this the performance measures can be explicitly computed for the statistical analysis of (simulations of) queueing systems.

As a reminder, we keep the discussion in these notes mostly at an intuitive level, and refer to El-Taha and Stidham Jr. [1998] for proofs and further background.

2.1 RATE STABILITY AND UTILIZATION

Theory and Exercises

In the analysis of any queueing process, the first step should be to check the relations between the arrival, service and departure rates. The concept of rate is crucial because it captures our intuition that when, in the long run, jobs arrive faster than they can leave, the system must 'explode'. As we will see, when the arrival rate is smaller than the service rate, the system is stable. Thus, the first performance measure we need to estimate for a queueing system is the ratio between the arrival and service rate. In this section, we develop some concepts and notation to formalize these ideas. We will use these concepts throughout the remainder of the book.

We first formalize the *arrival rate* and *departure rate* in terms of the *counting processes* $\{A(t)\}$ and $\{D(t)\}$. The *arrival rate* is the long-run average number of jobs that arrive per unit time, i.e.,

$$\lambda = \lim_{t\to\infty} \frac{A(t)}{t}. \tag{2.1.1}$$

We remark in passing that this limit does not necessarily exist if $A(t)$ is some pathological function. If, however, the inter-arrival times $\{X_k\}$ are the basic data, and $\{X_k\}$ are *independent and identically distributed (i.i.d.)* and distributed as a generic random variable X with finite mean $\mathsf{E}[X]$, we can construct $\{A_k\}$ and $\{A(t)\}$ as described in Section 1.5; the strong law of large numbers then guarantees that the above limit exists.

2.1.1. *Can you make an arrival process such that $A(t)/t$ does not have a limit?*

Observe that at time $t = A_n$, precisely n arrivals occurred. Thus, we see that $A(A_n) = n$, and therefore

$$\frac{1}{n}\sum_{k=1}^{n} X_k = \frac{A_n}{n} = \frac{A_n}{A(A_n)}.$$

But since $A_n \to \infty$ if $n \to \infty$, it follows from (2.1.1) that the average inter-arrival time between two consecutive jobs is

$$\mathsf{E}[X] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} X_k = \lim_{n\to\infty} \frac{A_n}{A(A_n)} = \lim_{t\to\infty} \frac{t}{A(t)} = \frac{1}{\lambda}, \tag{2.1.2}$$

where we take $t = A_n$ in the limit for $t \to \infty$. In words, the above states that the arrival rate λ is the inverse of the expected inter-arrival time.

2.1.2. *[Not obligatory] In* (2.1.2) *we replaced the limit with respect to n by a limit with respect to t. Use the notation $A_{A(t)}$ to show that this is allowed. Show next that the function $t \to A(t)$ as defined by* (1.5.3) *is right-continuous.*

The development of the departure times $\{D_k\}$ is entirely analogous to that of the arrival times; define the *departure rate* as

$$\delta = \lim_{t\to\infty} \frac{D(t)}{t}. \tag{2.1.3}$$

2.1.3. *Provide the details behind* (2.1.3).

Assume now that there is a single server. Let S_k be the required service time of the kth job to be served, and define

$$U_n = \sum_{k=1}^{n} S_k$$

as the total service time required by the first n jobs. With this, let

$$U(t) = \max\{n : U_n \le t\}$$

and define the *service rate* or *processing rate* as

$$\mu = \lim_{t\to\infty} \frac{U(t)}{t}.$$

In the same way as we derived that $\mathsf{E}[X] = 1/\lambda$, we obtain for the expected (or average) service time required by an individual job

$$\mathsf{E}[S] = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} S_k = \lim_{n\to\infty} \frac{U_n}{n} = \lim_{n\to\infty} \frac{U_n}{U(U_n)} = \lim_{t\to\infty} \frac{t}{U(t)} = \frac{1}{\mu}.$$

Now observe that, if the system is empty at time 0, it must be that at any time the number of departures must be smaller than or equal to the number of arrivals, i.e., $D(t) \le A(t)$ for all t. Therefore,

$$\delta = \lim_{t\to\infty} \frac{D(t)}{t} \le \lim_{t\to\infty} \frac{A(t)}{t} = \lambda. \tag{2.1.4}$$

We call a system *rate stable* if

$$\lambda = \delta,$$

in other words, the system is stable if, in the long run, jobs leave the system just as fast as they arrive. Of course, if $\lambda > \delta$, the system length process $L(t) \to \infty$ as $t \to \infty$.

It is also evident that jobs cannot depart faster than they can be served, hence, $D(t) \le U(t)$ for all t. Combining this with the fact that $\delta \le \lambda$, we get

$$\delta \le \min\{\lambda, \mu\}.$$

When $\mu \ge \lambda$ the above inequality reduces to $\delta = \lambda$ for rate-stable systems[1]. As it turns out, when $\mu = \lambda$ and the variance of the service time $\mathsf{V}[S] > 0$ or $\mathsf{V}[X] > 0$ the queue length process can behave in a very peculiar way. For this reason we henceforth (and implicitly) require that $\mu > \lambda$.

[1] It would be interesting to prove this.

2.1.4. *Define the random variables* $\{\tilde{X}_k, k = 1,\ldots\}$ *as* $\tilde{X}_k = S_{k-1} - X_k$. *For stability of the queueing process it is essential that* $\tilde{X}_k$ *has negative expectation, i.e.,* $\mathsf{E}\left[\tilde{X}_k\right] = \mathsf{E}[S_{k-1} - X_k] < 0$. *What is the conceptual meaning of this inequality?*

2.1.5. *Define* $\tilde{X}_k = S_{k-1} - X_k$. *Show that* $\mathsf{E}\left[\tilde{X}_k\right] < 0$ *implies that* $\lambda < \mu$.

2.1.6. *If the system starts empty, then we know that the number* $L(t)$ *in the system at time* t *is equal to* $A(t) - D(t)$. *Show that the system is rate-stable if* $L(t)$ *remains finite, or, more generally,* $L(t)/t \to 0$ *as* $t \to \infty$.

2.1.7. *Consider a paint factory that contains a paint mixing machine that serves two classes of jobs, A and B. The processing times of jobs of types A and B are constant and require* t_A *and* t_B *hours. The job arrival rate is* λ_A *for type A and* λ_B *for type B jobs. It takes a setup time of S hours to clean the mixing station when changing from paint type A to type B, and there is no time required to change from type B to A.*

To keep the system (rate) stable, it is necessary to produce the jobs in batches, for otherwise the server, i.e., the mixing machine, spends a too large fraction of time on setups, so that $\mu < \lambda$. *Thus, it is necessary to identify minimal batch sizes to ensure that* $\mu > \lambda$. *Motivate that the following linear program can be used to determine the minimal batch sizes:*

$$\textit{minimize } T$$

such that $T = k_A t_A + S + k_B t_B$, $\lambda_A T < k_A$ *and* $\lambda_B T < k_B$.

Hints

h.2.1.1. As a start, the function $\sin(t)$ does not have a limit as $t \to \infty$. However, the time-average $\sin(t)/t \to 0$. Now you need to make some function whose time-average does not converge, hence it should grow fast, or fluctuate wilder and wilder.

h.2.1.2. Use that $A_{A(t)} \le t < A_{A(t)+1}$. Divide by $A(t)$ and take suitable limits. BTW, such type of proof is used quite often to show that the existence of one limit implies, and is implied by, the existence of another type of limit.

h.2.1.5. Remember that $\{X_k\}$ and $\{S_k\}$ are sequences of i.i.d. random variables. What are the implications for the expectations?

h.2.1.7. Here are some questions to help you interpret this formulation.

1. What are the decision variables for this problem? In other words, what are the 'things' we can control/change?
2. What are the interpretations of $k_A t_A$, and $S + k_B t_B$?
3. What is the meaning of the first constraint? Realize that T represents one production cycle. After the completion of one such cycle, we start another cycle. Hence, the start of every cycle can be seen as a restart of the entire system.
4. What is the meaning of the other two constraints?
5. Why do we minimize the cycle time T?

6. Solve for k_A and k_B in terms of S, λ_A, λ_B and t_A, t_B.

7. Generalize this to m job classes and such that the cleaning time between jobs of class i and j is given by S_{ij}. (Thus, the setup times are sequence-dependent.)

Solutions

s.2.1.1. If $A(t) = 3t^2$, then clearly $A(t)/t = 3t$. This does not converge to a limit.

Another example, let the arrival rate $\lambda(t)$ be given as follows:

$$\lambda(t) = \begin{cases} 1 & \text{if } 2^{2k} \le t < 2^{2k+1} \\ 0 & \text{if } 2^{2k+1} \le t < 2^{2(k+1)}, \end{cases}$$

for $k = 0, 1, 2, \ldots$. Let $A(t) = \lambda(t)t$. Then $A(t)/t$ does not have limit. Of course, these examples are quite pathological, and are not representable for 'real life cases'. (Although this is also quite vague. What, then, is a real life case?)

For the mathematically interested, we seek a function for which its Cesàro limit does not exist.

s.2.1.2. This exercise is meant to provide some insight into what needs to be done to put everything on solid ground.

Observing that $A_{A(t)}$ is the arrival time of the last job before time t and that $A_{A(t)+1}$ is the arrival time of the first job after time t:

$$A_{A(t)} \le t < A_{A(t)+1} \Leftrightarrow \frac{A_{A(t)}}{A(t)} \le \frac{t}{A(t)} < \frac{A_{A(t)+1}}{A(t)} = \frac{A_{A(t)+1}}{A(t)+1}\frac{A(t)+1}{A(t)}.$$

Now $A(t)$ is a counting process such that $A(t) \to \infty$ as $t \to \infty$. Therefore, $\lim_{t\to\infty} A_{A(t)}/A(t) = \lim_{n\to\infty} A_n/n$. Moreover, it is evident that $\lim_{t\to\infty} A_{A(t)+1}/(A(t)+1) = \lim_{t\to\infty} A_{A(t)}/A(t)$, and that $(A(t)+1)/A(t) \to 1$ as $t \to \infty$. Thus it follows from the above inequalities that $\lim_{n\to\infty} A_n/n = \lim_{t\to\infty} t/A(t)$.

For the right-continuity of $A(t)$, define $f(t) = \mathbb{1}_{A_1 \le t}$. Observe first that $f(t)$ is increasing, and $f(t) \in \{0, 1\}$. Thus, if $f(t) = 1$ then $f(u) = 1$ for all $u \ge t$, and if $f(t) = 0$ then $f(u) = 0$ for all $u \le t$.

You may skip the rest of the proof below, but the above is essential to memorize; make a plot of $f(t)$, in particular, the behavior around A_1 is important.

We need to prove, for right-continuity, that $f(u) \to f(t)$ as $u \downarrow t$. When $f(t) = 1$, $f(u) = 1$ for any $u > 1$, by the definition of $f(x)$. When $f(t) = 0$ we have to do a bit more work. Formally, we have to prove that, for fixed t and for all $\epsilon > 0$, there is a $\delta > 0$ such that $u \in (t, t+\delta) \Rightarrow |f(u) - f(t)| < \epsilon$. (Note the differences with the regular definition of continuity.) Since, by assumption, t is such that $f(t) = 0$, and $f \in \{0, 1\}$ we need to show that $f(u) = 0$ for $u \in (t, t+\delta)$. Now, clearly, $f(t) = 0$ only if $t < A_1$. But, then for any $u \in (t, A_1)$, we have that $f(u) = 0$. Thus, taking $\delta = A_1 - t$ suffices.

The next step is to observe that $A(t)$ is a sum of right-continuous functions whose steps do not overlap since by assumption $0 < A_1 < A_2 < \cdots$. As A is (almost surely) a finite sum of bounded, increasing and right-continuous functions, it is also right-continuous.

If you like, you can try to prove this last step too.

Hopefully this problem, and its solution, clarifies that even such small details require attention. If we want to make some progress with respect to developing some queueing theory, we have to skip most of the proofs and mathematical problems; we simply don't have enough time in this course to be concerned with all theorems and proofs.

s.2.1.4. That the average time customers spend in service is smaller than the average time between the arrival of two subsequent jobs.

s.2.1.5. $0 > \mathsf{E}\left[\tilde{X}_k\right] = \mathsf{E}[S_{k-1} - X_k] = \mathsf{E}[S_{k-1}] - \mathsf{E}[X_k] = \mathsf{E}[S] - \mathsf{E}[X]$, where we use the fact that the $\{S_k\}$ and $\{X_k\}$ are i.i.d. sequences. Hence,

$$\mathsf{E}[X] > \mathsf{E}[S] \iff \frac{1}{\mathsf{E}[S]} > \frac{1}{\mathsf{E}[X]} \iff \mu > \lambda.$$

s.2.1.6. Since $L(t) = A(t) - D(t)$,

$$\lambda = \lim_{t\to\infty} \frac{A(t)}{t} = \lim_{t\to\infty} \frac{D(t)+L(t)}{t} = \lim_{t\to\infty} \frac{D(t)}{t} + \lim_{t\to\infty} \frac{L(t)}{t} = \delta.$$

Hence, when $L(t)/t \to 0$, the *up crossing rate* $\lim_{t\to\infty} A(t)/t = \lambda$ is equal to the *down-crossing rate* $\lim_{t\to\infty} D(t)/t = \delta$.

s.2.1.7. Realize that the machine works in cycles. A cycle starts with processing k_A jobs of type A, then does a setup, and processes k_B jobs of type B, and then a new cycle starts again. The time it takes to complete one such cycle is $T = k_A t_A + S + k_B t_B$. The number of jobs of type A processed during one such cycle is, of course, k_A. Observe next that the average number of jobs that arrive during one cycle is $\lambda_A T$. We of course want that $\lambda_A T < k_A$, i.e., fewer jobs of type A arrive on average per cycle than what we can process.

2.2 RENEWAL REWARD THEOREM AND LOAD

Theory and Exercises

We start with stating and proving (graphically) the *renewal reward theorem*. In the sequel, we will see many applications of this theorem. In this section, we use it to relate the fraction of time the server is busy in a $G/G/1$ queue to the job arrival rate and the expected job service time.

The renewal reward theorem is very useful and states intuitively that when customers arrive at rate λ and each customer pays an average amount X, then the system earns money at rate $Y = \lambda X$. Figure 8 provides graphical motivation about why this theorem is true; El-Taha and Stidham Jr. [1998] gives a (simple) proof.

Theorem 2.2.1 (Renewal Reward Theorem, $Y = \lambda X$)**.** Consider epochs $\{T_k, k = 0,1,\ldots\}$ such that $0 = T_0 < T_1 < \cdots$ Let $N = \{N(t), t \geq 0\}$ be the associated counting process with $N(t) = \max\{k : T_k \leq t\}$. Let $\{Y(t), t \geq 0\}$ be a non-decreasing right-continuous (deterministic) process. Define $X_k = Y(T_k) - Y(T_{k-1})$. Suppose that $N(t)/t \to \lambda$ as $t \to \infty$, where $0 < \lambda < \infty$. Then $Y(t)/t$ has a limit iff $n^{-1}\sum_{k=1}^{n} X_k$ has a limit, and then $Y = \lambda X$. In other words,

$$\lim_{t\to\infty} \frac{Y(t)}{t} = Y \iff \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} X_k = X,$$

and then $Y = \lambda X$.

Define the *load* or *utilization* as the limiting fraction of time the server is busy, i.e.,

$$\rho = \lim_{t\to\infty} \frac{1}{t} \int_0^t \mathbb{1}_{L(s)>0}\, \mathrm{d}s.$$

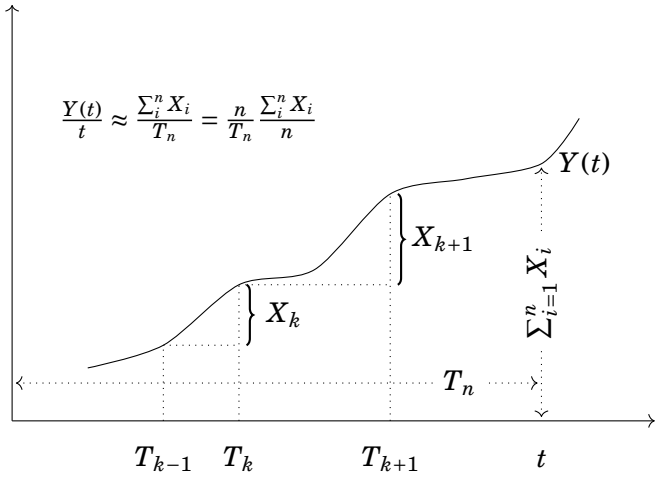


Figure 8: A graphical 'proof' of $Y = \lambda X$. Here $Y(t)/t \to Y$, $n/T_n \to \lambda$ and $n^{-1}\sum_i^n X_i \to X$. Observe that in the figure X_k does not represent an inter-arrival time; instead it corresponds to the increment of (the graph of) $Y(t)$ between two consecutive epochs T_{k-1} and T_k at which $Y(t)$ is observed.

2.2.1. *Use the renewal reward theorem to prove that $\rho = \lambda \mathsf{E}[S]$ for the rate-stable $G/G/1$ queue.*

2.2.2. *We can derive the relation $\rho = \lambda \mathsf{E}[S]$ in a somewhat more direct way by considering the fact that*

$$\sum_{k=1}^{A(t)} S_k \geq \int_0^t \mathbb{1}_{L(s)>0}\, ds \geq \sum_{k=1}^{D(t)} S_k.$$

Explain this, and complete the argument.

From the identities $\lambda^{-1} = \mathsf{E}[X]$ and $\mu^{-1} = \mathsf{E}[S]$, we get a further set of relations:

$$\rho = \lambda \mathsf{E}[S] = \frac{\lambda}{\mu} = \frac{\mathsf{E}[S]}{\mathsf{E}[X]}.$$

Thus, the load has also the interpretation as the rate at which jobs arrive times the average amount of work per job. Finally, recall that for a system to be rate-stable, it is necessary that $\mu > \lambda$, implying in turn that $\rho < 1$. The relation $\rho = \mathsf{E}[S]/\mathsf{E}[X] < 1$ then tells us that the average time it takes to serve a job must be less than the average time between two consecutive arrivals, i.e., $\mathsf{E}[S] < \mathsf{E}[X]$. In fact, when $\mu < \lambda$, it is easy to check with simulation that the queue length grows roughly linearly with slope $\lambda - \mu$.

2.2.3. *Consider a queueing system with c servers with identical production rates μ. What would be a reasonable stability criterion for this system?*

Hints

h.2.2.1. Define $Y(t) = \int_0^t \mathbb{1}_{L(s)>0}\, ds$ as the total amount of time the server has been busy up to the time t. Then take as epochs $T_k = D_k$ and use rate stability.

h.2.2.3. What is the rate in, and what is the service capacity?

Solutions

s.2.2.1. It is evident that $X_k = Y(D_k) - Y(D_{k-1}) = S_k$, hence $X = \lim_{n\to\infty} n^{-1}\sum_{k=1}^{n} X_k = \mathsf{E}[S]$. Also $\lim_{t\to\infty} Y(t)/t = \rho$. Finally, in the relation $Y = \lambda X$, the λ is δ since we consider departure epochs $T_k = D_k$, rather than A_k. By the renewal reward theorem $Y = \lambda X$ we get that $\rho = \delta\,\mathsf{E}[S]$. Finally, by rate-stability, the job arrival rate $\lambda = \delta$, hence $\rho = \lambda\,\mathsf{E}[S]$.

s.2.2.2. Observe that since t can lie half way a service interval and $A(t) \geq D(t)$. As $A(t) \to \infty$ as $t \to \infty$,

$$\lim_{t\to\infty} \frac{1}{t}\sum_{k=1}^{A(t)} S_k = \lim_{t\to\infty} \frac{A(t)}{t}\frac{1}{A(t)}\sum_{k=1}^{A(t)} S_k = \lim_{t\to\infty} \frac{A(t)}{t}\cdot \lim_{t\to\infty} \frac{1}{A(t)}\sum_{k=1}^{A(t)} S_k = \lambda\,\mathsf{E}[S].$$

Applying similar limits to the other inequality gives

$$\lambda\,\mathsf{E}[S] \geq \rho \geq \delta\,\mathsf{E}[S].$$

Hence, if $\delta = \lambda$, $\rho = \lambda\,\mathsf{E}[S]$.

Note that this is in fact the same argument as that underlies the renewal reward theorem. Henceforth we will just use the renewal reward theorem.

s.2.2.3. The criterion is that c must be such that $\lambda < c\mu$. (Thus, we interpret the number of servers as a *control*, i.e., a 'thing' we can change, while we assume that λ and μ cannot be easily changed.) To see this, we can take two different points of view. Imagine that the c servers are replaced by one server that works c times as fast. The service capacity of these two systems (i.e., the system with c servers and the system with one fast server) is the same, i.e., $c\mu$, where μ is the rate of one server. For the system with the fast server, the load is defined as $\rho = \lambda/c\mu$, and for stability we require $\rho < 1$. Another way to see it is to assume that the stream of jobs is split into c smaller streams, each with arrival rate λ/c. In this case, applying the condition that $(\lambda/c)/\mu < 1$ per server leads to the same condition that $\lambda/(c\mu) < 1$.

2.3 (LIMITS OF) EMPIRICAL PERFORMANCE MEASURES

Theory and Exercises

If the arrival and service processes are such that the queueing system is rate-stable, we can sensibly define other performance measures such as the average waiting time. In this section, we define the second most important performance measures; recall that the most important is the utilization ρ. At the end we provide an overview of the relations between these performance measures in Fig. 16.

With the construction of queueing processes in Section 1.5 we can compute the waiting time as observed by the first n, say, jobs. We therefore define the *expected waiting time* as

$$\mathsf{E}[W] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} W_k, \tag{2.3.1}$$

and the expected time in queue as

$$\mathsf{E}\left[W_Q\right] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} W_{Q,k}. \tag{2.3.2}$$

Note that these performance measures are limits of *empirical* measures. Note also that these statistics are as *observed by arriving jobs*: the first job has a waiting time W_1 at its arrival epoch, the second a waiting time W_2, and so on. For this reason, we colloquially say that $\mathsf{E}[W]$ is the average waiting time as 'seen by arrivals'. The *distribution of the waiting times at arrival times* can be found by counting:

$$\mathsf{P}(W \leq x) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{W_k \leq x}. \tag{2.3.3}$$

Finally, the (sample) *average number of jobs* in the system as seen by arrivals is given by

$$\mathsf{E}[L] = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} L(A_k-), \tag{2.3.4}$$

where $L(A_k-)$ is the number of jobs in the system at the arrival epoch of the kth job. The *distribution of* $\{L(t)\}$ *as seen by customers upon arrival*, is

$$\mathsf{P}(L \leq m) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{L(A_k-)\leq m}. \tag{2.3.5}$$

We call $\mathsf{P}(L > m)$ the *excess probability*.

A related set of performance measures follows by tracking the system's behavior over time and taking the *time-average*, rather than the average at sampling (observation) moments. Assuming the limit exists we use (1.5.10) to define the *time-average number of jobs* as

$$\mathsf{E}[L] = \lim_{t\to\infty} \frac{1}{t} \int_0^t L(s)\,\mathrm{d}s. \tag{2.3.6}$$

Observe that, notwithstanding that the symbols are the same, this expectation need not be the same as (2.3.4). In a loose sense we can say that $\mathsf{E}[L]$ is the average number in the system as perceived by the *server*. Next, define the *time-average fraction of time the system contains at most m jobs* as

$$\mathsf{P}(L \leq m) = \lim_{t\to\infty} \frac{1}{t} \int_0^t \mathbb{1}_{L(s)\leq m}\,\mathrm{d}s. \tag{2.3.7}$$

Again, this probability need not be the same as what customers see upon arrival.

2.3.1. *Design a queueing system to show that the average number of jobs in the system as seen by the server can be very different from what the customers see.*

2.3.2. *If $L(t)/t \to 0$ as $t \to \infty$, can it still be true that $\mathsf{E}[L] > 0$?*

2.3.3. *Consider a discrete-time model of a queueing system, such as the ones developed in Section 1.3. In such queueing systems, jobs arrive in batches, for instance, when $a_k = 3$, three jobs arrive in slot k. Assuming that $L_{k-1} = 5$, what queue length have these 3 arrivals seen?*

Provide one definition similar to (2.3.5) *for the case in which we say that all arrivals see the same number in the system. Provide a second in which we like to express that the first of a batch of arrivals sees less in the system than the last arrival of a batch.*

Hints

h.2.3.1. Consider a queueing system with constant service and inter-arrival times.

h.2.3.3. Realize that when jobs arrive in batches, the definition of loss fraction requires some care; not all definitions need to measure the same.

Solutions

s.2.3.1. Take $X_k = 10$ and $S_k = 10 - \epsilon$ for some tiny ϵ. Then $L(t) = 1$ nearly all of the time. In fact, $\mathsf{E}[L] = 1 - \epsilon/10$. However, $L(A_k-) = 0$ for all k.

s.2.3.2.

$$\mathsf{E}[L] = \lim_{t\to\infty} \frac{1}{t}\int_0^t L(s)\,\mathrm{d}s \neq \lim_{t\to\infty} \frac{L(t)}{t}.$$

If $L(t) = 1$ for all t, $\mathsf{E}[L] = 1$, but $L(t)/t \to 0$.

s.2.3.3. Suppose that we don't want to distinguish between jobs in a batch, but simply want to say that if one job sees a long queue, all see a long queue. In that case,

$$\frac{1}{A(n)}\sum_{k=1}^{n} a_k \mathbb{1}_{L_k>m}.$$

For the second code, observe that, since we deal with a system in discrete time, L_k is the queue length at the end of period k. Thus, $\sum_{k=1}^{n} \mathbb{1}_{L_k>m}$ counts the number of *periods* that the queue is larger than m. This is of course not the same as the number of *items* that see a queue larger than m; only when $a_k > 0$ the items in a batch would see a queue $L_k > m$. Thus,

$$\sum_{k=1}^{n} \mathbb{1}_{L_k>m}\,\mathbb{1}_{a_k>0},$$

counts the number of batches.

Next, by assumption, a_k items arrive during period k. The first of these items sees a queue length of $L_{k-1} - d_k$, the second $L_{k-1} - d_k + 1$, and so on until the last item, which sees a queue length of $L_k - 1 = L_{k-1} - d_k + a_k - 1$. Thus, of all items, the last item sees the largest queue. Hence, if $L_k \le m$, all items of the batch see a queue less than m. If, however, $L_k > m$, then $L_k - m$ customers saw m or more jobs in the system. Therefore, the fraction of arrivals that see a queue with m or more jobs is equal to

$$\frac{1}{A(n)}\sum_{k=1}^{n} (L_k - m)\,\mathbb{1}_{L_k>m}.$$

Here is the code for the second case.

```
>>> a = [0, 2, 5, 1, 2]
>>> c = [0, 1, 1, 0, 0]

>>> d = [0] * len(a)
>>> L = [0] * len(a)

>>> for k in range(1, len(a)):
...     d[k] = min(L[k - 1], c[k])
...     L[k] = L[k - 1] + a[k] - d[k]
...
>>> print(L)
[0, 2, 6, 7, 9]
```

```
>>> m = 5

>>> res = 0
>>> for k in range(1, len(a)):
...     res += (L[k] - m) * (L[k] - m)
...
>>> print(res, res / sum(a))
30 3.0
```

2.4 LEVEL CROSSING AND BALANCE EQUATIONS

Theory and Exercises

Let us say that the system is in *state* n at time t when it contains n jobs at that moment, i.e., when $L(t) = n$. The system *up-crosses level* n at time t when its state changes from n to $n+1$, due to an arrival, and it *down-crosses level* n when its state changes from $n+1$ to n, due to a departure. Clearly, the number of up-crossings and down-crossings must remain approximately the same, because it is only possible to up-cross level n after a down-crossing (or the other way around). This simple idea will prove a key stepping stone in the analysis of single-server queueing systems.

To establish the section's main result (2.4.5) we need a few definitions that are quite subtle and might seem a bit abstract, but below we will provide intuitive interpretations in terms of system KPIs. After this, we will generalize the principle of level-crossing to *balance equations* which allow us to deal with more general types of transitions. Note that Figure 17 at the end of the chapter summarizes all concepts we develop here.

LEVEL CROSSING Define

$$A(n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \le t} \mathbb{1}_{L(A_k-)=n} \tag{2.4.1a}$$

as the number of arrivals up to time t that saw n customers in the system upon their arrival, cf. Fig. 9.

2.4.1. *Why do we take $L(A_k-) = n$ rather than $L(A_k)$ in the definition of $A(n,t)$?*

2.4.2. *If $A(n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \le t} \mathbb{1}_{L(A_k-)=n}$, is $A(t) = \sum_{n=0}^{\infty} A(n,t)$?*

2.4.3. *Show that $A(n,t) \le A(t)$.*

2.4.4. *If $\lambda > \delta$ can it happen that $\lim_{t\to\infty} A(n,t)/t > 0$ for some (finite) n?*

Next, let

$$Y(n,t) = \int_0^t \mathbb{1}_{L(s)=n} \, \mathrm{d}s \tag{2.4.1b}$$

be the total time the system contains n jobs during $[0,t]$, and

$$p(n,t) = \frac{1}{t} \int_0^t \mathbb{1}_{L(s)=n} \, \mathrm{d}s = \frac{Y(n,t)}{t}, \tag{2.4.1c}$$

be the fraction of time that $L(s) = n$ in $[0,t]$. Figure 10 illustrates the relation between $Y(n,t)$ and $A(n,t)$.

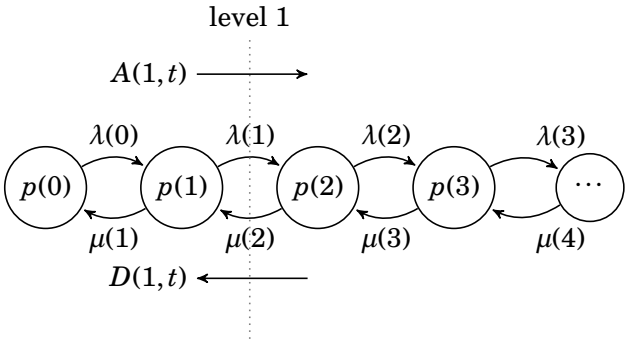


Figure 9: $A(1,t)$ counts the number of jobs up to time t that saw 1 job in the system upon arrival, and right after such arrivals the system contains 2 jobs. Thus, each time $A(1,t)$ increases by one, level 1 (the dotted line separating states 1 and 2) is crossed from below. Similarly, $D(1,t)$ counts the number of departures that leave 1 job behind, and just before such departures the system contains 2 jobs. Hence, level 1 is crossed from above. It is evident that the number of times this level is crossed from below must be the same (plus or minus 1) as the number of times it is crossed from above. (We introduce $\lambda(n)$, $\mu(n)$ and $p(n)$ below.)

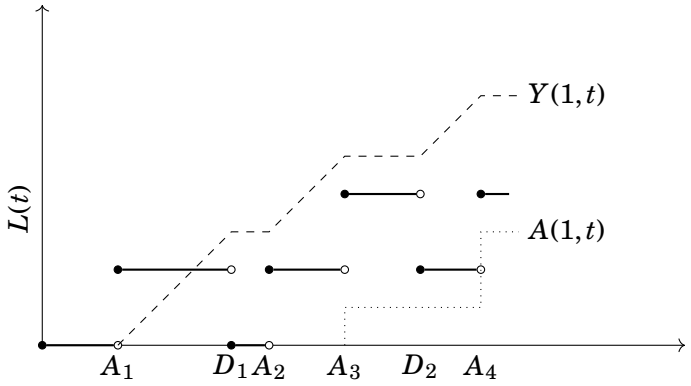


Figure 10: Plots of $Y(1,t)$ and $A(1,t)$. (For visual clarity, we subtracted 1/2 from $A(1,t)$, for otherwise its graph would partly overlap with the graph of Y.)

2.4.5. *Consider the following (silly) queueing process. At times* $0,2,4,\ldots$ *customers arrive, each customer requires* 1 *unit of service, and there is one server. Find an expression for* $A(n,t)$*. (What acronym would describe this queueing situation?)*

2.4.6. *[Continuation of 2.4.5] Find an expression for* $Y(n,t)$*.*

Define also the limits:

$$\lambda(n) = \lim_{t\to\infty} \frac{A(n,t)}{Y(n,t)}, \qquad\qquad p(n) = \lim_{t\to\infty} p(n,t), \tag{2.4.2}$$

as the *arrival rate in state* n and the *long-run fraction of time the system spends in state* n. To clarify the former definition, observe that $A(n,t)$ counts the number of arrivals that see n jobs in the system upon arrival, while $Y(n,t)$ tracks the amount of time the system contains n jobs. Suppose that at time T a job arrives that sees n jobs in the system. Then $A(n,T) = A(n,T-)+1$, and this job finishes an interval that is tracked by $Y(n,t)$, precisely because this job sees n jobs in the system just prior to its arrival. Thus, just as $A(t)/t$ is the total number of arrivals during

$[0,t]$ divided by t, $A(n,t)/Y(n,t)$ is the number of arrivals that see n jobs divided by the time the system contains n jobs.

2.4.7. *[Continuation of 2.4.6] Compute $p(n)$ and $\lambda(n)$.*

Similar to the definition for $A(n,t)$, let

$$D(n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{D_k \le t}\, \mathbb{1}_{L(D_k)=n}$$

denote the number of departures up to time t that *leave n customers behind*. Then, define

$$\mu(n+1) = \lim_{t\to\infty} \frac{D(n,t)}{Y(n+1,t)},$$

as *the departure rate from state $n+1$*. (It is easy to get confused here: to leave n jobs behind, the system must contain $n+1$ jobs just prior to the departure.) Figure 9 shows how $A(n,t)$ and $\lambda(n)$ relate to $D(n+1,t)$ and $\mu(n)$.

2.4.8. *Should we take $D(n-1,t)$ or $D(n,t)$ in the definition of $\mu(n)$?*

2.4.9. *[Continuation of 2.4.7] Compute $D(n,t)$ and $\mu(n+1)$ for $n \ge 0$.*

Observe that customers arrive and depart as single units. Thus, if $\{T_k\}$ is the ordered set of arrival and departure times of the customers, then $L(T_k) = L(T_k-) \pm 1$. But then we must also have that $|A(n,t) - D(n,t)| \le 1$ (think about this). From this observation it follows immediately that

$$\lim_{t\to\infty} \frac{A(n,t)}{t} = \lim_{t\to\infty} \frac{D(n,t)}{t}. \tag{2.4.3}$$

With this equation we can obtain two nice and fundamental identities. The first we develop now; the second follows in Section 2.7.

The rate of jobs that 'see the system with n jobs' can be defined as $A(n,t)/t$. Taking limits we get

$$\lim_{t\to\infty} \frac{A(n,t)}{t} = \lim_{t\to\infty} \frac{A(n,t)}{Y(n,t)} \frac{Y(n,t)}{t} = \lambda(n)p(n), \tag{2.4.4a}$$

where we use the above definitions for $\lambda(n)$ and $p(n)$. Similarly, the departure rate of jobs that leave n jobs behind is

$$\lim_{t\to\infty} \frac{D(n,t)}{t} = \lim_{t\to\infty} \frac{D(n,t)}{Y(n+1,t)} \frac{Y(n+1,t)}{t} = \mu(n+1)p(n+1). \tag{2.4.4b}$$

Combining this with (2.4.3) we arrive at the *level-crossing equations*

$$\lambda(n)p(n) = \mu(n+1)p(n+1). \tag{2.4.5}$$

2.4.10. *[Continuation of 2.4.9] Compute $\lambda(n)p(n)$ for $n \ge 0$, and check $\lambda(n)p(n) = \mu(n+1)p(n+1)$.*

Result (2.4.5) turns out to be exceedingly useful, as will become evident from Section 2.5 onward. More specifically, by specifying (i.e., modeling) $\lambda(n)$ and $\mu(n)$, we can compute the long-run fraction of time $p(n)$ that the system contains n jobs. To see this, rewrite the above into

$$p(n+1) = \frac{\lambda(n)}{\mu(n+1)} p(n). \tag{2.4.6}$$

Thus, this equation fixes the ratios between the probabilities. In other words, if we know $p(n)$ we can compute $p(n+1)$, and so on. Hence, if $p(0)$ is known, then $p(1)$ follows, from which $p(2)$ follows, and so on. A straightaway iteration then leads to

$$p(n+1) = \frac{\lambda(n)\lambda(n-1)\cdots\lambda(0)}{\mu(n+1)\mu(n)\cdots\mu(1)}p(0). \tag{2.4.7}$$

To determine $p(0)$ we can use the fact that the numbers $p(n)$ represent probabilities. Hence, from the normalizing condition $\sum_{n=0}^{\infty} p(n) = 1$, we get $p(0) = G^{-1}$ with G being the *normalization constant*

$$G = 1 + \sum_{n=0}^{\infty} \frac{\lambda(n)\lambda(n-1)\cdots\lambda(0)}{\mu(n+1)\mu(n)\cdots\mu(1)}. \tag{2.4.8}$$

In the next few sections we will make suitable choices for $\lambda(n)$ and $\mu(n)$ to model many different queueing situations so that, based on (2.4.5), we can obtain simple expressions for $p(n)$ in terms of the arrival and service rates.

With $p(n)$ we define two easy, but important performance measures. The time-average number of items in the system becomes

$$\mathsf{E}[L] = \sum_{n=0}^{\infty} np(n),$$

and the long-run fraction of time the system contains at least n jobs is

$$\mathsf{P}(L \geq n) = \sum_{i=n}^{\infty} p(i).$$

2.4.11. *Derive* $\mathsf{E}[L] = \sum_{n=0}^{\infty} np(n)$ *from* (2.3.6).

Finally, the following two exercises show that level-crossing arguments extend well beyond the queueing systems modeled by Fig. 9.

2.4.12. *Consider a single server that serves one queue and serves only in batches of 2 jobs at a time (so never 1 job or more than 2 jobs), i.e., the* $M/M^2/1/3$ *queue. Single jobs arrive at rate* λ *and the inter-arrival times are exponentially distributed so that we can assume that* $\lambda(n) = \lambda$*. The batch service times are exponentially distributed with mean* $1/\mu$*. Then, by the memoryless property,* $\mu(n) = \mu$*. At most 3 jobs fit in the system. Make a graph of the state-space and show, with arrows, the transitions that can occur.*

2.4.13. *Use the graph of 2.4.12 and a level-crossing argument to express the steady-state probabilities* $p(n), n = 0, \ldots, 3$ *in terms of* λ *and* μ*.*

INTERPRETATION The definitions in (2.4.1) may seem a bit abstract, but they obtain an immediate interpretation when relating them to applications. To see this, we discuss two examples.

Consider the sorting process of post parcels at a distribution center of a post-delivery company. Each day tens of thousands of incoming parcels have to be sorted to their final destination. In the first stage of the process, parcels are sorted to a region in the Netherlands. Incoming parcels are deposited on a conveyor belt. From the belt, they are carried to outlets (chutes), each chute corresponding to a specific region. Employees take out the parcels from the chutes and put the parcels in containers. The arrival rate of parcels for a certain chute may temporarily exceed the working capacity of the employees, as such the chute serves as a queue. When

the chute overflows, parcels are directed to an overflow container and are sorted the next day. The target of the sorting center is to deliver at least a certain percentage of the parcels within one day. Thus, the fraction of parcels rejected at the chute should remain small.

Suppose a chute can contain at most 20 parcels, say. Then, each parcel on the belt that 'sees' 20 parcels in its chute will be blocked. Let $L(t)$ be the number of parcels in the chute at time t. Then, $A(20,t)$ as defined in (2.4.1a) is the number of *blocked parcels* up to time t, and $A(20,t)/A(t)$ is the fraction of rejected parcels. In fact, $A(20,t)$ and $A(t)$ are continuously tracked by the sorting center and used to adapt employee capacity to control the fraction of rejected parcels. Thus, in simulations, if one wants to estimate loss fractions, $A(n,t)/A(t)$ is the most natural concept to consider.

For the second example, suppose there is a cost associated with keeping jobs in queue. Let w be the cost per job in queue per unit time so that the cost rate is nw when n jobs are in queue. But then $wnY(n,t)$ is the total cost up to time t to have n jobs in queue, hence the total cost up to time t is

$$C(t) = w \sum_{n=0}^{\infty} nY(n,t),$$

and the average cost is

$$\frac{C(t)}{t} = w \sum_{n=0}^{\infty} n \frac{Y(n,t)}{t} = w \sum_{n=0}^{\infty} np(n,t).$$

All in all, the concepts developed above have natural interpretations in practical queueing situations; they are useful in theory and in simulation, as they relate the theoretical concepts to actual measurements.

BALANCE EQUATIONS It is important to realize that the level-crossing argument cannot always be used as we do here. The reason is that sometimes there does not exist a line between two states such that the state space splits into two disjoint parts. For a more general approach, we focus on a single state and count how often this state is entered and left, cf. Fig. 11. Specifically, define

$$I(n,t) = A(n-1,t) + D(n,t),$$

as the number of times the queueing process enters state n either due to an arrival from state $n-1$ or due to a departure leaving n jobs behind. Similarly,

$$O(n,t) = A(n,t) + D(n-1,t),$$

counts how often state n is left either by an arrival (to state $n+1$) or a departure (to state $n-1$).

Of course, $|I(n,t) - O(n,t)| \le 1$. Thus, from the fact that

$$\lim_{t\to\infty} \frac{I(n,t)}{t} = \lim_{t\to\infty} \frac{A(n-1,t)}{t} + \lim_{t\to\infty} \frac{D(n,t)}{t} = \lambda(n-1)p(n-1) + \mu(n+1)p(n+1)$$

and

$$\lim_{t\to\infty} \frac{O(n,t)}{t} = \lim_{t\to\infty} \frac{A(n,t)}{t} + \lim_{t\to\infty} \frac{D(n-1,t)}{t} = \lambda(n)p(n) + \mu(n)p(n)$$

we get that

$$\lambda(n-1)p(n-1) + \mu(n+1)p(n+1) = (\lambda(n) + \mu(n))p(n).$$

These equations hold for any $n \ge 0$ and are known as the *balance equations*. We will use these equations when studying queueing systems in which level-crossing cannot be used, for instance for queueing networks.

Again, just by using properties, i.e., counting differences, that hold along any sensible sample path we obtain very useful statistical and probabilistic results.

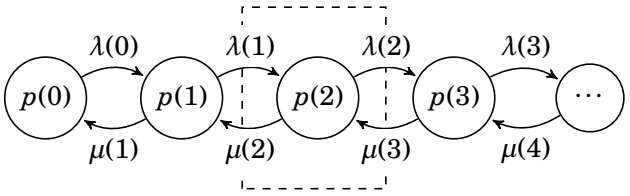


Figure 11: For the balance equations we count how often a box around a state is crossed from inside and outside. In the long run, the entering and leaving rates should be equal. For the example here, the rate out is $p(2)\lambda(2)+p(2)\mu(2)$ while the rate in is $p(1)\lambda(1)+p(3)\mu(3)$.

Hints

h.2.4.1. Recall that $L(t)$ is *right-continuous*.

h.2.4.5. For the acronym, observe that the service times and inter-arrival are deterministic and there is one server. For the computation of $Y(n,t)$, make a plot of $L(s)$ as a function of time for $n=1$.

Make a plot of $L(s)$ for $n=1$ as a function of time.

h.2.4.13. First, balance the rates across the levels. Then solve this in terms of $p(0)$.

Solutions

s.2.4.1. $L(t)$ is the number of customers in the system at time t. As such the function $t \to L(t)$ is *right-continuous*. The definition of $L(A_k-) = \lim_{t\uparrow A_k} L(t)$ is the limit from the left. The customer therefore 'sees' $L(A_k-)$ just before he/she arrives.

s.2.4.2. $A(t)$ counts all customers that arrive up to time t, i.e., during $[0,t]$. Note that this *includes* time t. $A(n,t)$ counts the jobs that see n jobs in the system just before they arrive. As there either 0, or 1, or 2, etc., jobs in the system upon arrival the summation covers all possible outcomes. Therefore the claim is true.

s.2.4.3. Observe that $\mathbb{1}_{A_k\le t}\mathbb{1}_{L(A_k-)=n} \le \mathbb{1}_{A_k\le t}$; the last inequality follows from the fact that $\mathbb{1}_{L(A_k-)=n} \le 1$. Therefore,

$$A(n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k\le t}\mathbb{1}_{L(A_k-)=n} \le \sum_{k=1}^{\infty} \mathbb{1}_{A_k\le t} = A(t).$$

For any 'normal' queueing system, $A(t) > A(n,t)$, because the queue length fluctuates.

s.2.4.4. If $\lambda > \delta$, then $L(t) \to \infty$. But then there must be a last time, s say, that $L(s) = n+1$, and $L(t) > n+1$ for all $t > s$. Hence, after time s no job will see the system with n jobs. Thus $A(n,t) = A(n,s)$ for all $t > s$. This is a finite number, while $t \to \infty$, so that $A(n,t)/t \to 0$.

s.2.4.5. It is the $D/D/1$ queue, since there is one server and the inter-arrival times and service times are constant, i.e., deterministic. $A_k = 2k$ as jobs arrive at $t = 0,2,4,\ldots$, hence, $A(t) \approx t/2$ when $t \gg 0$. We also know that $L(s) = 1$ if $s \in [2i, 2i+1)$ and $L(s) = 0$ for $s \in [2i-1, 2i)$ for $i = 0,1,2,\ldots$. Thus, $L(A_k-) = L(2k-) = 0$. Hence, $A(0,t) \approx t/2$ for $t \gg 0$, and $A(n,t) = 0$ for $n \ge 1$.

s.2.4.6. Next, to get $Y(n,t)$, observe that the system never contains more than 1 job. Hence, $Y(n,t) = 0$ for all $n \geq 2$. Then we see that $Y(1,t) = \int_0^t \mathbb{1}_{L(s)=1}\,\mathrm{d}s$. Now observe that for our queueing system $L(s) = 1$ for $s \in [0,1)$, $L(s) = 0$ for $s \in [1,2)$, $L(s) = 1$ for $s \in [2,3)$, and so on. Thus, when $t < 1$, $Y(1,t) = \int_0^t \mathbb{1}_{L(s)=1}\,\mathrm{d}s = \int_0^t 1\,\mathrm{d}s = t$. When $t \in [1,2)$,

$$L(t) = 0 \implies \mathbb{1}_{L(t)=0} \implies Y(1,t) \text{ does not change.}$$

Continuing to $[2,3)$ and so on gives

$$Y(1,t) = \begin{cases} t & t \in [0,1), \\ 1 & t \in [1,2), \\ 1+(t-2) & t \in [2,3), \\ 2 & t \in [3,4), \\ 2+(t-4) & t \in [4,5), \end{cases}$$

and so on. Since $Y(n,t) = 0$ for all $n \geq 2$, $L(s) = 1$ or $L(s) = 0$ for all s, therefore,

$$Y(0,t) = t - Y(1,t).$$

s.2.4.7. From the other exercises:

$$\begin{aligned} \lambda(0) &\approx \frac{A(0,t)}{Y(0,t)} \approx \frac{t/2}{t/2} = 1, \\ \lambda(1) &\approx \frac{A(1,t)}{Y(1,t)} \approx \frac{0}{t/2} = 0, \\ p(0) &\approx \frac{Y(0,t)}{t} \approx \frac{t/2}{t} = \frac{1}{2}, \\ p(1) &\approx \frac{Y(1,t)}{t} \approx \frac{t/2}{t} = \frac{1}{2}. \end{aligned}$$

For the rest $\lambda(n) = 0$, and $p(n) = 0$, for $n \geq 2$.

s.2.4.8. $D(n-1,t)$ counts the departures that leave $n-1$ behind. Thus, just before the customer leaves, the system contains n customers.

s.2.4.9. $D(0,t) = \sum_{k=1}^{\infty} \mathbb{1}_{D_k \leq t, L(D_k)=0}$. From the graph of $\{L(s)\}$ we see that all jobs leave an empty system behind. Thus, $D(0,t) \approx t/2$, and $D(n,t) = 0$ for $n \geq 1$. With this, $D(0,t)/Y(1,t) \sim (t/2)/(t/2) = 1$, and so,

$$\mu(1) = \lim_{t\to\infty} \frac{D(0,t)}{Y(1,t)} = 1,$$

and $\mu(n) = 0$ for $n \geq 2$.

s.2.4.10. $\lambda(0)p(0) = 1 \cdot 1/2 = 1/2$, $\lambda(n)p(n) = 0$ for $n > 1$, as $\lambda(n) = 0$ for $n > 0$.
From 2.4.9, $\mu(1) = 1$, hence $\mu(1)p(1) = 1 \cdot 1/2 = 1/2$. Moreover, $\mu(n) = 0$ for $n \geq 2$.
Clearly, for all n we have $\lambda(n)p(n) = \mu(n+1)p(n+1)$.

s.2.4.11. As $L(s)$ counts the number of jobs in the system at time s (thus $L(s)$ is an integer),

$$L(s) = \sum_{n=0}^{\infty} n\,\mathbb{1}_{L(s)=n}.$$

With this we can write for the time-average number of jobs in the system

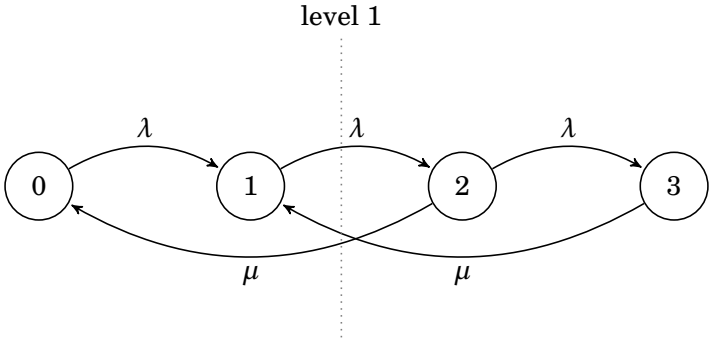
$$\frac{1}{t}\int_0^t L(s)\,\mathrm{d}s = \frac{1}{t}\int_0^t \left(\sum_{n=0}^{\infty} n\,\mathbb{1}_{L(s)=n}\right)\mathrm{d}s = \sum_{n=0}^{\infty}\frac{n}{t}\int_0^t \mathbb{1}_{L(s)=n}\,\mathrm{d}s, \tag{2.4.9}$$

where we interchange the integral and the summation[2]. It then follows from (2.4.1c) that

$$\frac{1}{t}\int_0^t L(s)\,\mathrm{d}s = \sum_{n=0}^{\infty} n\,p(n,t).$$

Finally, assuming that the limit $p(n,t)\to p(n)$ exists as $t\to\infty$ (and that the summation and limit can be interchanged in the above), the result follows.

s.2.4.12. See the figure below.



s.2.4.13. With level-crossing:

$$\begin{aligned}
&\lambda p(0) = \mu p(1), \quad \text{the level between 0 and 1,}\\
&\lambda p(1) = \mu p(2) + \mu p(3), \quad \text{see level 1,}\\
&\lambda p(2) = \mu p(3), \quad \text{the level between 2 and 3.}
\end{aligned}$$

Solving this in terms of $p(0)$ gives $p(2)=\rho p(0)$, $p(3)=\rho p(2)=\rho^2 p(0)$, and

$$\lambda p(1) = \mu(p(2)+p(3)) = \mu(\rho+\rho^2)p(0) = (\lambda+\lambda^2/\mu)p(0),$$

hence $p(1)=p(0)(\mu+\lambda)/\mu$.

2.5 $M/M/1$ QUEUE

Theory and Exercises

In the $M/M/1$ queue, one server serves jobs arriving with exponentially distributed inter-arrival times and each job requires an exponentially distributed processing time. With the level-crossing equations (2.4.6) we derive a number of important results for this queueing process.

Recall from Section 1.7 that we can construct the $M/M/1$ queue as a reflected random walk where the arrivals are generated by a Poisson process $N_\lambda(t)$ and the departures (provided the

[2] This is allowed as the integrand is non-negative. More generally, the interested reader should check Fubini's theorem.

number $L(t)$ in the system is positive) are generated according to the Poisson process $N_\mu(t)$. Since the rates of these processes do not depend on the state of the random walk nor on the queue process, it follows that $\lambda(n) = \lambda$ for all $n \geq 0$ and $\mu(n) = \mu$ for all $n \geq 1$. Thus, (2.4.6) reduces to

$$p(n+1) = \frac{\lambda(n)}{\mu(n+1)} p(n) = \frac{\lambda}{\mu} p(n) = \rho p(n),$$

where we use the definition of the load $\rho = \lambda/\mu$. Since this holds for any $n \geq 0$, it follows with recursion that

$$p(n+1) = \rho^{n+1} p(0).$$

Then, by using normalization, it follows from (2.4.8) and (1.1.1d) that

$$p(0) = 1 - \rho, \qquad p(n) = (1-\rho)\rho^n. \tag{2.5.1}$$

It is now easy to compute the most important performance measures. The utilization of the server is $\rho = \lambda/\mu$, as observed above. Then, with a bit of algebra,

$$\mathsf{E}[L] = \frac{\rho}{1-\rho}, \qquad \mathsf{V}[L] = \frac{\rho}{(1-\rho)^2}, \qquad \mathsf{P}(L > n) = \rho^{n+1}. \tag{2.5.2}$$

2.5.1. *What is the interpretation of* $\sum_{n=1}^{\infty} p(n)$*?*

2.5.2. *Derive* (2.5.2) *with indicator functions.*

2.5.3. *Derive* (2.5.2) *by differentiating the left-hand and right-hand side of the standard formula for a geometric series:* $\sum_{n=0}^{\infty} \rho^n = (1-\rho)^{-1}$ *for* $|\rho| < 1$.

2.5.4. *Now use moment-generating functions to derive* (2.5.2).

2.5.5. *Derive*

$$\mathsf{E}\left[L^2\right] = (1-\rho) \sum_{n=0}^{\infty} n^2 \rho^n = \frac{2\rho^2}{(1-\rho)^2} + \frac{\rho}{1-\rho} \tag{2.5.3}$$

by differentiating the standard formula for a geometric series twice.

2.5.6. *Show* (2.5.3) *by noting that* $\sum_{i=1}^{n} i = n(n+1)/2$ *from which we get that* $n^2 = -n + 2\sum_{i=1}^{n} i$. *Substitute this relation into* $\sum_n n^2 \rho^n$ *and simplify.*

2.5.7. *Derive* (2.5.3) *by using the moment-generating functions.*

2.5.8. *Show that for the* $M/M/1$ *queue* $\mathsf{V}[L] = \rho/(1-\rho)^2$.

2.5.9. *Show that for the* $M/M/1$ *queue the SCV of* L *is* $1/\rho$. *What do you conclude from this?*

2.5.10. *Show that the excess probability, i.e., the probability that a long queue occurs, is* $\mathsf{P}(L \geq n) = \rho^n$.

2.5.11. *Explain that for the* $M/M/1$ *queue* $\mathsf{E}\left[L_Q\right] = \sum_{n=1}^{\infty} (n-1)\pi(n)$ *and use this to find that* $\mathsf{E}\left[L_Q\right] = \rho^2/(1-\rho)$.

Let us interpret (2.5.2). The fact that $\mathsf{E}[L] \sim (1-\rho)^{-1}$ for $\rho \to 1$ implies that the average waiting time increases very fast when $\rho \to 1$. If we want to avoid long waiting times, this formula tells us that situations with $\rho \approx 1$ should be avoided. As a practical guideline, it is typically best to keep ρ quite a bit below 1, and accept that servers are not fully utilized.

Clearly, the probability that the queue length exceeds some threshold decreases geometrically fast (for $\rho < 1$). If we make the simple assumption that customers decide to leave (or rather, not join) the system when the queue is longer than 9 say, then $\mathsf{P}(L \geq 10) = \rho^{10}$ is an estimator for the fraction of customers lost.

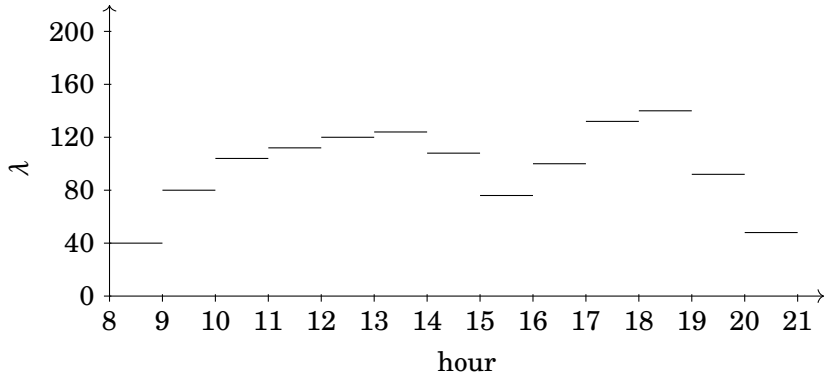


Figure 12: A demand profile of the arrival rate λ modeled as constant over each hour.

SUPERMARKET PLANNING Let us consider the example of cashier planning of a supermarket to demonstrate how to use the tools we developed up to now. Out of necessity, our approach is a bit heavy-handed—Turning the example into a practically useful scheme requires more sophisticated queueing models and data assembly—but the present example contains the essential analytic steps to solve the planning problem.

The *service objective* is to determine the minimal service capacity c (i.e., the number of cashiers) such that the fraction of the time that more than 10 people are in queue is less than 1%. (If the supermarket has 3 cashiers open, 10 people in queue means about 3 people per queue.)

The next step is to find the *relevant data*: the arrival process and the service time distribution. For the arrival process, it is reasonable to model it as a Poisson process. There are many potential customers, each choosing with a small probability to go to the supermarket at a certain moment in time. Thus, we only have to characterize the arrival rate. Estimating this for a supermarket is relatively easy: the cash registers track all customers payments. Thus, we know the number of customers that left the shop, hence entered the shop. (We neglect the time customers spend in the shop.) Based on these data we make a *demand profile*: the average number of customers arriving per hour, cf. Fig. 12. Then we model the arrival process as Poisson with an arrival rate that is constant during a certain hour as specified by the demand profile.

It is also easy to find the service distribution from the cash registers. The first item scanned after a payment determines the start of a new service, and the payment closes the service. (As there is always a bit of time between the payment and the start of a new service we might add 15 seconds, say, to any service.) To keep things simple here, we just model the service time distribution as exponential with a mean of 1.5 minutes.

We also *model* the behavior of all the cashiers together (a multi-server queue) as a single fast server. Thus, we neglect any differences between a station with, for instance, 3 cashiers and a single server that works 3 times as fast as a normal cashier. (We analyze in 2.6.5 the quality of this approximation.) As yet another simplification, we change the objective somewhat such that the number of jobs in the system, rather than the number in queue, should not exceed 10.

We now find a formula to convert the demand profile into the *load profile*, which is the minimal number of servers per hour needed to meet the service objective. We already know for the $M/M/1$ that $\mathsf{P}(L > 10) = \rho^{11}$. Combining this with the objective $\mathsf{P}(L > 10) \leq 1\%$, we get that

$\rho^{11} \leq 0.01$, which translates into $\rho \leq 0.67$. Using that $\rho = \lambda \mathsf{E}[S]/c$ and our estimate $\mathsf{E}[S] = 1.5$ minutes, we get the following rough bound on c:

$$c \geq \frac{\lambda \mathsf{E}[S]}{0.67} \approx \frac{3}{2} \cdot \lambda \cdot 1.5 = 2.25\lambda,$$

where λ is the arrival rate (per minute, *not* per hour). For instance, for the hour from 12 to 13, we read in the demand profile in Fig. 12 that $\lambda = 120$ customers per hour, hence $c = 2.25 \cdot 120/60 = 4.5$. With this formula, the conversion of the demand profile to the load profile becomes trivial: divide the hourly arrival rate by 60 and multiply by 2.25.

The last step is to *cover the load profile with service shifts*. This is typically not easy since shifts have to satisfy all kinds of rules, such as: after 2 hours of work a cashier should take a break of at least 10 minutes; a shift length must be at least four hours, and no longer than 9 hours including breaks; when the shift is longer than 4 hours it needs to contain at least one break of 30 minutes; and so on. These shifts also have different costs: shifts with hours after 18h are more expensive per hour; when the supermarket covers traveling costs, short shifts have higher marginal traveling costs; and so on.

The usual way to solve such covering problems is by means of an integer problem. First, generate all (or a subset of the) allowed shift types with associated starting times. For instance, suppose only 4 shift plans are available

1. $++-++$
2. $+++-+$
3. $++-+++$
4. $+++-++$,

where a $+$ indicates a working hour and $-$ a break of an hour. Then generate shift types for each of these plans with starting times 8 am, 9 am, and so on, until the end of the day. Thus, a shift type is a shift plan that starts at a certain hour. Let x_i be the number of shifts of type i and c_i the cost of this type. Write $t \in s_i$ if hour t is covered by shift type i. Then the problem is to solve

$$\min \sum_i c_i x_i,$$

such that

$$\sum_i x_i \, \mathbb{1}_{t \in s_i} \geq 2.25 \frac{\lambda_t}{60}$$

for all hours t the shop is open and λ_t is the demand for hour t.

Hints

h.2.5.3. Observe that we just need to compute the first second moment of a geometric random variable.

h.2.5.10. $\mathsf{P}(L \geq n) = \sum_{k \geq n} p(k)$.

Solutions

s.2.5.1. First, note that $p(0)$ must be the fraction of time the server is idle. Hence, the fraction of time the server is busy, i.e., the utilization, is

$$1 - p(0) = \rho = \sum_{n=1}^{\infty} p(n).$$

Here the last equation has the interpretation of the fraction of time the system contains at least 1 job.

s.2.5.2. A bit long, but I spell out every step:

$$\begin{aligned}
\mathsf{E}[L] &= \sum_{n=0}^{\infty} n p(n) \\
&= \sum_{n=0}^{\infty} \sum_{i=1}^{n} \mathbb{1}_{i \le n} p(n) && n = \sum_{i=1}^{n} \mathbb{1}_{i \le n} \\
&= \sum_{n=0}^{\infty} \sum_{i=1}^{\infty} \mathbb{1}_{i \le n} p(n) && i > n \implies \mathbb{1}_{i \le n} = 0 \\
&= \sum_{i=1}^{\infty} \sum_{n=0}^{\infty} \mathbb{1}_{i \le n} p(n) && \text{Fubini} \\
&= \sum_{i=1}^{\infty} \sum_{n=i}^{\infty} p(n) && n < i \implies \mathbb{1}_{i \le n} = 0 \\
&= \sum_{i=1}^{\infty} \sum_{n=i}^{\infty} (1-\rho)\rho^n && p(n) = (1-\rho)\rho^n \\
&= \sum_{i=1}^{\infty} \sum_{n=0}^{\infty} (1-\rho)\rho^{n+i} && n \to n+i \\
&= \sum_{i=1}^{\infty} (1-\rho)\rho^i \sum_{n=0}^{\infty} \rho^n && \rho^{n+i} = \rho^i \rho^n \\
&= \sum_{i=1}^{\infty} (1-\rho)\rho^i \frac{1}{1-\rho} \\
&= \sum_{i=1}^{\infty} \rho^i \\
&= \sum_{i=0}^{\infty} \rho^{i+1} && i \to i+1 \\
&= \rho \sum_{i=0}^{\infty} \rho^i \\
&= \frac{\rho}{1-\rho}.
\end{aligned}$$

Note that, since the summands are positive, we can use Fubini's theorem to justify the interchange of the summations.

s.2.5.3. Differentiate the left- and right-hand side with respect to ρ and then multiply with ρ to get

$$\frac{\rho}{(1-\rho)^2} = \sum_{n=0}^{\infty} n \rho^n.$$

Then multiply both sides by $1-\rho$ (recall that $p(n)=(1-\rho)\rho^n$).

Differentiating twice gives $\mathsf{E}\left[L^2\right]$, after which $\mathsf{V}[L]$ follows easily.

s.2.5.4.

$$M_L(s)=\mathsf{E}\left[e^{sL}\right]=\sum_{n=0}^{\infty}e^{sn}p(n)=(1-\rho)\sum_n e^{sn}\rho^n=\frac{1-\rho}{1-e^s\rho},$$

where we assume that s is such that $e^s\rho<1$. Then,

$$M_L'(s)=(1-\rho)\frac{1}{(1-e^s\rho)^2}e^s\rho.$$

Hence, $\mathsf{E}[L]=M_L'(0)=\rho/(1-\rho)$.

s.2.5.5. Starting from the result of 2.5.3, differentiating and multiplying with ρ a second time yields

$$\begin{aligned}\rho\frac{(1-\rho)^2+\rho 2(1-\rho)}{(1-\rho)^4}&=\rho\frac{1-2\rho+\rho^2+2\rho-2\rho^2}{(1-\rho)^4}\\&=\rho\frac{1-\rho^2}{(1-\rho)^4}\\&=\rho\frac{1+\rho}{(1-\rho)^3}\\&=\sum_{n=0}^{\infty}n^2\rho^n,\end{aligned}$$

and hence

$$\begin{aligned}(1-\rho)\sum_{n=0}^{\infty}n^2\rho^n=\rho\frac{1+\rho}{(1-\rho)^2}&=\frac{\rho}{(1-\rho)^2}+\frac{\rho^2}{(1-\rho)^2}\\&=\frac{2\rho^2}{(1-\rho)^2}+\frac{\rho}{(1-\rho)^2}-\frac{\rho^2}{(1-\rho)^2}\\&=\frac{2\rho^2}{(1-\rho)^2}+\rho\frac{(1-\rho)}{(1-\rho)^2}\\&=\frac{2\rho^2}{(1-\rho)^2}+\frac{\rho}{(1-\rho)^2}.\end{aligned}$$

Recall that $p(n)=(1-\rho)\rho^n$.

s.2.5.6.

$$\begin{aligned}\sum_{n=0}^{\infty}n^2\rho^n&=\sum_{n=0}^{\infty}\left(\sum_{i=1}^{\infty}2i\,\mathbb{1}_{i\le n}-n\right)\rho^n=\sum_{n=0}^{\infty}\sum_{i=0}^{\infty}2i\,\mathbb{1}_{i\le n}\rho^n-\sum_{n=0}^{\infty}n\rho^n\\&=\sum_{i=0}^{\infty}2i\sum_{n=i}^{\infty}\rho^n-\frac{\mathsf{E}[L]}{1-\rho}=\sum_{i=0}^{\infty}2i\rho^i\sum_{n=0}^{\infty}\rho^n-\frac{\mathsf{E}[L]}{1-\rho}\\&=\frac{2}{1-\rho}\sum_{i=0}^{\infty}i\rho^i-\frac{\mathsf{E}[L]}{1-\rho}=\frac{2}{(1-\rho)^2}\mathsf{E}[L]-\frac{\mathsf{E}[L]}{1-\rho}\\&=\frac{\mathsf{E}[L]}{1-\rho}\left(\frac{2}{1-\rho}-1\right)=\frac{\mathsf{E}[L]}{1-\rho}\frac{1+\rho}{1-\rho}\\&=\frac{\rho}{1-\rho}\frac{1+\rho}{(1-\rho)^2}.\end{aligned}$$

s.2.5.7. Using the results of 2.5.4 gives

$$\mathsf{E}\left[L^2\right] = M''(0) = \frac{2\rho^2}{(1-\rho)^2} + \frac{\rho}{1-\rho}.$$

s.2.5.8.

$$\mathsf{V}[L] = \mathsf{E}\left[L^2\right] - (\mathsf{E}[L])^2 = \frac{\rho(1+\rho)}{(1-\rho)^2} - \frac{\rho^2}{(1-\rho)^2} = \frac{\rho}{(1-\rho)^2}.$$

s.2.5.9. To see how large this variance is, relative to the mean number of jobs in the system, we typically consider the squared coefficient of variation (SCV). As $\mathsf{E}[L] = \rho/(1-\rho)$,

$$\frac{\mathsf{V}[L]}{(\mathsf{E}[L])^2} = \frac{1}{\rho}.$$

Thus, the SCV becomes smaller as ρ increases, but does not become lower than 1. So, realizing that the SCV of the exponential distribution is 1, the distribution of the number of jobs in the system has larger relative variability than the exponential distribution.

s.2.5.10.

$$\begin{aligned}\mathsf{P}(L \geq n) &= \sum_{k=n}^{\infty} p(k) = \sum_{k=n}^{\infty} p(0)\rho^k = (1-\rho)\sum_{k=n}^{\infty} \rho^k \\ &= (1-\rho)\rho^n \sum_{k=0}^{\infty} \rho^k = (1-\rho)\rho^n \frac{1}{1-\rho} = \rho^n.\end{aligned}$$

s.2.5.11. The fraction of time the system contains n jobs is $\pi(n)$ (by PASTA). When the system contains $n > 0$ jobs, the number in queue is one less, i.e., $n-1$.

$$\begin{aligned}\mathsf{E}\left[L_Q\right] &= \sum_{n=1}^{\infty} (n-1)\pi(n) = (1-\rho)\sum_{n=1}^{\infty} (n-1)\rho^n \\ &= \rho(1-\rho)\sum_{n=1}^{\infty} (n-1)\rho^{n-1} = \rho\sum_{n=1}^{\infty} (n-1)\pi(n-1) \\ &= \rho\sum_{n=0}^{\infty} n\pi(n) = \rho\frac{\rho}{1-\rho}.\end{aligned}$$

Another way to get the same result is by splitting:

$$\begin{aligned}\mathsf{E}\left[L_Q\right] &= \sum_{n=1}^{\infty} (n-1)\pi(n) = \sum_{n=1}^{\infty} n\pi(n) - \sum_{n=1}^{\infty} \pi(n) \\ &= \mathsf{E}[L] - (1-\pi(0)) = \mathsf{E}[L] - \rho.\end{aligned}$$

2.6 $M(n)/M(n)/1$ QUEUE

Theory and Exercises

As it turns out, many more single-server queueing situations than the $M/M/1$ queue can be analyzed by making a judicious choice of $\lambda(n)$ and $\mu(n)$ in the level-crossing equations (2.4.6). For these queueing systems, we just present the results. In the exercises we ask you to derive the formulas—the main challenge is not to make computational errors.

It is important to realize that the inter-arrival times and service times need to be memoryless for the analysis below; the rates, however, may depend on the number of jobs in the system. Specifically, we require that for all s and t,

$$\mathsf{P}\left(A_{A(t)+1} \le t+s \,\middle|\, L(t)=n\right) = 1-e^{-\lambda(n)s},$$

where we use that $A_{A(t)}+g$ is the arrival time of the next job after time t. Similarly, we assume for all t and s,

$$\mathsf{P}\left(D_{D(t)+1} \le t+s \,\middle|\, L(t)=n\right) = 1-e^{-\mu(n)s}.$$

2.6.1. *Model the $M/M/1/K$ queue in terms of an $M(n)/M(n)/1$ queue and compute $p(K)$, i.e., the fraction of time that the system is full.*

2.6.2. *Show that as $K \to \infty$, the performance measures of the $M/M/1/K$ converge to those of the $M/M/1$ queue.*

2.6.3. *Model the $M/M/c$ queue in terms of an $M(n)/M(n)/1$ queue and compute $\mathsf{E}\left[L_Q\right]$.*

2.6.4. *Check that the performance measures of the $M/M/c$ queue reduce to those of the $M/M/1$ queue if $c=1$.*

2.6.5. *It should be clear that the $M/M/c$ queue is a bit harder to analyze than the $M/M/1$ queue, at least the expressions are more extensive. It is tempting to approximate the $M/M/c$ queue by an $M/M/1$ queue with a server that works c times as fast. As we now have the formulas for the $M/M/c$ queue and the $M/M/q$ queue we can use these to obtain some basic understanding of the difference.*

Let us therefore consider a numerical example. Suppose that we have an $M/M/3$ queue, with arrival rate $\lambda=5$ per day and $\mu=2$ per server, and we compare it to an $M/M/1$ with the same arrival rate but with a service rate of $\mu=3\cdot 2=6$. Make a graph of the ratios of $\mathsf{E}[L]$ and $\mathsf{E}\left[L_Q\right]$ of both models as a function of ρ. Explain why these ratios become 1 as $\rho \uparrow 1$.

2.6.6. *Model the $M/M/c/c$ queue in terms of an $M(n)/M(n)/1$ queue and determine the performance measures. This model is also known as the Erlang B-formula and is often used to determine the number of beds at hospitals, where the beds act as servers and the patients as jobs.*

2.6.7. *Take the limit $c \to \infty$ in the $M/M/c$ queue (or the $M/M/c/c$ queue) and obtain the performance measures for the $M/M/\infty$ queue, i.e., a queueing system with ample servers.*

2.6.8. *Show that the $M/M/\infty$ queue is stable for any finite λ.*

2.6.9. *Why is $\mathsf{E}[L]=\rho$ for the $M/M/\infty$ queue?*

2.6.10. *Consider the $M/M/2/3$ queue with arrival rate λ and service rate μ (thus, at most 2 jobs can be in service and 1 in queue). Derive first the level-crossing equations for this queueing system, then derive closed form expressions for the state probabilities in steady state.*

2.6.11. *[Multi-server queue with blocking] Consider the $M/M/c/c+K$ queue in which at most c jobs can be in service and K in queue. Try to derive the steady state probabilities $p(0)$, $p(1)$,.... You do not have to compute the normalization constant G.*

2.6.12. *Derive the steady state probabilities $p(n)$ for a single-server queue with a finite calling population with N jobs, i.e., jobs that are in service cannot arrive to the system. Check the answer you obtained for the cases $N = 1$ and $N = 2$. What happens if $N \to \infty$? Interpret the results.*

2.6.13. *Give an example of a system with a finite calling population.*

2.6.14. *Derive the steady state probabilities $p(n)$ for a queue with a finite calling population with N jobs and N servers, i.e., the number of servers in the queueing system is equal to the size of the calling population. What happens if $N \to \infty$?*

Finally, we consider queues with *balking*, that is, queues in which customers leave when they find the queue too long at the moment they arrive. A simple example model with customer balking is given by

$$\lambda(n) = \begin{cases} \lambda, & \text{if } n = 0, \\ \lambda/2, & \text{if } n = 1, \\ \lambda/4, & \text{if } n = 2, \\ 0, & \text{if } n > 2, \end{cases}$$

and $\mu(n) = \mu$.

Observe that here we make a subtle implicit assumption; in Section 2.7 we elaborate on this assumption. To make the problem clear, note that balking customers *decide at the moment they arrive* to either join or leave; in other words, they decide based on what they 'see upon arrival'. In yet other words, they make decisions based on the state of the system at arrival moments, not on time-averages. However, the notion of $p(n)$ is a long-run *time-average*, and is typically not the same as what customers 'see upon arrival'. As a consequence, the performance measure $\mathsf{P}(L \le n)$ is not necessarily in accordance with the perception of customers. To relate these two 'views', i.e., time-average versus observer-average, we need a new concept, *PASTA*, to be developed in Section 2.7.

2.6.15. *In what way is a queueing system with balking, at level b say, different from a queueing system with finite calling population of size b?*

Hints

h.2.6.1. Take $\lambda(n) = \lambda \mathbb{1}_{n<K}$, $\mu(n) = \mu$, use the equations around (2.4.7).

h.2.6.2. Use that $\sum_{i=0}^{n} x^i = (1 - x^{n+1})/(1 - x)$. BTW, is it necessary for this expression to be true that $|x| < 1$? What should you require for $|x|$ when you want to take the limit $n \to \infty$?

h.2.6.3. Take $\lambda(n) = \lambda$, and $\mu(n) = \min\{n, c\}\mu$.

h.2.6.4. Fill in $c = 1$. Realize that this is a check on the formulas.

h.2.6.7. Use that for any x, $x^n/n! \to 0$ as $n \to \infty$.

h.2.6.10. Think about what would be the appropriate model choices for $\lambda(n)$ and $\mu(n)$ and use the level-crossing equations $\lambda(n)p(n) = \mu(n+1)p(n+1)$. For instance, realize that $\lambda(3) = 0$: the system cannot contain more than 3 jobs, hence a state with 4 jobs must be impossible. We can achieve that by setting $\lambda(3) = 0$. For the service rate, how many servers are busy when the system contains 2 or more jobs? What does this say about $\mu(k)$ for $k = 2$ or $k = 3$.

h.2.6.11. Use $\lambda(n)p(n) = \mu(n+1)p(n+1)$ and find suitable expressions for $\lambda(n)$ and $\mu(n+1)$.

h.2.6.12. Use $\lambda(n)p(n) = \mu(n+1)p(n+1)$, and realize that for this case $\lambda(n) = (N-n)\lambda$ and $\mu(n) = \mu$.

Solutions

s.2.6.1. Note that

$$1 = \sum_{i=0}^{K} p(i) = p(0)\sum_{i=0}^{K} \rho^i = p(0)\frac{1-\rho^{K+1}}{1-\rho}.$$

Thus,

$$p(n) = \frac{\rho^n}{G}, \quad 0 \le n \le K, \tag{2.6.1a}$$

$$p(K) = \frac{1-\rho}{1-\rho^{K+1}}\rho^K. \tag{2.6.1b}$$

s.2.6.2. To take the limit $K \to \infty$—mind, not the limit $n \to \infty$—, write

$$G = \frac{1-\rho^{K+1}}{1-\rho} = \frac{1}{1-\rho} - \frac{\rho^{K+1}}{1-\rho}.$$

Since $\rho^{K+1} \to 0$ as $K \to \infty$ (recall, $\rho < 1$), we get

$$G \to \frac{1}{1-\rho},$$

as $K \to \infty$. Therefore, $p(n) = \rho^n/G \to \rho^n(1-\rho)$, and the latter are the steady-state probabilities of the $M/M/1$ queue. Finally, if the steady-state probabilities are the same, the performance measures (which are derived from $p(n)$) must be the same.

s.2.6.3. First we use the hint to establish a generic relation for $p(n)$. Taking $\rho = \lambda/(c\mu)$,

$$\begin{aligned} p(n) &= \frac{\lambda(n-1)}{\mu(n)}p(n-1) = \frac{\lambda}{\min\{c,n\}\mu}p(n-1) = \frac{1}{\min\{c,n\}}(c\rho)p(n-1) \\ &= \frac{1}{\min\{c,n\}\min\{c,n-1\}}(c\rho)^2 p(n-2) \\ &= \frac{1}{\Pi_{k=1}^{n}\min\{c,k\}}(c\rho)^n p(0). \end{aligned}$$

Thus, if $n < c$:

$$p(n) = \frac{(c\rho)^n}{n!}p(0). \tag{2.6.2}$$

If $n \ge c$:

$$\begin{aligned} p(n) &= \frac{1}{\Pi_{k=1}^{c}k \cdot \Pi_{k=c+1}^{n}c}(c\rho)^n p(0) \\ &= \frac{1}{c!c^{n-c}}c^n\rho^n p(0) \\ &= \frac{c^c}{c!}\rho^n p(0). \end{aligned}$$

To obtain the normalization constant G,

$$\begin{aligned}
1 &= \sum_{n=0}^{\infty} p(n) = \sum_{n=0}^{c-1} p(n) + \sum_{n=c}^{\infty} p(n) \\
&= p(0) \sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + p(0) \sum_{n=c}^{\infty} \frac{c^c}{c!} \rho^n \\
&= p(0) \sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + p(0) \sum_{n=c}^{\infty} \frac{(c\rho)^c}{c!} \rho^{n-c} \\
&= p(0) \sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + p(0) \frac{(c\rho)^c}{c!} \sum_{n=0}^{\infty} \rho^n \\
&= p(0) \sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + p(0) \frac{(c\rho)^c}{c!(1-\rho)}.
\end{aligned}$$

Hence,

$$G = \sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + \frac{(c\rho)^c}{c!(1-\rho)}. \tag{2.6.3}$$

Next,

$$\begin{aligned}
\mathsf{E}\left[L_Q\right] &= \sum_{n=c}^{\infty} (n-c) p(n) \\
&= \sum_{n=c}^{\infty} (n-c) \frac{c^c}{c!} \rho^n p(0) \\
&= \frac{c^c \rho^c}{Gc!} \sum_{n=c}^{\infty} (n-c) \rho^{n-c} \\
&= \frac{c^c \rho^c}{Gc!} \sum_{n=0}^{\infty} n \rho^n = \frac{c^c \rho^c}{Gc!} \frac{\rho}{(1-\rho)^2}.
\end{aligned}$$

The derivation of the expected number of jobs in service becomes easier if we pre-multiply the normalization constant G:

$$\begin{aligned}
G\,\mathsf{E}[L_S] &= G \left(\sum_{n=0}^{c} n p(n) + \sum_{n=c+1}^{\infty} c p(n) \right) \\
&= \sum_{n=1}^{c} n \frac{(c\rho)^n}{n!} + \sum_{n=c+1}^{\infty} c \frac{c^c \rho^n}{c!} = \sum_{n=1}^{c} \frac{(c\rho)^n}{(n-1)!} + \frac{c^{c+1}}{c!} \sum_{n=c+1}^{\infty} \rho^n \\
&= \sum_{n=0}^{c-1} \frac{(c\rho)^{n+1}}{n!} + \frac{(c\rho)^{c+1}}{c!} \sum_{n=0}^{\infty} \rho^n = c\rho \left(\sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + \frac{(c\rho)^c}{c!(1-\rho)} \right).
\end{aligned}$$

Observe that the right-hand side is precisely equal to $\rho c G$, and hence,

$$\mathsf{E}[L_S] = c\rho = \frac{\lambda}{\mu}.$$

s.2.6.4. Take $c = 1$

$$p(n) = \frac{1}{G} \frac{(c\rho)^0}{0!} = \frac{1}{G}, \quad n = 0, \ldots, 1-1 \tag{2.6.4a}$$

$$p(n) = \frac{1}{G} \frac{c^c \rho^n}{c!} = \frac{1}{G} \frac{1^1 \rho^n}{1!} = \frac{\rho^n}{G}, \quad n = 1, 1+1, \ldots \tag{2.6.4b}$$

$$G = \sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + \frac{(c\rho)^c}{(1-\rho)c!} = \sum_{n=0}^{0} \frac{\rho^0}{0!} + \frac{\rho}{(1-\rho)} = 1 + \frac{\rho}{1-\rho} = \frac{1}{1-\rho}, \tag{2.6.4c}$$

$$\mathsf{E}[L_Q] = \frac{(c\rho)^c}{c!G}\frac{\rho}{(1-\rho)^2} = \frac{\rho}{1/(1-\rho)}\frac{\rho}{(1-\rho)^2} = \frac{\rho^2}{1-\rho}, \tag{2.6.4d}$$

$$\mathsf{E}[L_S] = \sum_{n=0}^{c} np(n) + \sum_{n=c+1}^{\infty} cp(n) = p(1) + 1\sum_{n=2}^{\infty} p(n) = 1 - p(0) = \rho. \tag{2.6.4e}$$

Everything is in accordance to the formulas we derived earlier for the $M/M/1$ queue.

s.2.6.5. I implement the formulas of 2.6.3 in Python. First the results for the $M/M/3$ queue.

```
>>> from math import exp, factorial

>>> labda = 5
>>> mu = 2
>>> c = 3

>>> rho = labda / mu / c
>>> rho
0.8333333333333334

>>> G = sum((c * rho)**n / factorial(n) for n in range(c))
>>> G += (c * rho)**c / ((1 - rho) * factorial(c))
>>> G
22.250000000000004

>>> ELQ = (c * rho)**c / (factorial(c) * G) * rho / (1 - rho)**2
>>> ELQ
3.511235955056181
>>> ELS = rho * c
>>> ELS
2.5
>>> EL = ELQ + ELS
>>> EL
6.011235955056181
```

Now for the $M/M/1$ queue:

```
>>> labda = 5
>>> c = 3
>>> mu = 2*c

>>> rho = labda / mu
>>> rho
0.8333333333333334

>>> ELQ = rho**2/(1-rho)
>>> ELQ
```

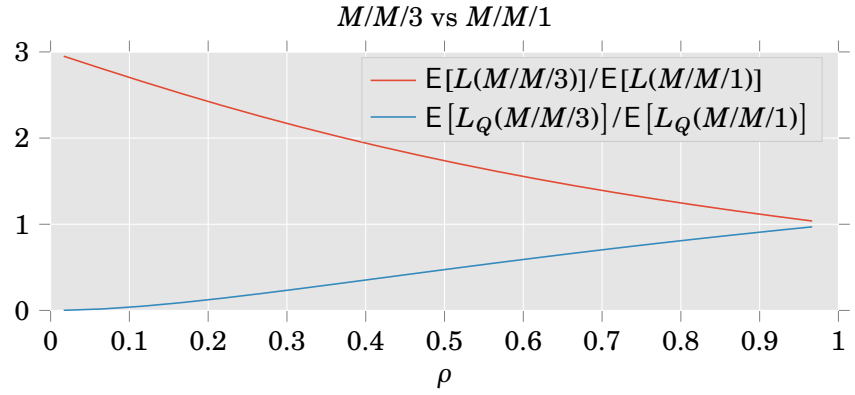
```
4.166666666666668
>>> ELS = rho
>>> ELS
0.8333333333333334
>>> EL = ELS + ELQ
>>> EL
5.000000000000001

>>> rho/(1-rho) # this must also be EL, just a check
5.000000000000002
```

Note the last check. As a rule, you should always compare your results with known results. BTW, that is one of the reasons I prefer to code the formulas instead of using a calculator. Testing with code is relatively easy, whereas with a calculator it is impossible. (You simply can't check what you typed at the calculator.)

So, returning to the results, as expected, the number of jobs in queue is smaller for the $M/M/3$ queue, but the number in service is higher.

To put things in a larger perspective, see the figure below where we plot the ratio of the queue lengths and the system length as functions of ρ. We see, in case of high load, that $\mathsf{E}\left[L_Q\right]$ and $\mathsf{E}[L]$ are nearly the same for both systems. This is as expected: when the load is high, most jobs should be in the queue. Therefore, $\mathsf{E}\left[L_Q\right]/\mathsf{E}[L] \to 1$ as $\rho \to 1$. When ρ is small, the difference is quite large. This is also reasonable, because the service time in the fast $M/M/1$ is 3 times as small as the service time in the $M/M/3$ queue. Hence, as ρ is small, the time in the system is dominated by service time, as there is hardly any queueing time, if at all. Thus, there must be more jobs in the system on average in the $M/M/3$ queue than in the fast $M/M/1$ queue.



The code can be found on `github` in the `progs` directory.

s.2.6.6. Take, $\lambda(n) = \lambda$ if $n < c$, and $\lambda(n) = 0$ for $n \geq c$. Also, let, $\mu(n) = n\mu$ for $n \leq c$. (And n can never be larger than c, since $\lambda(n) = 0$ for $n \geq c$.) Define $\rho = \lambda/(c\mu)$. Then, we see that $p(n) = p(0)(c\rho)^n/n!$. For the normalization

$$1 = \sum_{n=0}^{c} p(n) = p(0) \sum_{n=0}^{c} \frac{(c\rho)^n}{n!}.$$

Thus, the normalization constant $G = \sum_{n=0}^{c} \frac{(c\rho)^n}{n!}$, and $p(0) = G^{-1}$.

Since there are as many servers as places available in the system, $\mathsf{E}[L_Q] = 0$. The expected number of servers busy is

$$\begin{aligned}
\mathsf{E}[L_S] &= \sum_{n=0}^{c} np(n) = \sum_{n=1}^{c} np(n) \\
&= G^{-1} \sum_{n=1}^{c} n \frac{(\lambda/\mu)^n}{n!} = G^{-1} \sum_{n=1}^{c} \frac{(\lambda/\mu)^n}{(n-1)!} \\
&= \frac{\lambda}{\mu G} \sum_{n=0}^{c-1} \frac{(\lambda/\mu)^n}{n!} = \frac{\lambda}{G\mu} \left(G - \frac{(\lambda/\mu)^c}{c!} \right) \\
&= \frac{\lambda}{\mu} \left(1 - \frac{1}{G} \frac{(\lambda/\mu)^c}{c!} \right) \\
&= \frac{\lambda}{\mu} (1 - p(c)).
\end{aligned}$$

This can be explained as follows: $\lambda(1-p(c))$ is the rate of accepted jobs (since a fraction $p(c)$ is lost). Thus, the load is $\lambda(1-p(c))/\mu$, and the load is the fraction of time the servers are busy.

s.2.6.7. By taking the limit $c \to \infty$, note first that in (2.6.3),

$$\frac{(c\rho)^c}{(1-\rho)c!} = \frac{(\lambda/\mu)^c}{(1-\rho)c!} \to 0, \quad \text{as } c \to \infty.$$

Hence

$$G = \sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + \frac{(c\rho)^c}{(1-\rho)c!} \to \sum_{n=0}^{\infty} \frac{(c\rho)^n}{n!} = e^{\lambda/\mu}.$$

Next, for any fixed n, eventually $c > n$, and then, as $\rho = \lambda/(\mu c)$,

$$p(n) = \frac{1}{G} \frac{(c\rho)^n}{n!} = \frac{1}{G} \frac{(\lambda/\mu)^n}{n!} \to e^{-\lambda/\mu} \frac{(\lambda/\mu)^n}{n!}, \quad \text{as } c \to \infty.$$

Moreover, there is no fixed n such that $n > c$. Thus, the probabilities in 2.6.3 are no longer present. Thus, we see that the number of busy servers in the $M/M/\infty$ queue is Poisson distributed with parameter λ/μ, and $\mathsf{E}[L] = \mathsf{E}[L_S] = \lambda/\mu$. Observe that now λ/μ has no longer the interpretation of the fraction of time the server(s) are busy; it is the average number of busy servers.

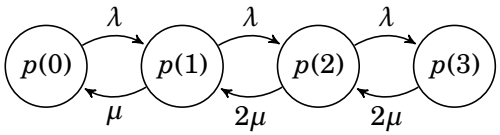
We mention in passing—but do not prove it—that the same results also hold for the $M/G/\infty$ queue with $\lambda\,\mathsf{E}[S]$ rather than λ/μ.

s.2.6.8. No matter how many jobs are in service, there is always another free server available when a new job arrives. Thus, jobs never have to wait in queue, and only spend time in service. Since $\mathsf{E}[S] < \infty$ by assumption, jobs spend a finite time (with probability one) at a server.

s.2.6.9. Write $\rho = \lambda/\mu$. Then, from the formulas for the $M/M/\infty$ queue, it follows that $p(n) = e^{-\rho}\rho^n/n!$. Interestingly, we see that this is equal to $\mathsf{P}(N = n)$ where N is a Poisson r.v. with parameter ρ. Thus, the number in the system L is Poisson distributed with parameter ρ, thus $\mathsf{E}[L] = \rho$.

Another way to see that $\mathsf{E}[L] = \rho$ is by noting that in the $M/M/\infty$ queue jobs do not interact with each other in the queue. When they arrive, there is always a free server available. Since work arrives at rate ρ, and all jobs are in service simultaneously, the average number of busy servers must also be ρ.

s.2.6.10. Use the figure below. Make sure you understand why $\mu(2) = 2\mu$ and so on.



From this figure it follows right away that:

$$\lambda p(0) = \mu p(1)$$
$$\lambda p(1) = 2\mu p(2)$$
$$\lambda p(2) = 2\mu p(3).$$

Then, from the above, with $\rho = \lambda/\mu$:

$$p(1) = \rho p(0),$$
$$p(2) = (\rho/2)p(1) = (\rho^2/2)p(0),$$
$$p(3) = (\rho/2)p(2) = (\rho^3/4)p(0).$$

Now we normalize to find $p(0)$. Thus, we want that:

$$1 = p(0) + p(1) + p(2) + p(3) = p(0)\left(1 + \rho + \frac{\rho^2}{2} + \frac{\rho^3}{4}\right),$$

hence,

$$p(0) = (1 + \rho + \rho^2/2 + \rho^3/4)^{-1}.$$

s.2.6.11. $\lambda(n) \equiv \lambda$ for all $n < c + K$. When $n = c + K$, $\lambda(n) = 0$, since then the system is full, and all arriving jobs will be dropped; in other words, there will still be jobs arriving to the system when $L = c + K$, but these jobs will be rejected, hence cannot generate a transition from state $c + K$ to $c + K + 1$. When $n < c$, $\mu(n) = n\mu$ since only n servers are active/occupied when the system contains n jobs. When $n \geq c$, $\mu(n) = c\mu$. Thus, using $\rho = \lambda/(c\mu)$, for $n < c$,

$$p(n) = \frac{\lambda}{n\mu} p(n-1) = \frac{(\lambda/\mu)^n}{n!} p(0) = \frac{(c\rho)^n}{n!} p(0).$$

For $c \leq n \leq c + K$ and using the above to get $p(c-1)$:

$$\begin{aligned} p(n) &= \frac{\lambda}{c\mu} p(n-1) = \rho p(n-1) = \rho^2 p(n-2) = \dots \\ &= \rho^{n-c+1} p(c-1) = \rho^{n-c+1} \frac{(c\rho)^{c-1}}{(c-1)!} p(0) \\ &= \rho^n \frac{(c)^{c-1}}{(c-1)!} p(0) = \rho^n \frac{(c)^{c-1} c}{(c-1)! c} p(0) = \frac{c^c \rho^n}{c!} p(0). \end{aligned}$$

The normalization is trivial, numerically at least.

s.2.6.12. Take $\lambda(n) = (N-n)\lambda$ and $\mu(n) = \mu$, and solve (2.4.6) and (2.4.8). Thus:

$$p(n+1) = \frac{(N-n)\lambda}{\mu} p(n) = \rho(N-n)p(n)$$

$$
\begin{aligned}
&= \rho^2(N-n)(N-(n-1))p(n-1)\\
&= \rho^3(N-n)(N-(n-1))(N-(n-2))p(n-2)\\
&= \rho^{n+1}(N-n)(N-(n-1))\cdots(N-(0))p(0)\\
&= \rho^{n+1}\frac{N!}{(N-(n+1))!}p(0).
\end{aligned}
$$

Next, we need to normalize this. Observe that $p(N+1) = p(N+2) = \ldots = 0$ since there are just N customers, so that the system can never contain more than N customers. Thus, we want $p(0)$ to be such that

$$1 = \sum_{n=0}^{N} p(n) = p(0)\sum_{n=0}^{N} \rho^n \frac{N!}{(N-n)!}.$$

We see from this that $p(0)$ times some constant must be 1. Hence, dividing by this constant, we get

$$p(0) = \left(\sum_{n=0}^{N} \rho^n \frac{N!}{(N-n)!}\right)^{-1}.$$

I asked WolframAlpha to simplify this, but the answer I got was not particularly revealing.

s.2.6.13. A finite calling population occurs for instance at a factory with a number of machines. When a machine breaks down, it becomes a (repair) job at the repair department. Thus, a break down forms an arrival at the repair shop. The mechanics at the repair department form a set of parallel servers. Typically, the number of machines is quite small, 10 or so, and when a machine is 'down', i.e., broken, it cannot break again. Hence, when 2, say, machines are in repair, the number of 'customers' that can arrive to the queueing system is only 8.

s.2.6.14. Take $\lambda(n) = (N-n)\lambda$ and $\mu(n) = n\mu$. Then

$$
\begin{aligned}
p(n+1) &= \frac{\lambda(n)}{\mu(n+1)}p(n) = \frac{(N-n)\lambda}{(n+1)\mu}p(n) = \frac{(N-n)(N-(n-1))}{(n+1)n}\frac{\lambda^2}{\mu^2}p(n-1)\\
&= \frac{N!}{(N-(n+1))!}\frac{1}{(n+1)!}\rho^{n+1}p(0) = \binom{N}{n+1}\rho^{n+1}p(0).
\end{aligned}
$$

Hence, after normalization, i.e., requiring that $p(0)$ is such that $\sum_{n=0}^{N} p(n) = 1$, so that $p(0) = \left(\sum_{k=0}^{N} \rho^k \binom{N}{k}\right)^{-1}$, the final result becomes

$$p(n) = \frac{\rho^n \binom{N}{n}}{\sum_{k=0}^{N} \rho^k \binom{N}{k}}.$$

s.2.6.15. In a queueing system with balking, customers may decide to balk at a level b. Thus, whether only b customers are admitted to the system (i.e., blocked), or balk at level b, the effect is the same: the number of people in the system remains at or below b. However, a fraction of the customers may already balk at lower levels, like in the example above, so that the arrival stream is 'thinned' due to balking customers. In that respect, a queueing system with balking behaves differently.

2.7 POISSON ARRIVALS SEE TIME AVERAGES

Theory and Exercises

Suppose the following limit exists:

$$\pi(n) = \lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} \mathbb{1}_{L(A_k-)=n}, \tag{2.7.1}$$

then $\pi(n)$ is the long-run fraction of jobs that observe n customers in the system at the moment an arbitrary job arrives. It is natural to ask whether $\pi(n)$ and $p(n)$, as defined by (2.4.2), are related, that is, whether what customers see upon arrival is related to the time-average behavior of the system. In this section we will derive the famous *Poisson arrivals see time averages* (*PASTA*) condition that ensures that $\pi(n) = p(n)$ if jobs arrive in accordance with a Poisson process.

Since $A(t) \to \infty$ as $t \to \infty$, it is reasonable that (see 2.7.6 for a proof)

$$\begin{aligned} \pi(n) &= \lim_{t\to\infty} \frac{1}{A(t)} \sum_{k=1}^{A(t)} \mathbb{1}_{L(A_k-)=n} = \lim_{t\to\infty} \frac{1}{A(t)} \sum_{k=1}^{\infty} \mathbb{1}_{A_k\le t, L(A_k-)=n} \\ &= \lim_{t\to\infty} \frac{A(n,t)}{A(t)}, \end{aligned} \tag{2.7.2}$$

where we use (2.4.1a) in the last row. But, with (2.1.1),

$$\frac{A(n,t)}{t} = \frac{A(t)}{t}\frac{A(n,t)}{A(t)} \to \lambda\pi(n), \quad \text{as } t\to\infty, \tag{2.7.3}$$

while by (2.4.4),

$$\frac{A(n,t)}{t} = \frac{A(n,t)}{Y(n,t)}\frac{Y(n,t)}{t} \to \lambda(n)p(n), \quad \text{as } t\to\infty.$$

Thus

$$\lambda\pi(n) = \lambda(n)p(n). \tag{2.7.4}$$

This leads to our final result:

$$\lambda(n) = \lambda \iff \pi(n) = p(n).$$

This means that if the arrival rate does not depend on the state of the system, i.e., $\lambda(n) = \lambda$, the sample probabilities $\{\pi(n)\}$ are equal to the time-average probabilities $\{p(n)\}$. In other words, the customer perception at arrival moments is the same as the server perception.

As the next exercises show, this property is not satisfied in general. However, when the arrival process is Poisson we have that $\lambda(n) = \lambda$. This fact is called *PASTA: Poisson Arrivals See Time Averages*. Thus, for the $M/M/1$ queue in particular,

$$\pi(n) = p(n) = (1-\rho)\rho^n.$$

2.7.1. *Show for the case of 2.4.6 that* $\pi(0) = 1$ *and* $\pi(n) = 0$*, for* $n > 0$*.*

2.7.2. *Check that* (2.7.4) *holds for the system of 2.7.1.*

With the above reasoning, we can also establish a relation between $\pi(n)$ and the statistics of the system as obtained by the departures. Define, analogous to (2.7.2),

$$\delta(n) = \lim_{t\to\infty} \frac{D(n,t)}{D(t)} \tag{2.7.5}$$

as the long-run fraction of jobs that leave n jobs *behind*. From (2.4.3)

$$\frac{A(t)}{t}\frac{A(n,t)}{A(t)} = \frac{A(n,t)}{t} \approx \frac{D(n,t)}{t} = \frac{D(t)}{t}\frac{D(n,t)}{D(t)}.$$

Taking limits at the left and right, and using (2.1.3), we obtain for (queueing) systems in which customers arrive and leave as single units that

$$\lambda\pi(n) = \delta\delta(n). \tag{2.7.6}$$

Thus, if the system is rate-stable and transitions occur one-by-one, the statistics obtained by arrivals is the same as statistics obtained by departures, i.e.,

$$\lambda = \delta \iff \pi(n) = \delta(n). \tag{2.7.7}$$

Note that the derivation of this fact does not depend on the distribution of the inter-arrival or service times. Hence, it is true for the rate-stable $G/G/1$ queue.

2.7.3. *When $\lambda \neq \delta$, is $\pi(n) \geq \delta(n)$?*

2.7.4. *Show that*

$$\lambda\pi(n) = \lambda(n)p(n) = \mu(n+1)p(n+1) = \delta\delta(n).$$

What is the important condition for this to be true?

2.7.5. *Use PASTA and the balance equations of the $M/M/1$ queue to derive that $(\lambda+\mu)\pi(n) = \lambda\pi(n-1) + \mu\pi(n+1)$.*

2.7.6. *There is a subtle problem in the transition from* (2.7.1) *to* (2.7.2) *and the derivation of* (2.7.3)*: $\pi(n)$ is defined as a limit over arrival epochs while in $A(n,t)/t$ we take the limit over time. Now the observant reader might ask why these limits should relate at all. Use the renewal reward theorem to show that* (2.7.2) *is valid.*

With the PASTA property we can determine the distribution of the inter-departure times of the $M/M/1$ queue. Observing that in a network of queues the departures from one queueing station form the arrivals at another station, we can use this result to analyze networks of queues

2.7.7. *Try to prove* Burke's law *which states that the departure process of the $M/M/1$ queue is a Poisson process with rate λ.*

2.7.8. *Why is the output rate of the (stable) $M/M/1$ queue equal to λ and not μ?*

2.7.9. *Why is $\mu e^{-\mu t}$ not a reasonable density for the inter-departure times? In fact, the simplest guess for the inter-departure density might be $\lambda e^{-\lambda t}$; so this is what we will try to prove below. We will focus on departure moments and use* (2.7.7)*, in particular that departures 'see' what arrivals 'see', i.e., $\delta(n) = \pi(n)$, and PASTA.*

2.7.10. *Show that the probability that a job leaves behind a busy station is ρ, hence $1-\rho$ is the probability to leave an idle server behind.*

2.7.11. *If job $n-1$, say, leaves behind an empty system, show that the expected time until the next departure is $\mathsf{E}[D_n - D_{n-1}] = 1/\lambda + 1/\mu$.*

2.7.12. *Show that the density of $D_n - D_{n-1}$ is*

$$f_{X+S}(t) = \frac{\lambda\mu}{\lambda-\mu}(e^{-\mu t} - e^{-\lambda t})$$

if the server is idle after D_{n-1}.

2.7.13. *Show that when the queue is not empty at a departure time, the density of the next inter-departure time is $f_D(t) = \mu e^{-\mu t}$.*

2.7.14. *Use conditioning on the server being idle or busy at a departure to show that the density of the inter-departure time is $\lambda e^{-\lambda t}$.*

Hints

h.2.7.3. Use that $\lambda \geq \delta$ always holds. Thus, when $\lambda \neq \delta$, it must be that $\lambda > \delta$. What are the consequences of this inequality; how does the queue length behave as a function of time?

h.2.7.4. Check all definitions of $Y(n,t)/t$ and so on.

h.2.7.5. Consider some state n (not a level) and count all transitions that 'go in and out of' this state. Specifically, $A(n,t) + D(n-1,t)$ counts all transitions out of state n: $A(n,t)$ counts the number of arrivals that see n in the system upon arrival, hence immediately after such arrivals the system contains $n+1$ jobs; likewise, $D(n-1,t)$ counts all jobs that leave $n-1$ jobs behind, hence immediately before such jobs depart the system contains n jobs. In a similar way, $A(n-1,t) + D(n,t)$ counts all transitions into state n (Recall once again, $D(n,t)$ counts the jobs that leave n behind. Hence, when such departures occur, state n is entered). Now use that 'what goes in must go out'.

h.2.7.6. Check that the conditions of the renewal reward theorem are satisfied in the above proof of (2.7.3). Then define

$$Y(t) := A(n,t) = \sum_{k=1}^{A(t)} \mathbb{1}_{L(A_k-)=n}$$
$$X_k := Y(A_k) - Y(A_{k-1}) = A(n,A_k) - A(n,A_{k-1}) = \mathbb{1}_{L(A_k-)=n}.$$

h.2.7.11. After job $n-1$ left, job n has to arrive, so we need to wait first for this inter-arrival time. Then job n must be served. This adds up to $1/\lambda + 1/\mu$.

h.2.7.14. Conditioning leads to

$$f_D(t) = f_{X+S}(t)\mathsf{P}(\text{server is idle}) + f_S(t)\mathsf{P}(\text{ server is busy }) = (1-\rho)f_{X+S}(t) + \rho\mu e^{-\mu t}.$$

Now use the above exercises to simplify.

Solutions

s.2.7.1. All arrivals see an empty system. Hence $A(0,t)/A(t) \approx (t/2)/(t/2) = 1$, and $A(n,t) = 0$ for $n > 0$. Thus, $\pi(0) = \lim_{t\to\infty} A(0,t)/A(t) = 1$ and $\pi(n) = 0$ for $n > 0$. Recall from the other exercises that $p(0) = 1/2$. Hence, statistics as obtained via time averages are not necessarily the same as statistics obtained at arrival moments (or any other point process).

s.2.7.2. From the relevant previous exercises, $\lambda = \lim_{t\to\infty} A(t)/t = 1/2$. $\lambda(0) = 1$, $p(0) = 1/2$, and $\pi(0) = 1$. Hence,

$$\lambda\pi(0) = \lambda(0)p(0) \implies \frac{1}{2} \times 1 = 1 \times \frac{1}{2}.$$

For $n > 0$ it's easy, everything is 0.

s.2.7.3. The assumptions lead us to conclude that $\lambda > \delta$. As a consequence, the queue length must increase in the long run (jobs come in faster than they leave). Therefore, $A(n,t)/t \to 0$ for all n, and also $D(n,t)/t \to 0$. Consequently, $\pi(n) = \delta(n) = 0$, which is the only sensible reconciliation with (2.7.6).

s.2.7.4. The important condition is that transitions occur as single steps. In other words, the relation is true for processes with *one-step transitions*, i.e., when $|A(n,t) - D(n,t)| \leq 1$. In that case,

$$\begin{aligned}
\frac{A(n,t)}{t} &= \frac{A(n,t)}{A(t)}\frac{A(t)}{t} \to \pi(n)\lambda \\
\frac{A(n,t)}{t} &= \frac{A(n,t)}{Y(n,t)}\frac{Y(n,t)}{t} \to \lambda(n)p(n) \\
\frac{D(n,t)}{t} &= \frac{D(n,t)}{Y(n+1,t)}\frac{Y(n+1,t)}{t} \to \mu(n+1)p(n+1) \\
\frac{D(n,t)}{t} &= \frac{D(n,t)}{D(t)}\frac{D(t)}{t} \to \delta(n)\delta.
\end{aligned}$$

s.2.7.5. By the hint, the difference between the 'out transitions' and the 'in transitions' is at most 1 for all t. Thus, we can write

$$\begin{aligned}
\text{transitions out} &\approx \text{transitions in} \iff \\
A(n,t) + D(n-1,t) &\approx A(n-1,t) + D(n,t) \iff \\
\frac{A(n,t) + D(n-1,t)}{t} &\approx \frac{A(n-1,t) + D(n,t)}{t} \iff \\
\frac{A(n,t)}{t} + \frac{D(n-1,t)}{t} &\approx \frac{A(n-1,t)}{t} + \frac{D(n,t)}{t}.
\end{aligned}$$

Using the ideas of Section 2.4 this becomes for $t \to \infty$,

$$(\lambda(n) + \mu(n))p(n) = \lambda(n-1)p(n-1) + \mu(n+1)p(n+1).$$

Since we are concerned here with the $M/M/1$ queue we have that $\lambda(n) = \lambda$ and $\mu(n) = \mu$, and using PASTA we have that $p(n) = \pi(n)$. We are done.

s.2.7.6. First we check the conditions. The counting process here is $\{A(t)\}$ and the epochs at which $A(t)$ increases are $\{A_k\}$. By assumption, $A_k \to \infty$, hence $A(t) \to \infty$ as $t \to \infty$. Moreover, by assumption $A(t)/t \to \lambda$. Also $A(n,t)$ is evidently non-decreasing and $A(n,t) \to \infty$ as $t \to \infty$.

From the definitions in the hint,

$$X = \lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} X_k = \lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} \mathbb{1}_{L(A_k-)=n} = \pi(n).$$

Since $Y = \lim_{t\to\infty} Y(t)/t = \lim_{t\to\infty} A(n,t)/t$ it follows from the renewal reward theorem that

$$Y = \lambda X \implies \lim_{t\to\infty} \frac{A(n,t)}{t} = \lambda X = \lambda\pi(n).$$

Thus, (2.7.3) follows from the renewal reward theorem.

s.2.7.7. It follows from 2.7.8 to 2.7.14 that inter-departures times have the same density, i.e., $\lambda e^{-\lambda t}$. It can also be shown that the inter-departure times are independent.

Thus, the inter-departures times form a set of i.i.d. exponentially distributed random variables with mean $1/\lambda$. Consequently, the departures times form a Poisson process with rate λ.

s.2.7.8. Jobs arrive at rate λ. For a stable queue, $\mu > \lambda$. Moreover, jobs can never leave faster than they arrive.

s.2.7.9. Because jobs do not leave at rate μ.

s.2.7.10. Observe that ρ is the fraction of time the server is busy. Then, from PASTA, the fraction of jobs that see a busy server is also ρ. This fraction of jobs is $\sum_{n=1}^{\infty}\pi(n)$. Finally, $\delta(n) = \pi(n)$, a fraction ρ of the departures leaves a busy system behind.

s.2.7.11. With the hint, we first have to wait for an inter-arrival time X_n. Then, since job n's service starts right away, it leaves when $D_n = D_{n-1} + X_n + S_n$. Now observe that, due to the memoryless property of the inter-arrival times, $\mathsf{E}[X_n] = \mathsf{E}[A_n - D_{n-1}] = 1/\lambda$. Thus, the expected duration is $\mathsf{E}[X_n + S_n] = 1/\lambda + 1/\mu$.

s.2.7.12. By the previous point, the density of $D_n - D_{n-1}$ is the same as the density of $X_n + S_n$. Since $\{X_n\}$ and $\{S_n\}$ are both i.i.d. sequences, the problem becomes to find the density of $X + S$. We will use two ways of computing this.

Since $X \sim \text{Exp}(\lambda)$ and $S \sim \text{Exp}(\mu)$, and X and S are independent, their joint density is $f_{X,S}(x,y) = \lambda\mu e^{-\lambda x - \mu y}$. With this,

$$\begin{aligned}
\mathsf{P}(X+S\le t) &= \lambda\mu\int_0^\infty\int_0^\infty e^{-\lambda x-\mu y}\,\mathbb{1}_{x+y\le t}\,\mathrm{d}x\,\mathrm{d}y\\
&= \lambda\mu\int_0^t\int_0^{t-x} e^{-\lambda x-\mu y}\,\mathrm{d}y\,\mathrm{d}x\\
&= \lambda\mu\int_0^t e^{-\lambda x}\int_0^{t-x} e^{-\mu y}\,\mathrm{d}y\,\mathrm{d}x\\
&= \lambda\int_0^t e^{-\lambda x}(1-e^{-\mu(t-x)})\,\mathrm{d}x\\
&= \lambda\int_0^t e^{-\lambda x}\,\mathrm{d}x - \lambda e^{-\mu t}\int_0^t e^{(\mu-\lambda)x}\,\mathrm{d}x\\
&= 1-e^{-\lambda t} - \frac{\lambda}{\mu-\lambda}e^{-\mu t}(e^{(\mu-\lambda)t}-1)\\
&= 1-e^{-\lambda t} - \frac{\lambda}{\mu-\lambda}e^{-\lambda t} + \frac{\lambda}{\mu-\lambda}e^{-\mu t}\\
&= 1-\frac{\mu}{\mu-\lambda}e^{-\lambda t} + \frac{\lambda}{\mu-\lambda}e^{-\mu t}.
\end{aligned}$$

The density $f_{X+S}(t)$ is the derivative of this expression with respect to t, hence,

$$\begin{aligned} f_{X+S}(t) &= \frac{\lambda\mu}{\mu-\lambda}e^{-\lambda t} - \frac{\mu\lambda}{\mu-\lambda}e^{-\mu t} \\ &= \frac{\lambda\mu}{\lambda-\mu}(e^{-\mu t} - e^{-\lambda t}). \end{aligned}$$

Conditioning is much faster, but requires the concept of conditional density. You can skip the rest if you are not interested.

$$\begin{aligned} f_{X+S}(t) &= \mathsf{P}(X+S \in \mathrm{d}t) \\ &= \int \mathsf{P}(S+x \in \mathrm{d}t)\mathsf{P}(X \in \mathrm{d}x) \\ &= \int_0^t f_S(t-x)f_X(x)\mathrm{d}x \\ &= \int_0^t \mu e^{-\mu(t-x)}\lambda e^{-\lambda x}\,\mathrm{d}x \\ &= \lambda\mu e^{-\mu t}\int_0^t e^{x(\mu-\lambda)}\,\mathrm{d}x \\ &= \frac{\lambda\mu}{\lambda-\mu}\left(e^{-\mu t} - e^{-\lambda t}\right). \end{aligned}$$

s.2.7.13. After the departure, the server can start right away with the job at the head of the queue. The inter-departure time of this job is $\mathrm{Exp}(\mu)$.

s.2.7.14.

$$\begin{aligned} f_D(t) &= (1-\rho)f_{X+S}(t) + \rho\mu e^{-\mu t} \\ &= (1-\rho)\frac{\mu\lambda}{\lambda-\mu}\left(e^{-\mu t} - e^{-\lambda t}\right) + \rho\mu e^{-\mu t} \\ &= \left(1-\frac{\lambda}{\mu}\right)\frac{\mu\lambda}{\lambda-\mu}\left(e^{-\mu t} - e^{-\lambda t}\right) + \rho\mu e^{-\mu t} \\ &= \frac{\mu-\lambda}{\mu}\frac{\mu\lambda}{\lambda-\mu}\left(e^{-\mu t} - e^{-\lambda t}\right) + \frac{\lambda}{\mu}\mu e^{-\mu t} \\ &= -\lambda\left(e^{-\mu t} - e^{-\lambda t}\right) + \lambda e^{-\mu t} \\ &= \lambda e^{-\lambda t}. \end{aligned}$$

2.8 LITTLE'S LAW

Theory and Exercises

There is an important relation between the average time $\mathsf{E}[W]$ a job spends in the system and the long-run time-average number $\mathsf{E}[L]$ of jobs that is contained in the system, which is called *Little's law*:

$$\mathsf{E}[L] = \lambda\,\mathsf{E}[W]. \tag{2.8.1}$$

2.8.2 provides a proof of this under some simple conditions. In the forthcoming sections, we will apply Little's law often. Part of the usefulness of Little's law is that it applies to all input-output systems, whether it is a queueing system or an inventory system or some much more general system.

We start by defining a few intuitively useful concepts. From (1.5.10), we see that

$$\frac{1}{t}\int_0^t L(s)\,\mathrm{d}s = \frac{1}{t}\int_0^t (A(s)-D(s))\,\mathrm{d}s$$

is the time-average of the number of jobs in the system during $[0,t]$. Next, the waiting time of the kth job is the time between the moment the job arrives and departs, that is,

$$W_k = \int_0^\infty \mathbb{1}_{A_k\le s<D_k}\,\mathrm{d}s.$$

Fig. 4 relates W_k to $L(t)$.

Consider a departure time T at which the system is empty so that $A(T)=D(T)$. Then, for $k\le A(T)$,

$$W_k = \int_0^T \mathbb{1}_{A_k\le s<D_k}\,\mathrm{d}s,$$

and for $s\le T$,

$$L(s) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k\le s<D_k} = \sum_{k=1}^{A(T)} \mathbb{1}_{A_k\le s<D_k}.$$

2.8.1. *Show that*

$$\int_0^T L(s)\,ds = \sum_{k=1}^{A(T)} W_k.$$

2.8.2. *Prove Little's law under the assumptions that $A(T_i)=D(T_i)$ for an infinite number of times $\{T_i\}$ such $T_i\to\infty$ and that all limits exist.*

2.8.3. *Which assumptions have we used to prove Little's law?*

2.8.4. *Observe that the area between the graphs of $A(s)$ and $D(s)$ must be equal to the total waiting time spent by all jobs in the system until T. Use this to provide a graphical interpretation of the proof of Little's law.*

2.8.5. *Use the (physical) dimensions of the components of Little's law to check that $\mathsf{E}[W]\neq\lambda\mathsf{E}[L]$. (With this check, you can prevent making an often-made mistake.)*

2.8.6. *Consider the server of the $G/G/1$ queue as a system by itself. The time jobs stay in this system is $\mathsf{E}[S]$, and jobs arrive at rate λ. Use Little's law to conclude that $\lambda\mathsf{E}[S]=\rho:=\lim_{t\to\infty}t^{-1}\int_0^t L_S(s)\,ds$.*

2.8.7. *For a given single-server queueing system the average number of customers in the system is $\mathsf{E}[L]=10$, customers arrive at rate $\lambda=5$ per hour and are served at rate $\mu=6$ per hour. What is the average time customers spend in the system?*

2.8.8. *For a given single-server queueing system the average number of customers in the system is $\mathsf{E}[L]=10$, customers arrive at rate $\lambda=5$ per hour and are served at rate $\mu=6$ per hour. Suppose that at the moment you join the system, the number of customers in the system is 10. What is your expected time in the system?*

With the PASTA property and Little's law it becomes quite easy to derive simple expressions for the average queue length and waiting times for the $M/M/1$ queue. The average waiting time $\mathsf{E}[W]$ in the entire system is the expected time in queue plus the expected time in service, i.e.,

$$\mathsf{E}[W] = \mathsf{E}\left[W_Q\right] + \mathsf{E}[S]. \tag{2.8.2}$$

By the PASTA property we have for the $M/M/1$ queue that

$$\mathsf{E}\left[W_Q\right] = \mathsf{E}[L]\,\mathsf{E}[S]. \tag{2.8.3}$$

2.8.9. *Use Little's law to show for the $M/M/1$ queue that*

$$\mathsf{E}[W] = \frac{\mathsf{E}[S]}{1-\rho}, \qquad \mathsf{E}[L] = \frac{\rho}{1-\rho},$$
$$\mathsf{E}\left[L_Q\right] = \frac{\rho^2}{1-\rho}, \qquad \mathsf{E}[L_s] = \rho.$$

2.8.10. *Why is* (2.8.3) not *true in general for the $M/G/1$ queue?*

The following problems show how combining PASTA with Little's law allows the analysis of some non-trivial practical queueing situations.

2.8.11. *[Hall 5.2] After observing a single-server queue for several days, the following steady-state probabilities have been determined: $p(0) = 0.4$, $p(1) = 0.3$, $p(2) = 0.2$, $p(3) = 0.05$ and $p(4) = 0.05$. The arrival rate is 10 customers per hour.*

1. *Determine $\mathsf{E}[L]$ and $\mathsf{E}\left[L_Q\right]$.*
2. *Using Little's formula, determine $\mathsf{E}[W]$ and $\mathsf{E}\left[W_Q\right]$.*
3. *Determine $\mathsf{V}[L]$ and $\mathsf{V}\left[L_Q\right]$.*
4. *Determine the service time and the utilization.*

2.8.12. *(Hall 5.5) An $M/M/1$ queue has an arrival rate of 100 per hour and a service rate of 140 per hour. What is $p(n)$? What are $\mathsf{E}\left[L_Q\right]$ and $\mathsf{E}[L]$?*

2.8.13. *[Hall 5.6] An $M/M/1$ queue has been found to have an average waiting time in queue of 1 minute. The arrival rate is known to be 5 customers per minute. What are the service rate and utilization? Calculate $\mathsf{E}\left[L_Q\right]$, $\mathsf{E}[L]$ and $\mathsf{E}[W]$. Finally, the queue operator would like to provide chairs for waiting customers. He would like to have a sufficient number so that all customers can sit down at least 90 percent of the time. How many chairs should he provide?*

2.8.14. *(Hall 5.7) A single-server queueing system is known to have Poisson arrivals and exponential service times. However, the arrival rate and service time are state dependent. As the queue becomes longer, servers work faster, and the arrival rate declines, yielding the following functions (all in units of number per hour): $\lambda(0) = 5$, $\lambda(1) = 3$, $\lambda(2) = 2$, $\lambda(n) = 0, n \geq 3$, $\mu(0) = 0$, $\mu(1) = 2$, $\mu(2) = 3$, $\mu(n) = 4, n \geq 3$. Calculate the state probabilities, i.e., $p(n)$ for $n = 0, \ldots$*

2.8.15. *(Hall 5.14) An airline phone reservation line has one server and a buffer for two customers. The arrival rate is 6 customers per hour, and a service rate of just 5 customers per hour. Arrivals are Poisson and service times are exponential. Estimate $\mathsf{E}\left[L_Q\right]$ and the average number of customers served per hour. Then, estimate $\mathsf{E}\left[L_Q\right]$ for a buffer of size 5. What is the impact of the increased buffer size on the number of customers served per hour?*

2.8.16. *[Hall 5.3] After observing a queue with two servers for several days, the following steady-state probabilities have been determined:* $p(0) = 0.4$, $p(1) = 0.3$, $p(2) = 0.2$, $p(3) = 0.05$ *and* $p(4) = 0.05$. *The arrival rate is 10 customers per hour.*

1. *Determine* $\mathsf{E}[L]$ *and* $\mathsf{E}\left[L_Q\right]$.

2. *Using Little's formula, determine* $\mathsf{E}[W]$ *and* $\mathsf{E}\left[W_Q\right]$.

3. *Determine* $\mathsf{V}[L]$ *and* $\mathsf{V}\left[L_Q\right]$.

4. *Determine the service time and the utilization.*

2.8.17. *[Hall 5.8] The queueing system at a fast-food stand behaves in a peculiar fashion. When there is no one in the queue, people are reluctant to use the stand, fearing that the food is unsavory. People are also reluctant to use the stand when the queue is long. This yields the following arrival rates (in numbers per hour):* $\lambda(0) = 10$, $\lambda(1) = 15$, $\lambda(2) = 15$, $\lambda(3) = 10$, $\lambda(4) = 5$, $\lambda(n) = 0, n \geq 5$. *The stand has two servers, each of which can operate at 5 per hour. Service times are exponential, and the arrival process is Poisson. Calculate the steady state probabilities. Next, what is the average arrival rate? Finally, determine* $\mathsf{E}[L]$, $\mathsf{E}\left[L_Q\right]$, $\mathsf{E}[W]$ *and* $\mathsf{E}\left[W_Q\right]$.

2.8.18. *[Hall 5.10] A repair/maintenance facility would like to determine how many employees should be working in its tool crib. The service time is exponential, with mean 4 minutes, and customers arrive by a Poisson process with rate 28 per hour. The customers are actually maintenance workers at the facility, and are compensated at the same rate as the tool crib employees. What is* $\mathsf{E}[W]$ *for* $c = 1, 2, 3,$ *or* 4 *servers? How many employees should work in the tool crib?*

2.8.19. *[Hall 5.22] At a large hotel, taxi cabs arrive at a rate of 15 per hour, and parties of riders arrive at the rate of 12 per hour. Whenever taxicabs are waiting, riders are served immediately upon arrival. Whenever riders are waiting, taxicabs are loaded immediately upon arrival. A maximum of three cabs can wait at a time (other cabs must go elsewhere).*

1. *Let* p_{ij} *be the steady-state probability of there being* i *parties of riders and* j *taxicabs waiting at the hotel. Write the state transition equation for the system.*

2. *Calculate the expected number of cabs waiting and the expected number of parties waiting.*

3. *Calculate the expected waiting time for cabs and the expected waiting time for parties. (For cabs, compute the average among those that do not go elsewhere.)*

4. *In words, what would be the impact of allowing four cabs to wait at a time?*

2.8.20. *[Continuation of 2.8.19] Suppose cabs are not allowed to wait. What is the expected waiting time for a party of riders?*

2.8.21. *[Continuation of 2.8.19] Did you have to use the PASTA property to solve 2.8.19? If so, how did you use it? If not, why not?*

2.8.22. *[Continuation of 2.8.19] Suppose cabs can contain at most 4 riders, and the size of a party (i.e., a batch) has distribution B_k with $\mathsf{P}(B_k = i) = 1/7$ for $i = 1,\ldots,7$. Parties of riders have the same destination, so riders of different parties cannot be served by one taxi. Provide a set of recursions to simulate this system. (This is a real hard exercise, but doable. I asked it at an exam to see who would deserve the highest grade. I was lenient with the grading...)*

In the above proof of Little's law we assumed that there is a sequence of moments $\{T_k, k = 0,1,\ldots\}$ at which the system is empty and such that $T_k < T_{k+1}$ and $T_k \to \infty$. However, in many practical queueing situations the system is never empty. Thus, to be able to apply Little's law to such more general situations we should slacken the assumption that such a sequence exists. The aim of this set of questions is to find an educated guess for a more general assumption under which Little's law can hold.

2.8.23. *Motivate in words (or with a derivation, if you prefer this) why the following is true:*

$$\sum_{k=1}^{A(t)} W_k \geq \int_0^t L(s)\,ds \geq \sum_{k=1}^{D(t)} W_k.$$

2.8.24. *Take suitable limits (and assume all these limits exist) to show that*

$$\lambda\,\mathsf{E}[W] \geq \mathsf{E}[L] \geq \delta\,\mathsf{E}[W].$$

Make explicit all the points where you use the strong law of large numbers.

2.8.25. *Suppose that $A(t) = \lambda t$ and $D(t) = [A(t) - 10]^+$. Explain that for this system the above assumption on $\{T_k\}$ is violated. Show that Little's law is still true.*

2.8.26. *Based on the above formulate an educated guess for more general conditions under which Little's law holds. (You don't have to prove Little's law under your condition; postpone that to after the exam.)*

Hints

h.2.8.1. Substitute the definition of $L(s)$ in the left-hand side, then reverse the integral and summation.

h.2.8.2. In the result of 2.8.1, divide both sides by T. At the right-hand side use that $1/T = A(T)/T \cdot 1/A(T)$. Take limits.

h.2.8.4. Make a drawing of $A(t)$ and $D(t)$ until time T, i.e., the first time the system is empty. Observe that $A(t) - D(t)$ is the number of jobs in the system. Take some level k, and compute $A_k = A^{-1}(k)$ and $D_k = D^{-1}(k)$. Observe that $D_k - A_k = D^{-1}(k) - A^{-1}(k)$ is the waiting time of job k.

h.2.8.5. Checking the dimensions in the formula prevents painful mistakes.

h.2.8.7. Start with checking the units when applying Little's law.

h.2.8.9. Combine (2.8.2) and (2.8.3) and apply Little's law.

h.2.8.13. $\mathsf{E}\left[L_Q\right]$ follows right away from an application of Little's law. For the other quantities we need to find $\mathsf{E}[S]$. One can use that

$$\mathsf{E}\left[W_Q\right] = \mathsf{E}[L]\,\mathsf{E}[S] = (\mathsf{E}\left[L_Q\right] + \mathsf{E}[L_S])\mathsf{E}[S] = (\mathsf{E}\left[L_Q\right] + \lambda\,\mathsf{E}[S])\mathsf{E}[S].$$

Now λ and $\mathsf{E}\left[W_Q\right] = 1$ are given, and $\mathsf{E}\left[L_Q\right]$ has just been computed. Hence, $\mathsf{E}[S]$ (which is the unknown here) can computed with the quadratic formula.

Another way is to realize that, for the $M/M/1$-queue, $\mathsf{E}\left[W_Q\right] = \frac{1}{\lambda}\frac{\rho^2}{1-\rho}$. Then solve for ρ, and since λ is known, $\mathsf{E}[S]$ follows.

h.2.8.14. Use the level-crossing equations of the $M(n)/M(n)/1$ queue.

h.2.8.15. This is a queueing system with loss, in particular the $M/M/1/1+2$ queue.

h.2.8.18. Realize that we have to control the number of servers. Hence, we are dealing with a multi-server queue, i.e., the $M/M/c$ queue. Use 2.6.3.

The remark that maintenance workers are compensated at the same rate as the tool crib workers confused me a bit at first. Some thought revealed that the consequence of this remark is that is it just as expensive to let the tool crib workers wait (to help maintenance workers) as to let the maintenance workers wait for tools. (Recall, in queueing systems always somebody has to wait, either the customer in queue or the server being idle. If it is very expensive to let customers wait, the number of servers must be high, whereas if servers are relatively expensive, customers have to do the waiting.)

h.2.8.22. Realize that you have model the server process separately from the queueing process.

Solutions

s.2.8.1.

$$\begin{aligned}\int_0^T L(s)\,\mathrm{d}s &= \int_0^T \sum_{k=1}^{A(T)} 1\{A_k \le s < D_k\}\,\mathrm{d}s \\ &= \sum_{k=1}^{A(T)} \int_0^T 1\{A_k \le s < D_k\}\,\mathrm{d}s = \sum_{k=1}^{A(T)} W_k.\end{aligned}$$

s.2.8.2. First solve 2.8.1. Then,

$$\frac{1}{T}\int_0^T L(s)\,\mathrm{d}s = \frac{A(T)}{T}\frac{1}{A(T)}\sum_{k=1}^{A(T)} W_k.$$

Assuming there are an infinite number of times $0 \le T_i < T_{i+1} < \cdots$, $T_i \to \infty$, at which $A(T_i) = D(T_i)$ and the following limits exist

$$\frac{1}{T}\int_0^T L(s)\,\mathrm{d}s \to \mathsf{E}[L], \qquad \frac{A(T_i)}{T_i} \to \lambda, \qquad \frac{1}{A(T_i)}\sum_{k=1}^{A(T_i)} W_k \to \mathsf{E}[W],$$

we obtain Little's law.

s.2.8.3. We assumed first that $A(t)/t \to \lambda$ as $t \to \infty$, i.e., $A(t)/t$ has a limit as t converges to ∞. Second, there exists a sequence of points $T_k, k = 0,1,2,\ldots$ in time such that the server is idle. Third, either of the limits $\sum_k^n W_k/n = \sum_k^n S_k/n$ or $t^{-1}\int_0^t L(s)\mathrm{d}s$ exists, in which case the other exists.

s.2.8.4. The area enclosed between the graphs of $A(t)$ and $D(t)$ until T can be 'chopped up' in two ways: in the horizontal and in the vertical direction. (Please make the drawing as you go along...) A horizontal line between $A(t)$ and $D(t)$ corresponds to the waiting time of a job, while a vertical line corresponds to the number of jobs in the system at time t. Now adding all horizontal lines (by integrating along the y-axis) makes up the total amount of waiting done by all the jobs until time T. On the other hand, adding the vertical lines (by integrating along the x-axis) is equal to the summation of all jobs in the system. Since the area is the same no matter whether you sum it in the horizontal or vertical direction:

$$\sum_{k=1}^{A(T)} W_k = \text{enclosed area} = \int_0^T (A(t) - D(t))\,dt.$$

Dividing both sides by $A(T)$ gives

$$\frac{1}{A(T)} \sum_{k=1}^{A(T)} W_k = \frac{1}{A(T)} \int_0^T (A(t) - D(t))\,dt.$$

Finally, observe that this equality holds between any two times T_i, T_{i+1}, where times $\{T_i\}$ are such that $A(T_i) = D(T_i)$. Then, as $T_i \to \infty$, which we assumed from the on-set, $\frac{1}{A(T_i)} \sum_{k=1}^{A(T_i)} W_k \to \mathsf{E}[W]$, and

$$\frac{T_i}{A(T_i)} \frac{1}{T_i} \int_0^{T_i} (A(t) - D(t))\,dt \to \lambda^{-1}\,\mathsf{E}[L].$$

Hence, Little's law follows.

s.2.8.5. Sometimes (often?) students memorize Little's law in the wrong way. Thus, as an easy check, use the dimensions of the concepts: $\mathsf{E}[L]$ is an average *number*, λ is a *rate*, i.e., *numbers per unit time*, and $\mathsf{E}[W]$ is waiting *time*.

s.2.8.6. The arrival rate at the server must be λ and the time a job remains at the server is $\mathsf{E}[S]$. The fraction of time the server is busy is precisely the fraction of time there is a job present at the server. Thus, applying Little's law to the server itself, we see that $\rho = \mathsf{E}[L_S] = \lambda\,\mathsf{E}[S]$.

s.2.8.7.

$$\mathsf{E}[W] = \mathsf{E}[L]/\lambda = 10/\lambda = 10/5 = 2.$$

s.2.8.8. If you arrive at a queueing system, you first have to wait until the job in service is finished. Then you need to wait until the 9 jobs in queue are finished. This takes, in expectation, $9/\mu$. (Recall, 1 job is in service at the moment you arrive, so 9 are in queue.) Assuming that service times are exponential, so that, by the memoryless property, the remaining service time of the job in service is still $\mathsf{E}[S]$ when you arrive, you spend $10/\mu + 1/\mu = 11/6 \neq 2$. (To account for the last $+1/\mu$, observe that yourself also have to be served to compute the time you spend in the system.)

Now in this question, it is *given* that the system length is 10 at the moment of arrival. However, L as 'seen' upon arrival by this given customer is in general not the same as the time-average $\mathsf{E}[L]$.

Thus, Little's law need not hold at all moments in time; it is a statement about *averages*.

s.2.8.9.

$$\begin{aligned}
\mathsf{E}[W] &= \mathsf{E}[L]\,\mathsf{E}[S] + \mathsf{E}[S] = \lambda\,\mathsf{E}[W]\,\mathsf{E}[S] + \mathsf{E}[S] = \rho\,\mathsf{E}[W] + \mathsf{E}[S],\\
\mathsf{E}[L] &= \lambda\,\mathsf{E}[W] = \frac{\lambda\,\mathsf{E}[S]}{1-\rho} = \frac{\rho}{1-\rho},\\
\mathsf{E}\left[W_q\right] &= \mathsf{E}[W] - \mathsf{E}[S] = \frac{\mathsf{E}[S]}{1-\rho} - \mathsf{E}[S] = \frac{\rho}{1-\rho}\,\mathsf{E}[S],\\
\mathsf{E}\left[L_Q\right] &= \lambda\,\mathsf{E}\left[W_Q\right] = \frac{\rho^2}{1-\rho},\\
\mathsf{E}[L_s] &= \mathsf{E}[L] - \mathsf{E}\left[L_Q\right] = \frac{\rho}{1-\rho} - \frac{\rho^2}{1-\rho} = \rho,
\end{aligned}$$

s.2.8.10. By the memoryless property of the (exponential) distributed service times of the $M/M/1$ queue, the duration of a job in service, if any, is $\mathrm{Exp}(\mu)$ also at an arrival moment. Therefore, at an arrival moment, all jobs in the system (whether in service or not) have the same expected duration. Hence, the expected time to spend in queue is the expected number of jobs in the system times the expected service time of each job, i.e., $\mathsf{E}\left[W_q\right] = \mathsf{E}[L]\,\mathsf{E}[S]$. Note that we use PASTA to see that the expected number of jobs in the system at an arrival is $\mathsf{E}[L]$. For the $M/G/1$ queue, the job in service (if any) does not have the same distribution as a job in queue. Hence, the expected time in queue is not $\mathsf{E}[L]\,\mathsf{E}[S]$.

s.2.8.11. First find $\mathsf{E}[L]$

```
>>> P = [0.4, 0.3, 0.2, 0.05, 0.05]
>>> EL = sum(n*P[n] for n in range(len(P)))
>>> EL
1.05
```

There can only be a queue when a job is in service. Since there is $m = 1$ server, we subtract m from the amount of jobs in the system. Before we do this, we need to ensure that $n - m$ does not become negative. Thus, $\mathsf{E}\left[L_Q\right] = \sum_n \max\{n-m, 0\}p(n)$.

```
>>> m = 1
>>> ELq = sum(max(n-m,0)*P[n] for n in range(len(P)))
>>> ELq
0.45000000000000007

>>> labda = 10./60
>>> Wq = ELq/labda # in minutes
>>> Wq
2.7000000000000006
>>> Wq/60 # in hours
0.04500000000000001

>>> W = EL/labda # in minutes
>>> W
6.300000000000001
>>> W/60 # in hours
0.10500000000000001
```

Let's use the standard definition of the variance, i.e., $\mathsf{V}[X] = \sum_i (x_i - \mathsf{E}[X])^2 \mathsf{P}(X = x_i)$, for once.

```
>>> from math import sqrt
>>> var_L = sum((n-EL)**2*P[n] for n in range(len(P)))
>>> var_L
1.2475
>>> sqrt(var_L)
1.116915395184434

>>> var_Lq = sum((max(n-m,0)-ELq)**2*P[n] for n in range(len(P)))
>>> var_Lq
0.6475
>>> sqrt(var_Lq)
0.8046738469715541

>>> mu = 1./(W-Wq)
>>> 1./mu # in minutes
3.5999999999999996

>>> rho = labda/mu
>>> rho
0.6

>>> rho = EL-ELq
>>> rho
0.6
```

This checks with the previous line.

The utilization must also by equal to the fraction of time the server is busy.

```
>>> u = 1 - P[0]
>>> u
0.6
```

Yet another way: Suppose we have m servers. If the system is empty, all m servers are idle. If the system contains one customer, $m-1$ servers are idle. Therefore, in general, the average fraction of time the server is idle is

$$1 - u = \sum_{n=0}^{\infty} \max\{n - m, 0\} p_n,$$

as in the case there are more than m customers in the system, the number of idle servers is 0.

```
>>> idle = sum( max(m-n,0)*P[n] for n in range(len(P)))
>>> idle
0.4
```

s.2.8.12. First, $p(n) = (1-\rho)\rho^n$. Now,

```
>>> labda = 100. # per hour
>>> mu = 140. # per hour
>>> ES = 1./mu
>>> rho = labda/mu
>>> rho
0.7142857142857143
>>> 1-rho
0.2857142857142857

>>> L = rho/(1.-rho)
>>> L
2.5
>>> Lq = rho**2/(1.-rho)
>>> Lq
1.7857142857142858

>>> W = 1./(1.-rho) * ES
>>> W
0.024999999999999998
>>> Wq = rho/(1.-rho) * ES
>>> Wq
0.017857142857142856
```

s.2.8.13. $\mathsf{E}[W_Q] = \frac{1}{\lambda}\frac{\rho^2}{1-\rho}$. Since $\mathsf{E}[W_Q]$ and λ is given we can use this formula to solve for ρ with the quadratic formula (and using that $\rho > 0$):

```
>>> labda = 5. # per minute
>>> Wq = 1.
>>> a = 1.
>>> b = labda*Wq
>>> c = -labda*Wq
>>> rho = (-b + sqrt(b*b-4*a*c))/(2*a)
>>> rho
0.8541019662496847

>>> ES = rho/labda
>>> ES
0.17082039324993695

>>> Lq = labda*Wq
>>> Lq
5.0

>>> W = Wq + ES
>>> W
1.170820393249937

>>> L = labda*W
```

```
>>> L
5.854101966249685
```

The next problem is to find n such that $\sum_{j=0}^{n} p_j > 0.9$.

```
>>> total = 0.
>>> j = 0
>>> while total <= 0.9:
...  total += (1-rho)*rho**j
...  j += 1
...
>>> total
0.9061042738330157
>>> j
15
>>> n = j- 1 # the number of chairs
```

Observe that j is one too high once the condition is satisfied, thus subtract one.
As a check, I use that $(1-\rho)\sum_{j=0}^{n} \rho^j = 1-\rho^{n+1}$.

```
>>> 1-rho**(n) # this must be too small.
0.890064968964683
>>> 1-rho**(n+1) # this must be OK.
0.9061042738330156
```

And indeed, we found the right n.

s.2.8.14. Follows right away from the hint.

s.2.8.15. First compute $\mathsf{E}[L_Q]$ for the case with a buffer for 2 customers.

```
>>> labda = 6.
>>> mu = 5.
>>> rho = labda/mu
>>> c = 1
>>> b = 2
```

Set $p(n) = \rho^n$ initially, and normalize later. Use the expressions for the $M(n)/M(n)/1$ queue. Observe that $\rho > 1$. Since the size of the system is $c+b+1$ is finite, all formulas work for this case too.

There are 4 states in total: $0, 1, 2, 3$. (The reason to import `numpy` here and convert the lists to arrays is to fix the output precision to 3, otherwise we get long floats in the output.)

```
>>> import numpy as np
>>> np.set_printoptions(precision=3)

>>> P = np.array([rho**n for n in range(c+b+1)])
>>> P
array([1.   , 1.2  , 1.44 , 1.728])
```

```
>>> G = sum(P)
>>> G
5.368

>>> P /= G # normalize
>>> P
array([0.186, 0.224, 0.268, 0.322])

>>> L = sum(n*P[n] for n in range(len(P)))
>>> L
1.725782414307004

>>> Lq = sum((n-c)*P[n] for n in range(c,len(P)))
>>> Lq
0.9120715350223545
```

The number of jobs served per hour must be equal to the number of jobs accepted, i.e., not lost. The fraction of customers lost is equal to the fraction of customers that sees a full system.

```
>>> lost = labda*P[-1] # the last element of P
>>> lost
1.9314456035767507

>>> accepted = labda*(1.-P[-1]) # rate at which jobs are accepted
>>> accepted
4.06855439642325
```

Now increase the buffer b to 5.

```
>>> b = 5
>>> P = np.array([rho**n for n in range(c+b+1)])
>>> P
array([1.   , 1.2  , 1.44 , 1.728, 2.074, 2.488, 2.986])
>>> G = sum(P)
>>> G
12.915903999999998

>>> P /= G # normalize
>>> P
array([0.077, 0.093, 0.111, 0.134, 0.161, 0.193, 0.231])

>>> L = sum(n*P[n] for n in range(len(P)))
>>> L
3.7098374221424995

>>> accepted = labda*(1.-P[-1])
>>> accepted
4.6128803682653565
```

s.2.8.16. Determine $\mathsf{E}[L]$ and $\mathsf{E}\left[L_Q\right]$.

```
>>> P = [0.4, 0.3, 0.2, 0.05, 0.05]

>>> c = 2
>>> Lq = sum((n-c)*P[n] for n in range(c,len(P)))
>>> Lq
0.15000000000000002

>>> L= sum(n*P[n] for n in range(len(P)))
>>> L
1.05
```

Using Little's formula, determine $\mathsf{E}[W]$ and $\mathsf{E}\left[W_Q\right]$.

```
>>> labda = 10./60
>>> Wq = Lq/labda # in minutes
>>> Wq
0.9000000000000001
>>> Wq/60 # in hours
0.015000000000000003

>>> W = L/labda
>>> W
6.300000000000001
```

Determine $\mathsf{V}[L]$ and $\mathsf{V}\left[L_Q\right]$.

```
>>> from math import sqrt
>>> var_L = sum((n-L)**2*P[n] for n in range(len(P)))
>>> var_L
1.2475
>>> sqrt(var_L)
1.116915395184434

>>> var_Lq = sum((max(n-c,0)-Lq)**2*P[n] for n in range(len(P)))
>>> var_Lq
0.22750000000000004
```

Determine the service time and the utilization.

```
>>> mu = 1./(W-Wq)
>>> 1./mu # in minutes
5.4

>>> rho = labda/mu
>>> rho
0.9
```

```
>>> rho = L-Lq
>>> rho
0.9
```

This checks the previous line.

The utilization must also by equal to the fraction of time the server is busy.

```
>>> u = 1 - P[0]
>>> u
0.6
```

s.2.8.17. First the service rates.

```
>>> import numpy as np
>>> from math import factorial
>>> labda = [10., 15., 15., 10., 5.]
>>> c = 2
>>> mn = 2*np.ones(len(labda)+1, dtype=int) # number of active servers
>>> mn[0] = 0 # no service if system is empty
>>> mn[1] = 1 # one busy server if just one job present
>>> mu = 5*mn # service rate is 5 times no of active servers
>>> mu
array([ 0,  5, 10, 10, 10, 10])
```

Since there can be arrivals in states $0,\ldots,4$, the system can contain 0 to 5 customers, i.e., $p(0),\ldots,p(5)$.

Use the level-crossing result for the $M(n)/M(n)/1$ queue:

```
>>> P = [1]*(len(labda)+1)
>>> for i in range(1,len(P)):
...  P[i] = labda[i-1]/mu[i]*P[i-1]
...
>>> P = np.array(P) # unnormalized probabilities
>>> P
array([1.  , 2.  , 3.  , 4.5 , 4.5 , 2.25])

>>> G = sum(P) # normalization constant
>>> G
17.25
>>> P /= G # normalize
>>> P
array([0.058, 0.116, 0.174, 0.261, 0.261, 0.13 ])
```

$\lambda = \sum_n \lambda(n)p(n)$.

```
>>> labdaBar = sum(labda[n]*P[n] for n in range(len(labda)))
>>> labdaBar
8.840579710144928
```

The average number in the system is:

```
>>> Ls = sum(n*P[n] for n in range(len(P)))
>>> Ls
2.942028985507246
```

The average number in queue:

```
>>> c = 2
>>> Lq = sum((n-c)*P[n] for n in range(c,len(P)))
>>> Lq
1.1739130434782608
```

And now the waiting times:

```
>>> Ws = Ls/labdaBar
>>> Ws # time in the system
0.3327868852459016
```

```
>>> Wq = Lq/labdaBar
>>> Wq # time in queue
0.13278688524590163
```

s.2.8.18. Would one server/person do?

```
>>> labda = 28./60 # arrivals per minute
>>> ES = 4.
>>> labda*ES
1.8666666666666667
```

If $c = 1$, the load $\rho = \lambda \mathsf{E}[S]/c > 1$ is clearly undesirable for one server. We need at least two servers.

It is not relevant to focus on the time in the system, as time in service needs to be spent anyway. Hence, we focus on the waiting time in queue.

I just convert the formulas of 2.6.3 to Python code. This saves me time during the computations.

```
>>> def WQ(c, labda, ES):
...  from math import factorial
...  rho = labda*ES/c
...  G = sum([(c*rho)**n/factorial(n) for n in range(c)])
...  G += (c*rho)**c/(1.-rho)/factorial(c)
...  Lq = (c*rho)**c/(factorial(c)*G) * rho/(1.-rho)**2
...  return Lq/labda # Wq, Little's law
...
```

Considering the scenario with one server is superfluous as $\rho > 1$ in that case.

What is the waiting time for $c = 2$ servers?

```
>>> WQ(2, 28./60, 4) # in minutes
27.034482758620694
>>> WQ(2, 28./60, 4)/60. # in hours
0.4505747126436782
```

What is the waiting time for $c = 3$ servers?

```
>>> WQ(3, 28./60, 4) # in minutes
1.3542675591474136
>>> WQ(3, 28./60, 4)/60. # in hours
0.022571125985790228
```

What is the waiting time for $c = 4$ servers?

```
>>> WQ(4, 28./60, 4) # in minutes
0.26778942672317635
>>> WQ(4, 28./60, 4)/60. # in hours
0.0044631571120529396
```

In the next part of the question we will interpret these numbers.

Since both types of workers cost the same amount of money per unit time, it is best to divide the amount of waiting/idleness equally over both types of workers. I am inclined to reason as follows. The average amount of waiting time done by the maintenance workers per hour is $\lambda \mathsf{E}[W_Q]$. To see this, note that maintenance workers arrive at rate λ, and each worker waits on average $\mathsf{E}[W_Q]$ minutes. Thus, worker time is wasted at rate $\lambda \mathsf{E}[W_Q]$. Interestingly, with Little's law, $\mathsf{E}[L_Q] = \lambda \mathsf{E}[W_Q]$, i.e., the rate at which workers waste capacity (i.e. waiting in queue) is $\mathsf{E}[L_Q]$. On the other hand, the rate of work capacity wasted by the tool crib employees being idle is $c - \lambda \mathsf{E}[S]$, as $\lambda \mathsf{E}[S]$ is the average number of servers busy, while c crib servers are available.

As both types of employees are equally expensive, we need to choose c such that the number of maintenance workers waiting (i.e., being idle because they are waiting in queue), is equal to the number of crib workers being idle. In other words, we search for a c such that $\mathsf{E}[L_Q] \approx c - \lambda \mathsf{E}[S]$ (where, of course, $\mathsf{E}[L_Q]$ depends on c).

```
>>> labda = 28./60
>>> ES = 4.
>>> c = 2
>>> ELQ = labda*WQ(c, labda, ES)
>>> ELQ
12.616091954022991
>>> c-labda*ES
0.1333333333333333
```

Now the maintenance employees wait more than the tool crib employees.

```
>>> c = 3
>>> ELQ = labda*WQ(c, labda, ES)
>>> ELQ
0.6319915276021264
>>> c-labda*ES
1.1333333333333333

>>> c = 4
>>> ELQ = labda*WQ(c, labda, ES)
```

```
>>> ELQ
0.1249683991374823
>>> c-labda*ES
2.1333333333333333
```

Clearly, $c = 3$ should do.

s.2.8.19. Let p_{ij} be the fraction of time that the system contains i riders and j taxi cabs.

I assume that all members of a party of riders can be served by a single cab (that is, the parties do not exceed the capacity of a cab and all members of a party have the same destination).

For clarity, write μ for the rate at which cabs arrive, and λ for the arrival rate of parties of riders.

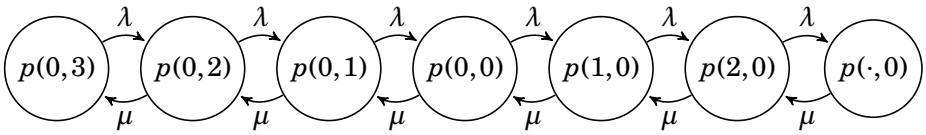
Then the transitions are as in the figure below.

Suppose first that there are 3 taxi cabs.

When a group arrives (at rate λ), there is one taxi less, and so on, until there are no more taxis left.

Finally, if yet more groups arrive, they have to wait.

When a new taxi arrives, the number of groups is reduced by one, and so on, until there are 3 taxis waiting and no groups of people.



From this figure, we see that

$$\begin{aligned}
\lambda p_{0,3} &= \mu p_{0,2} \\
(\lambda+\mu)p_{0,2} &= \mu p_{0,1} + \lambda p_{0,3} \\
(\lambda+\mu)p_{0,1} &= \mu p_{0,0} + \lambda p_{0,2} \\
(\lambda+\mu)p_{0,0} &= \mu p_{1,0} + \lambda p_{0,1} \\
(\lambda+\mu)p_{1,0} &= \mu p_{2,0} + \lambda p_{0,0} \\
(\lambda+\mu)p_{2,0} &= \mu p_{3,0} + \lambda p_{1,0}
\end{aligned}$$

and so on. Thus, it is left to compute p_{ij}. Observe from this scheme, or the above figure, that the situation with the taxis correspond to an $M/M/1$ queue, only the states have a 'different name'. Let q be the number of jobs in an M/M/1 queue. Some thought will reveal that the queueing system with cabs and parties can be mapped to an equivalent M/M/1 queueing system. In fact, consider the following table

j	i	q
3	0	0
2	0	1
1	0	2
0	0	3
0	1	4
0	2	5

and so on. Therefore, in general, it must be that

$$q = 3 - j + i.$$

From the M/M/1 queue we know right away that $p_q = \rho^q(1-\rho)$. With the above relation we can therefore immediately find that $p_{ij} = \rho^{3-j+i}(1-\rho)$, save that i and j must satisfy the constraints imposed by the model.

Second, the expected number of cabs waiting must be

$$1p_{0,1} + 2p_{0,2} + 3p_{0,3}$$

and the expected number of parties waiting must be $\sum_{j=1}^{\infty} jp_{j,0}$.

```
>>> labda = 12. # per hour
>>> mu = 15. # per hour
>>> rho = labda/mu

>>> def p(i,j):
...  q = 3 - j + i
...  return rho**q*(1.-rho)
...
```

Expected number of cabs waiting:

```
>>> Lc = sum(j*p(0,j) for j in range(0,4))
>>> # Recall this sums up to 4, not including 4
>>> Lc
1.0479999999999998
```

To compute the expected number of parties waiting we formally have to sum to infinity. Rather than doing the algebra, I chose to truncate the summation at an i such that $\rho^i \ll 1$, i.e., negligible. Truncating at 30 seems reasonable enough:

```
>>> trunc = 30
>>> rho**trunc
0.0012379400392853823
```

At second thought this is not yet really small.

```
>>> trunc = 50
>>> rho**trunc
1.4272476927059638e-05
```

This is better. Now go for what we want to know:

```
>>> Lp = sum(i*p(i,0) for i in range(trunc))
>>> Lp
2.0476053945579213
```

For the last part: This is tricky. I first, naively, computed $W_q = L_c/\mu$. This seems to make sense, as cabs arrive at rate μ, so that this expression follows from a standard application of Little's law. However, this is wrong, of course. When using Little's law to relate the number of jobs in queue (i.e., in the M/M/1 queue) and the queueing time we need to use λ, not μ. Similarly (and more formally by the mapping developed in part a), for our cab system we also need to use λ.

```
>>> Wq = Lc/labda
>>> Wq
0.08733333333333332
```

Thinking in templates is often useful, but makes one sloppy...

What would be the impact of allowing 4 cabs? Funny question, and with the above, trivial to answer.

```
>>> def p(i,j):
...  q = 4 - j + i
...  return rho**q*(1.-rho)
...

>>> Lc = sum(j*p(0,j) for j in range(0,4))
>>> Lc
0.8383999999999999

>>> Lp = sum(i*p(i,0) for i in range(trunc))
>>> Lp
1.638084315646337
```

s.2.8.20. Now we have the standard $M/M/1$ queue. The number of parties waiting must be $\mathsf{E}[L]$ of the $M/M/1$ queue. We can use Little's law to compute the waiting time.

s.2.8.21. Actually, you don't have to use PASTA. So why is that? For instance, $\sum_{i=1}^{3} i p(0,i)$ is the number of taxis waiting, and this is a time average (since we use $p(0,i)$). In the proof of Little's law, it is also clear that the $\mathsf{E}[L]$ is a time average. Also, in the proof of Little's law, we compute the waiting time as $\mathsf{E}[W] = \lim_{n\to\infty} n^{-1} \sum_{k=1}^{n} W_k$, where W_k is the waiting time as perceived by the kth job. Thus, here $\mathsf{E}[W]$ is the average as observed by jobs.

This was an old exam question. Some students used the PK-formula, see Section 2.10, to compute the average waiting time. The derivation of this *does* depend on the PASTA property.

s.2.8.22. We concentrate on departure epochs of the taxis. Thus the kth period is the time between the departure of taxi $k-1$ and taxi k. During the kth epoch a_k batches can arrive.

The system starts with a_0 batches in queue.

Suppose that the first batch contains 5 riders. Then the first taxi takes 4 riders, and 1 rider of the batch remains. This 1 rider will take the next taxi, and no riders of other groups can join because the riders of different parties have different destinations. Once all riders of a party are served, the next party in line can move to the 'server' and wait to be served.

The recursions are as follows; realize that the order in which you carry out these recursions is important. The meaning of each line is explained below the line.

$$d_s = \min\{Q_s, 4\},$$

the number of riders that can depart from the party of riders in service,

$$Q'_s = Q_s - d_s,$$

the remaining of number riders of a party after being served by one taxi,

$$Q = Q + a_k,$$

the number of parties in queue (not in the system) just after the arrival of the batches during the kth interdeparture time,

$$pd_q = \min\{Q, \mathbb{1}_{Q'_s=0}\},$$

only move a party from the queue if 'the server is free',

$$Q = Q - d_q,$$

move the party from the queue to the server, if allowed,

$$Q_s = Q'_s \mathbb{1}_{Q'_s>0} + B_b \mathbb{1}_{Q'_s=0},$$

if the server is not free, the number of riders is equal to Q'_s, otherwise send the bth batch to the server,

$$b = b + \mathbb{1}_{Q'_s=0},$$

if the server is free, move the index of the batch in service to the next batch to be served once the server becomes free again.

In the code A_k corresponds to a list of batches arriving on the kth day, B_i to the size of the ith batch, and a_k to the number of batches arriving on the kth period. I used the `pysnooper` module to debug the code. It is quite hard to get it right.

```
A = [[5, 3, 4], [3], [6], [1, 1], [], [2], [], [], [3], []]

a = [len(A[i]) for i in range(len(A))]

B = [item for sublist in A for item in sublist]

Qs = 0
b = 0
Q = 0

for k in range(len(a)):
 ds = min(Qs, 4)
 Qs_p = Qs - ds
 Q += a[k]
 dq = min(Q, 1 * (Qs_p == 0))
 Q -= dq
 Qs = (Qs_p > 0) * Qs_p + (Qs_p == 0) * B[b]
 b += Qs_p == 0
 print(f"ds={ds}, Qs_p={Qs_p}, dq={dq}, Q={Q}, Qs={Qs}, b={b}")
```

s.2.8.23. Intuitively, the left term is all the work that arrived up to time t, the middle term is all the work that has been processed, and the right term all the work that left. Any job that that is half way its service counts for full at the left, for half in the middle expression, and not in the right.

More formally, for any job k and time t, we have $W_k \mathbb{1}_{A_k \le t} \ge \int_0^t \mathbb{1}_{A_k \le s < D_k} \mathrm{d}s \ge W_k \mathbb{1}_{D_k \le t}$. (To see this, fix k, and check the three cases $t < A_k, A_k \le t < D_k, D_k < t$.) Then,

$$\sum_{k=1}^{\infty} W_k \mathbb{1}_{A_k \le t} \ge \int_0^t \sum_{k=1}^{\infty} \mathbb{1}_{A_k \le t < D_k} \mathrm{d}s \ge \sum_{k=1}^{\infty} W_k \mathbb{1}_{D_k \le t}.$$

Finally, note that $\sum_{k=1}^{\infty} W_k \mathbb{1}_{A_k \le t} = \sum_{k=1}^{A(t)} W_k$ and $\sum_{k=1}^{\infty} W_k \mathbb{1}_{D_k \le t} = \sum_{k=1}^{D(t)} W_k$, and use the definition of $L(s)$.

s.2.8.24.

$$\lim_{t\to\infty} \frac{A(t)}{t} \frac{1}{A(t)} \sum_{k=1}^{A(t)} W_k \ge \lim_{t\to\infty} \frac{1}{t} \int_0^t L(s)\,\mathrm{d}s \ge \lim_{t\to\infty} \frac{D(t)}{t} \frac{1}{D(t)} \sum_{k=1}^{D(t)} W_k.$$

We use the strong law of large numbers to conclude that limits converges to $n^{-1}\sum_{k=1}^{n} W_k \to \mathsf{E}[W]$, and we assume that $\{W_k, k \ge N\}$ forms a sequence of i.i.d. random variables for N sufficiently large.

s.2.8.25. When $t > 10$, $L(t) = 10$. Hence the system is never empty. Since still $\delta = \lim_{t\to\infty} D(t)/t = \lambda$, and $\lim_{n\to\infty} n^{-1}\sum_{k=1}^{n} W_k$ exists, Little's law follows.

s.2.8.26. We seem to need that $\lambda = \delta$ and that $\lim_{n\to\infty} n^{-1}\sum_{k=1}^{n} W_k$ exists.

2.9 $M^X/M/1$ QUEUE: EXPECTED WAITING TIME

Theory and Exercises

Sometimes jobs arrive in batches, rather than as single units. For instance, when a car or a bus arrives at a fast-food restaurant, a batch consists of the number of people in the vehicle. When the batches arrive as a Poisson process and the individual items within a batch have exponential service times we denote such queueing systems by the shorthand $M^X/M/1$. We derive expressions for the load and the expected waiting time and queue length for this queueing model.

Assume that jobs arrive as a Poisson process with rate λ and each *job* contains multiple *items*. Let A_k be the arrival time of job k and $A(t)$ the number of (job) arrivals up to time t. Denote by B_k the batch size of the kth job, i.e., the number of items of job k. We assume that $\{B_k\}$ is a sequence of independent discrete random variables each distributed as the generic random variable B. Let $\mathsf{P}(B = k) = f(k)$ be given. The service time of each item is $\mathsf{E}[S]$.

2.9.1. *Explain that the average time to serve an entire batch is* $\mathsf{E}[B]\,\mathsf{E}[S]$*, so that the load must given by* $\rho = \lambda\,\mathsf{E}[B]\,\mathsf{E}[S]$.

The aim of the remainder of the section is to derive a cornerstone of queueing theory, which is the following formula for the expected time an item spends in queue:

$$\mathsf{E}\left[W_Q\right] = \frac{1+C_s^2}{2} \frac{\rho}{1-\rho} \mathsf{E}[B]\,\mathsf{E}[S] + \frac{1}{2} \frac{\rho}{1-\rho} \mathsf{E}[S], \tag{2.9.1}$$

where C_s^2 is the SCV of the batch size distribution. By applying (2.8.3), it follows right away that the expected number of items in the system takes the form

$$\mathsf{E}[L] = \frac{\mathsf{E}\left[W_Q\right]}{\mathsf{E}[S]} = \frac{1+C_s^2}{2}\frac{\rho}{1-\rho}\mathsf{E}[B] + \frac{1}{2}\frac{\rho}{1-\rho}. \tag{2.9.2}$$

Note that $\rho < 1$ is required, as usual.

Before deriving the above, let us try to use it.

2.9.2. *What is* $\mathsf{E}[L]$ *in case* $B_k = 3$ *always, and* $\lambda = 1$, $\mu = 6$*?*

2.9.3. *If the batch size is geometrically distributed with success probability* p *, what is* $\mathsf{E}[L]$*?*

2.9.4. *A common operational problem is a machine that receives batches of various sizes. Management likes to know how a reduction of the variability of the batch sizes would affect the average queueing time. Suppose, for the sake of an example, that the batch size*

$$\mathsf{P}(B=1) = \mathsf{P}(B=2) = \mathsf{P}(B=3) = \frac{1}{3}.$$

Batches arrive at rate 1 per hour. The average processing time for an item is 25 *minutes. Compute by how much the number of items in the system would decreases if batch sizes were constant and equal to* 2*; hence the load is the same in both cases.*

2.9.5. *Show that when the batch size is 1, the expression* $\mathsf{E}\left[L(M^X/M/1)\right]$*, i.e., the system length for the* $M^X/M/1$ *queue, reduces to* $\mathsf{E}[L(M/M/1)]$*, i.e., the system length for the* $M/M/1$ *queue. Realize the importance of such checks.*

2.9.6. *Show that* $\mathsf{E}\left[W_Q(M^X/M/1)\right] \geq \mathsf{E}\left[W_Q(M/M/1)\right]$ *when the loads are the same. What do you conclude? (This solution of this exercise is more useful than you might think.)*

Let us now focus on deriving (2.9.1). Assume that an arriving batch joins the end of the queue (if present), and once the queue in front of it has been cleared, it moves in its entirety to the server. Thus, all items in one batch spend the same time in queue. Once the batch moves to the server, the server processes the items one after another until the batch is empty. Write $\mathsf{E}\left[L_Q^B\right]$ for the number of batches in queue and $\mathsf{E}\left[L_S^B\right]$ for the number of items of the job (if any) at the server. Observe first that the average time an item spends in queue is

$$\mathsf{E}\left[W_Q\right] = \mathsf{E}[L]\,\mathsf{E}[S] = \left(\mathsf{E}\left[L_Q^B\right]\mathsf{E}[B] + \mathsf{E}\left[L_S^B\right]\right)\mathsf{E}[S].$$

We also see that the average time a batch spends in queue is

$$\mathsf{E}\left[W_Q^B\right] = \mathsf{E}\left[L_Q^B\right]\mathsf{E}[B]\,\mathsf{E}[S] + \mathsf{E}\left[L_S^B\right]\mathsf{E}[S].$$

Hence, $\mathsf{E}\left[W_Q\right] = \mathsf{E}\left[W_Q^B\right]$.

2.9.7. *Use Little's law to show that*

$$\mathsf{E}\left[W_Q^B\right] = \frac{\mathsf{E}[L_S]}{1-\rho}\mathsf{E}[S].$$

Clearly, we are done if we can find an expression for $\mathsf{E}[L_S]$. For this we can use the renewal reward theorem; in fact, we can use 2.2.1 as inspiration. (Solve this exercise if you have not done yet.) Define $Y(t) = \int_0^t L_S^B(s)\mathrm{d}s$ to see that $\mathsf{E}\left[L_S^B\right] = Y = \lim_{t\to\infty} Y(t)/t$.

2.9.8. *Let* $X_k = Y(D_k) - Y(D_{k-1})$. *Then* $\mathsf{E}[X] = \lim_{n\to\infty} n^{-1}\sum_{k=1}^{n} X_k$. *With this, show that*

$$\mathsf{E}[X] = \frac{\mathsf{E}\left[B^2\right] + \mathsf{E}[B]}{2}\,\mathsf{E}[S].$$

2.9.9. *Use the renewal reward theorem with the sampling epochs* $T_k = D_k$ *to prove that*

$$\mathsf{E}\left[L_S^B\right] = \lambda\frac{\mathsf{E}\left[B^2\right]}{2}\,\mathsf{E}[S] + \frac{\rho}{2}.$$

2.9.10. *Show that*

$$\frac{\mathsf{E}\left[B^2\right]}{(\mathsf{E}[B])^2} = 1 + C_s^2$$

2.9.11. *Use 2.9.9 to conclude* (2.9.1).

Hints

h.2.9.2. Use (2.9.2). What are $\mathsf{E}\left[B^2\right]$, $\mathsf{E}[B]$ and $\mathsf{V}[B]$ for this case?

h.2.9.3. $f_k = q^{k-1}p$ with $q = 1-p$. Use generating functions to compute $\mathsf{E}[B]$ and $\mathsf{E}\left[B^2\right]$.

h.2.9.5. What is the distribution of the batch size B for the $M/M/1$ queue?

h.2.9.6. Use 2.9.5 and Jensen's inequality.

h.2.9.7. Use Little's law in $\mathsf{E}\left[W_Q^B\right] = \mathsf{E}\left[L_Q^B\right]\mathsf{E}[B]\,\mathsf{E}[S] + \mathsf{E}\left[L_S^B\right]\mathsf{E}[S]$.

h.2.9.8. Realize that $X_k = \int_{D_{k-1}}^{D_k} L_S^B(s)\mathrm{d}s$ is precisely the amount added to Y by the kth batch. Suppose now that the size of the kth batch is $B_k = 2$. Writing S_1 for the service time of the first item of this batch and S_2 for the service of the second item, we see that $X_k = 2\cdot S_1 + 1\dot{S}_2$. Now consider general batch sizes B.

h.2.9.9. Solve 2.9.8 first.

Solutions

s.2.9.1. The expected service time of a batch is $\mathsf{E}[B]\,\mathsf{E}[S]$, because $\mathsf{E}[B]$ is the expected number of items in a job, and the expected service time of each item is $\mathsf{E}[S]$. Since jobs arrive at rate λ, work arrives at rate $\lambda\,\mathsf{E}[B]\,\mathsf{E}[S]$.

s.2.9.2. As B is constant and equal to 3, $\mathsf{E}\left[B^2\right] = 9$. Hence, $\mathsf{V}[B] = 0$, which implies $C_s^2 = 0$. Also, $\rho = \lambda\,\mathsf{E}[B]/\mu = 1\cdot 3/6 = 1/2$. Hence,

$$\mathsf{E}[L] = \frac{1}{2}\frac{1/2}{1-1/2}\cdot 3 + \frac{1}{2}\frac{1/2}{1-1/2}.$$

s.2.9.3. We need $\mathsf{V}[B]$ and $\mathsf{E}[B]$. Consider

$$M_B(s) = \mathsf{E}\left[e^{sB}\right] = \sum_{k=0}^{\infty} e^{sk}\mathsf{P}(B=k)$$

$$= \sum_{k=0}^{\infty} e^{sk} p q^{k-1} = \frac{p}{q} \sum_{k=0}^{\infty} (qe^s)^k = \frac{p}{q} \frac{1}{1-qe^s},$$

$$\mathsf{E}[B] = M_B'(0) = \frac{p}{q} \frac{q}{(1-qe^s)^2}\bigg|_{s=0} = \frac{p}{(1-q)^2} = \frac{1}{p},$$

$$\mathsf{E}\left[B^2\right] = M_B''(0) = \frac{2}{p^2} - \frac{1}{p},$$

$$\mathsf{V}[B] = \mathsf{E}\left[B^2\right] - (\mathsf{E}[B])^2 = \frac{2}{p^2} - \frac{1}{p} - \frac{1}{p^2} = \frac{1}{p^2} - \frac{1}{p},$$

$$C_s^2 = \frac{\mathsf{V}[B]}{(\mathsf{E}[B])^2} = p^2\left(\frac{1}{p^2} - \frac{1}{p}\right) = 1-p,$$

$$(1+C_s^2)/2 = 1 - p/2,$$

$$\mathsf{E}[L] = \left(1 - \frac{p}{2}\right)\frac{\rho}{1-\rho}\frac{1}{p} + \frac{1}{2}\frac{\rho}{1-\rho} = \frac{\rho}{1-\rho}\frac{1}{p}.$$

Can we check this in a simple way? If $\mathsf{P}(B=1) = f_1 = p = 1$, then $\mathsf{E}[L] = \rho/(1-\rho)$. Thus, we get the result for the $M/M/1$ queue. The result is at least consistent with earlier work.

s.2.9.4. Start with the simple case, $B \equiv 2$. Then $\mathsf{V}[B] = 0$ and $\mathsf{E}[B] = 2$. The load is $\rho = \lambda\,\mathsf{E}[B]\,\mathsf{E}[S] = 1 \cdot 2 \cdot 25/60 = 5/6$. Hence,

$$\mathsf{E}[L] = \frac{1}{2}\frac{5/6}{1/6}2 + \frac{1}{2}\frac{5/6}{1/6} = 5 + \frac{5}{2}.$$

Now the other case. $\mathsf{E}\left[B^2\right] = (1+4+9)/3 = 14/3$. Hence, $\mathsf{V}[B] = 14/3 - 4 = 2/3$. Hence,

$$C_s^2 = \frac{\mathsf{V}[B]}{(\mathsf{E}[B])^2} = \frac{2/3}{4} = \frac{1}{6}.$$

And thus,

$$\mathsf{E}[L] = \frac{1+1/6}{2}\frac{5/6}{1/6}2 + \frac{1}{2}\frac{5/6}{1/6} = \frac{7}{6}5 + \frac{5}{2}.$$

If we divide these two answers, we see that the ratio between $\mathsf{E}[L]$ for both answers is 10/9. In other words, we can reduce about 10% of the number of items in the system by working in fixed batch sizes.

Observe how easy it is with these models to get insight into the order of magnitude of queue length reductions or waiting times that can be achieved with changing work habits, such as making batch sizes constant rather than allowing them to vary. Observe also that it is up to management to decide whether such reductions outweigh any efforts to reduce the variation in batch sizes.

s.2.9.5. For the $M/M/1$ queue, each job contains just one item. Thus, $B \equiv 1$, hence $\mathsf{P}(B=1) = 1$, $\mathsf{E}\left[B^2\right] = \mathsf{E}[B] = 1$. Therefore, $\mathsf{E}\left[L_S^B(M/M/1)\right] = \rho$, and $\mathsf{E}[L(M/M/1)] = \rho/(1-\rho)$.

s.2.9.6.

$$\frac{\mathsf{E}\left[W_Q(M^X/M/1)\right]}{\mathsf{E}\left[W_Q(M/M/1)\right]} = \frac{\mathsf{E}\left[L_S(M^X/M/1)\right]}{\mathsf{E}[L_S(M/M/1)]} = \frac{\mathsf{E}\left[L_S(M^X/M/1)\right]}{\rho} = \frac{\mathsf{E}\left[B^2\right]}{2\mathsf{E}[B]} + \frac{1}{2}.$$

With this we can check whether this condition

$$1 \le \frac{\mathsf{E}\left[W_Q(M^X/M/1)\right]}{\mathsf{E}\left[W_Q(M/M/1)\right]} = \frac{\mathsf{E}\left[B^2\right]}{2\mathsf{E}[B]} + \frac{1}{2}$$

is always true. Clearly, it reduces to

$$\mathsf{E}[B] \leq \mathsf{E}\left[B^2\right].$$

Multiply this by $\mathsf{E}[B]$ for reasons to become clear presently to get

$$(\mathsf{E}[B])^2 \leq \mathsf{E}\left[B^2\right]\mathsf{E}[B].$$

So, the initial inequality is converted to this, and we like to know whether this always true.

To see this, we can use Jensen's inequality $\phi(\mathsf{E}[X]) \leq \mathsf{E}\left[\phi(X)\right]$ when ϕ is convex. In this case take $\phi(x) = x^2$, so that Jensen's inequality states that $(\mathsf{E}[B])^2 \leq \mathsf{E}\left[B^2\right]$. (BTW, note that Jensen's inequality implies that $\mathsf{V}[X] = \mathsf{E}\left[X^2\right] - (\mathsf{E}[X])^2 \geq 0$.) Now noting that $B \geq 1$, as a job minimally contains one item, we get

$$\begin{aligned}(\mathsf{E}[B])^2 &\leq \mathsf{E}\left[B^2\right], \quad \text{by Jensen's inequality}\\ &\leq \mathsf{E}\left[B^2\right]\mathsf{E}[B], \quad \text{as } B \geq 1.\end{aligned}$$

Clearly, this is the inequality we tried to show. As a result,

$$1 \leq \frac{\mathsf{E}\left[W_Q(M^X/M/1)\right]}{\mathsf{E}\left[W_Q(M/M/1)\right]}$$

for all B.

In conclusion, if work arrives in batches, the average number of jobs in the system increases, hence the average waiting time increases.

s.2.9.7.

$$\mathsf{E}\left[W_Q^B\right] = \mathsf{E}\left[L_Q^B\right]\mathsf{E}[B]\,\mathsf{E}[S] + \mathsf{E}\left[L_S^B\right]\mathsf{E}[S]\,\lambda\,\mathsf{E}\left[W_Q^B\right]\mathsf{E}[B]\,\mathsf{E}[+]\,\mathsf{E}\left[L_S^B\right]\mathsf{E}[S].$$

Use that $\rho = \lambda\,\mathsf{E}[B]\,\mathsf{E}[S]$, and simplify.

s.2.9.8. Check the hint. Now, suppose that the kth batch size is B (a random variable), then

$$\begin{aligned}X_k &= \int_{D_{k-1}}^{D_k} L_S^B(s)\,\mathrm{d}s\\ &= BS_1 + (B-1)S_2 + \cdots + 1S_B.\end{aligned}$$

Realize that the batch sizes are i.i.d., as are the service times, and the batch sizes and service times are also independent (by assumption). This implies that all X_k are distributed as a common random variable $X = BS_1 + (B-1)S_2 + \cdots + 1S_B$. With this,

$$\begin{aligned}\mathsf{E}[X] &= \mathsf{E}[BS_1 + (B-1)S_2 + \cdot 1S_B]\\ &= \mathsf{E}\left[\sum_{j=1}^{B} jS_{B+1-j}\right]\\ &= \mathsf{E}\left[\sum_{j=1}^{B} j\right]\mathsf{E}[S], \quad \text{by indep. of } S \text{ and } B\\ &= \mathsf{E}[B(B+1)/2]\,\mathsf{E}[S]\\ &= \frac{\mathsf{E}\left[B^2\right] + \mathsf{E}[B]}{2}\mathsf{E}[S].\end{aligned}$$

Here you should be careful: observe that we sum over a *random* number of summands. In such cases, it is not allowed to use the rule that the expectation of a sum is a sum of expectations!

s.2.9.9. From the above and 2.9.8,

$$\mathsf{E}\left[L_S^B\right] = Y = \delta X = \lambda \frac{\mathsf{E}\left[B^2\right] + \mathsf{E}[B]}{2} \mathsf{E}[S],$$

where we use that $\rho = \lambda \mathsf{E}[B]\,\mathsf{E}[S]$ and $\delta = \lambda$ (rate-stability).

s.2.9.10. We have

$$\begin{aligned} \frac{\mathsf{E}\left[B^2\right]}{(\mathsf{E}[B])^2} &= \frac{\mathsf{E}\left[B^2\right] - (\mathsf{E}[B])^2 + (\mathsf{E}[B])^2}{(\mathsf{E}[B])^2} \\ &= \frac{\mathsf{V}[B] + (\mathsf{E}[B])^2}{(\mathsf{E}[B])^2} = C_s^2 + 1. \end{aligned}$$

s.2.9.11. Use 2.9.10 and the definition of ρ to see that

$$\lambda \frac{\mathsf{E}\left[B^2\right]}{2(\mathsf{E}[B])^2} (\mathsf{E}[B])^2 \mathsf{E}[S] = \frac{1 + C_S^2}{2} \rho \mathsf{E}[B]\,\mathsf{E}[S].$$

The rest is straightforward.

2.10 $M/G/1$ QUEUE: EXPECTED WAITING TIME

Theory and Exercises

In many practical single-server queueing systems the service times are not really well approximated by the exponential distribution. The $M/G/1$ queue then becomes a better model than the $M/M/1$ queue. In this section we first present a formula to compute the average waiting time in queue for the $M/G/1$ queue, and then we derive it by means of sample path arguments. The derivation is also of general interest as it develops some general results of renewal theory.

The fundamentally important *Pollaczek-Khinchine formula*, or *PK formula*, for the average waiting time in queue for the $M/G/1$ queue has the form

$$\mathsf{E}\left[W_Q\right] = \frac{1 + C_s^2}{2} \frac{\rho}{1 - \rho} \mathsf{E}[S]. \tag{2.10.1}$$

Before deriving this formula, let us apply it.

2.10.1. *Show that when services are exponential, the expected waiting time* $\mathsf{E}\left[W_Q(M/G/1)\right]$ *reduces to* $\mathsf{E}\left[W_Q(M/M/1)\right]$.

2.10.2. *Compute* $\mathsf{E}\left[W_Q\right]$ *and* $\mathsf{E}[L]$ *for the* $M/D/1$ *queue.*

2.10.3. *Compute* $\mathsf{E}[L]$ *for the* $M/G/1$ *queue with* $S \sim U[0, \alpha]$.

2.10.4. *A queueing system receives Poisson arrivals at the rate of 5 per hour. The single server has a uniform service time distribution, with a range of 4 minutes to 6 minutes. Determine* $\mathsf{E}\left[L_Q\right]$, $\mathsf{E}[L]$, $\mathsf{E}\left[W_Q\right]$, $\mathsf{E}[W]$.

2.10.5. *Consider a workstation with just one machine. We model the job arrival process as a Poisson process with rate* $\lambda = 3$ *per day. The average service time* $\mathsf{E}[S] = 2$ *hours,* $C_s^2 = 1/2$*, and the shop is open for 8 hours. What is* $\mathsf{E}\left[W_Q\right]$*?*

Suppose the expected waiting time has to be reduced to 1h. How to achieve this?

2.10.6. *[Hall 5.16] The manager of a small firm would like to determine which of two people to hire. One employee is fast, on average, but somewhat inconsistent. The other is a bit slower, but very consistent. The first has a mean service time of* 2 *minutes, with a standard deviation of 1 minute. The second has a mean service time of* 2.1 *minutes, with a standard deviation of* 0.1 *minutes. If the arrival rate is Poisson with rate 20 per hour, which employee would minimize* $\mathsf{E}[L_Q]$*? Which would minimize* $\mathsf{E}[L]$*?*

2.10.7. *Show that for the M/G/*1 *queue, the expected idle time is* $\mathsf{E}[I] = 1/\lambda$*.*

2.10.8. *Express the utilization of the M/G/*1/1 *queue in terms of* λ *and* $\mathsf{E}[S]$*.*

2.10.9. *For the M/G/*1/1 *queue what is the fraction of jobs rejected?*

2.10.10. *Why is the fraction of lost jobs at a M/G/*1/1 *queue not necessarily the same as for a G/G/*1/1 *queue with the same load?*

To derive the PK-formula, suppose at first that we know the expected *remaining service time* $\mathsf{E}[S_r]$, i.e., the expected time it takes to complete the job in service, if present, at the time a job arrives.

2.10.11. *Show for the M/G/*1 *queue that the expected time in queue is*

$$\mathsf{E}[W_Q] = \mathsf{E}[S_r] + \mathsf{E}[L_Q]\,\mathsf{E}[S]. \tag{2.10.2}$$

2.10.12. *Show that, given* $\mathsf{E}[S_r]$*,*

$$\mathsf{E}[W_Q] = \frac{\mathsf{E}[S_r]}{1-\rho}. \tag{2.10.3}$$

It remains to compute the average remaining service time $\mathsf{E}[S_r]$ for generally distributed service times. Just like in Section 2.9 we use the renewal reward theorem. Consider the kth job of some sample path of the $M/G/1$ queueing process. Let its service time start at time $\tilde{A}_k$ so that it departs at time $D_k = \tilde{A}_k + S_k$.

2.10.13. *Use Fig. 13 to explain that the remaining service time of job* k *at time* s *is given by* $(D_k - s)\mathbb{1}_{\tilde{A}_k \le s < D_k}$*. With this, explain that*

$$Y(t) = \int_0^t (D_{D(s)+1} - s)\,\mathbb{1}_{L(s)>0}\,ds$$

is the total remaining service time as seen by the server up to t*.*

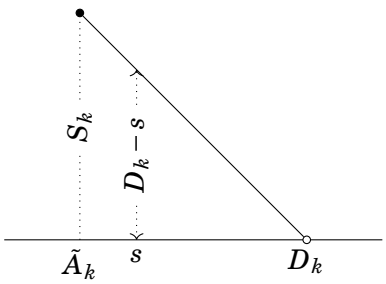


Figure 13: Remaining service time.

2.10.14. *Apply the renewal reward theorem to the result of 2.10.13 to prove that*

$$\mathsf{E}[S_r] = \frac{\lambda}{2}\mathsf{E}\left[S^2\right]. \tag{2.10.4}$$

Then simplify to get (2.10.1).

2.10.15. *Use* $C_s^2 = \mathsf{V}[S]/(\mathsf{E}[S])^2$ *to show that* $\lambda\mathsf{E}\left[S^2\right] = (1+C_s^2)\rho\,\mathsf{E}[S]$.

2.10.16. *Use the PASTA property to show that*

$$\mathsf{E}[S_r] = \rho\,\mathsf{E}[S_r \mid S_r > 0]. \tag{2.10.5}$$

2.10.17. *It is an easy mistake to think that* $\mathsf{E}[S_r] = \mathsf{E}[S]$ *when service times are exponential. Why is this wrong?*

2.10.18. *Show from* (2.10.4) *that*

$$\mathsf{E}[S_r \mid S_r > 0] = \frac{\mathsf{E}\left[S^2\right]}{2\mathsf{E}[S]}. \tag{2.10.6}$$

2.10.19. *When* $\mathsf{V}[S] = 0$, *is it true that*

$$\mathsf{E}[S_r \mid S_r > 0] = \frac{\mathsf{E}\left[S^2\right]}{2\mathsf{E}[S]} \implies \mathsf{E}[S_r \mid S_r > 0] = \frac{\mathsf{E}[S]}{2}?$$

2.10.20. *What is* $\mathsf{E}[S_r \mid S_r > 0]$ *for the* $M/D/1$ *queue?*

2.10.21. *Show that* $\mathsf{E}[S_r \mid S_r > 0] = \alpha/3$ *when* $S \sim U[0,\alpha]$.

2.10.22. *Show that* $\mathsf{E}[S_r \mid S_r > 0] = \mu^{-1}$ *when* $S \sim \mathrm{Exp}(\mu)$.

2.10.23. *A machine serves two types of jobs. The processing time of jobs of type* i, $i = 1,2$, *is exponentially distributed with parameter* μ_i. *The type* T *of a job is random and independent of anything else, and such that* $\mathsf{P}(T=1) = p = 1-q = 1-\mathsf{P}(T=2)$. *(An example is a desk serving men and women, both requiring different average service times, and* p *is the probability that the customer in service is a man.) Show that the expected processing time and variance are given by*

$$\begin{aligned}\mathsf{E}[S] &= p\,\mathsf{E}[S_1] + q\,\mathsf{E}[S_2]\\ \mathsf{V}[S] &= p\,\mathsf{V}[S_1] + q\,\mathsf{V}[S_2] + pq(\mathsf{E}[S_1] - \mathsf{E}[S_2])^2.\end{aligned}$$

Interestingly, we see that even if $\mathsf{V}[S_1] = \mathsf{V}[S_2] = 0$, $\mathsf{V}[S] > 0$ *if* $\mathsf{E}[S_1] \neq \mathsf{E}[S_2]$. *Bear this in mind; we will use these ideas later when we discuss the effects of failures on the variance of service times of jobs.*

Hints

h.2.10.2. Use that $\mathsf{V}[(]S) = 0$ for the $M/D/1$ queue.

h.2.10.3. Integrate $\alpha^{-1}\int_0^\alpha x\,\mathrm{d}x$, and likewise for the second moment.

h.2.10.7. What is the average time between two arrivals? Observe that the inter-arrivals are memoryless, hence the average time until the next arrival after the server becomes idle is also $1/\lambda$.

h.2.10.9. The rate of accepted jobs is $\lambda\pi(0)$. What is the load of these jobs? Equate this to $1-\pi(0)$ as this must also be the load. Then solve for $\pi(0)$.

h.2.10.10. Provide an example.

h.2.10.11. What happens when you enter a queue? First you have to wait until the job in service (if there is any) completes, and then you have to wait for the queue to clear.

h.2.10.12. Use 2.10.11. Then use Little's law to see that $\mathsf{E}\left[L_Q\right] = \lambda\,\mathsf{E}\left[W_Q\right]$. Substitute this in (2.10.2) and simplify.

h.2.10.14. Choose $T_k = D_k$ as epochs in the renewal reward theorem.

h.2.10.16. Use 2.7.10.

h.2.10.17. Realize again that $\mathsf{E}[S_r]$ includes the jobs that arrive at an empty system.

h.2.10.19. $\mathsf{V}[S] = 0$ implies that S is deterministic.

h.2.10.23. Let S be the processing (or service) time at the server, and S_i the service time of a type i job. Then,

$$S = \mathbb{1}_{T=1}S_1 + \mathbb{1}_{T=2}S_2,$$

where $\mathbb{1}_{\cdot}$ is the indicator function, that is, $\mathbb{1}_A = 1$ if the event A is true, and $\mathbb{1}_A = 0$ if A is not true.

Solutions

s.2.10.1. Since the SCV for the exponential distribution is 1, hence $C_s^2 = 1$, we get

$$\mathsf{E}\left[W_Q\right] = \frac{1+1}{2}\frac{\rho}{1-\rho}\mathsf{E}[S] = \frac{\rho}{1-\rho}\mathsf{E}[S],$$

which we also found in a previous section.

s.2.10.2. For the $M/D/1$ the service time is deterministic. Thus, the service time $S = T$ always. Hence, $\mathsf{E}[S] = T$ and $\mathsf{E}\left[S^2\right] = T^2$, hence $\mathsf{V}[S] = \mathsf{E}\left[S^2\right] - (\mathsf{E}[S])^2 = 0$, hence $C_s^2 = 0$. From the Pollaczek-Khinchine formula, the first term is $1 + C_s^2 = 1$ for the $M/D/1$ queue, and $1 + C_s^2 = 2$ for the $M/M/1$ queue. Hence,

$$\mathsf{E}\left[W_Q(M/D/1)\right] = \frac{\mathsf{E}\left[W_Q(M/M/1)\right]}{2}.$$

Therefore,

$$\begin{aligned}\mathsf{E}[L(M/D/1)] = \mathsf{E}\left[L_Q(M/D/1)\right] + \rho &= \frac{\mathsf{E}\left[L_Q(M/M/1)\right]}{2} + \rho \\ &= \frac{\rho^2}{2(1-\rho)} + \rho = \frac{\rho(2-\rho)}{2(1-\rho)}.\end{aligned}$$

s.2.10.3.

$$\begin{aligned}
\mathsf{E}[S] &= \alpha/2,\\
\mathsf{E}\left[S^2\right] &= \int_0^\alpha x^2\,\mathrm{d}x/\alpha = \alpha^2/3,\\
\mathsf{V}[S] &= \alpha^2/3 - \alpha^2/4 = \alpha^2/12,\\
C_s^2 &= (\alpha^2/12)/(\alpha^2/4) = 1/3,\\
\rho &= \lambda\alpha/2,\\
\mathsf{E}\left[W_Q\right] &= \frac{1+C_s^2}{2}\frac{\lambda\alpha/2}{1-\lambda\alpha/2}\frac{\alpha}{2},\\
\mathsf{E}[W] &= \mathsf{E}\left[W_Q\right] + \frac{\alpha}{2},\\
\mathsf{E}[L] &= \lambda\,\mathsf{E}[W].
\end{aligned}$$

s.2.10.4. First the load.

```
>>> labda = 5./60 # arrivals per minute
>>> a = 4.
>>> b = 6.
>>> ES = (a+b)/2. # service time in minutes
>>> rho = labda*ES
>>> rho
0.41666666666666663
```

Next, the variance and SCV. With this the waiting times follow right away.

```
>>> Var = (b-a)*(b-a)/12.
>>> SCV = Var/(ES**2)


>>> Wq = (1+SCV)/2.*rho/(1.-rho)*ES
>>> Wq # in minutes
1.8095238095238095
>>> Wq/60. # in hour
0.03015873015873016


>>> W = Wq + ES
>>> W
6.809523809523809
>>> Lq = labda*Wq
>>> Lq
0.15079365079365079
>>> L = labda*W
>>> L
0.5674603174603174
```

s.2.10.5. $\mathsf{E}\left[W_Q\right] = 4.5$ h. $\rho = \lambda\,\mathsf{E}[S] = (3/8)\cdot 2 = 3/4$.

One way to increase the capacity/reduce the average service time is to choose $\mathsf{E}[S] = 1$ hour and reduce C_s^2 to 1/4. Of course, there are many more ways. Reducing C_S^2 to zero is (nearly) impossible or very costly. Hence, ρ must go down, i.e., capacity up or service time down. Another possibility is to plan the arrival of jobs, i.e., reduce the variability in the arrival process. However, typically this is not possible. For instance, would you accept this as a customer?

s.2.10.6. The arrival process is assumed to be Poisson. There is also just one server. Hence, we can use the PK formula to compute the average queue length.

```
>>> labda = 20./60 # per minute
>>> ES = 2. # minutes
>>> sigma = 1.
>>> SCV = sigma*sigma/(ES*ES)
>>> rho = labda*ES
>>> rho
0.6666666666666666

>>> Wq = (1+SCV)/2 * rho/(1-rho) * ES
>>> Wq
2.4999999999999996
>>> Lq = labda * Wq # Little's law
>>> Lq
0.8333333333333331
>>> W = Wq + ES
>>> W
4.5
>>> L = labda * W
>>> L
1.5

>>> ES = 2.1
>>> SD = 0.1
>>> SCV = SD**2/ES**2
>>> rho = labda*ES
>>> rho
0.7

>>> Lq = rho**2/(1.-rho)*(1.+SCV)/2.
>>> Lq
0.8185185185185182
>>> L = rho + Lq
>>> L
1.5185185185185182
```

s.2.10.7. Since the inter-arrival times are memoryless, the expected time to the first arrival after the system becomes empty is also $\mathsf{E}[X] = 1/\lambda$.

s.2.10.8. The system can contain at most 1 job. Necessarily, if the system contains a job, this job must be in service. All jobs that arrive while the server is busy are rejected. Just after a

departure, the average time until the next arrival is $1/\lambda$, and then a new service starts with an average duration of $\mathsf{E}[S]$. After this departure, a new cycle starts. Thus, the utilization is $\mathsf{E}[S]/(1/\lambda + \mathsf{E}[S]) = \lambda\,\mathsf{E}[S]/(1+\lambda\,\mathsf{E}[S])$. Since not all jobs are accepted, the utilization ρ cannot be equal to $\lambda\,\mathsf{E}[S]$.

s.2.10.9. Let $p(0)$ be the fraction of time the server is idle. By PASTA, $\pi(0) = p(0)$. Thus, the rate of accepted jobs is $\lambda\pi(0)$. Therefore, the departure rate $\delta = \lambda\pi(0)$. The loss rate is $\lambda - \delta = \lambda(1-\pi(0))$.

Since $\lambda\pi(0)$ is the rate at which jobs enter the system, the load must be $\lambda\pi(0)\,\mathsf{E}[S]$. Since the load is also $1-\pi(0)$, it follows from equating that

$$\lambda\pi(0)\,\mathsf{E}[S] = 1-\pi(0) \iff \pi(0) = \frac{1}{1+\lambda\,\mathsf{E}[S]} \iff 1-\pi(0) = \frac{\lambda\,\mathsf{E}[S]}{1+\lambda\,\mathsf{E}[S]},$$

which is the same as in the previous problem.

Note that $\delta < \lambda$, as it should be the case.

s.2.10.10. Typically, in the $G/G/1$ queue, the arrivals do not see time-averages. Consequently, the fraction of arrivals that are blocked is not necessarily equal to the utilization ρ.

Again, take jobs with a duration 59 minutes and inter-arrival times of 1 hour. The load is 59/60, but no job is lost, also not in the $G/G/1/1$ case. Thus, $\delta = \lambda$ in this case.

s.2.10.11. It is evident that the expected waiting time for an arriving customer is the expected remaining service time plus the expected time in queue. The expected time in queue must be equal to the expected number of customers in queue at an arrival epoch times the expected service time per customer, assuming that service times are i.i.d. If the arrival process is Poisson, it follows from PASTA that the average number of jobs in queue perceived by arriving customers is also the *time-average* number of jobs in queue $\mathsf{E}\left[L_Q\right]$.

s.2.10.12. With Little's law $\mathsf{E}\left[L_Q\right] = \lambda\,\mathsf{E}\left[W_Q\right]$. Using this,

$$\mathsf{E}\left[W_Q\right] = \mathsf{E}[S_r] + \lambda\,\mathsf{E}\left[W_Q\right]\mathsf{E}[S] = \mathsf{E}[S_r] + \rho\,\mathsf{E}\left[W_Q\right],$$

since $\rho = \lambda\,\mathsf{E}[S]$. But this gives for the $M/G/1$ queue that

$$\mathsf{E}\left[W_Q\right] = \frac{\mathsf{E}[S_r]}{1-\rho}.$$

s.2.10.13. Observe that when $s \in [\tilde{A}_k, D_k)$, the remaining service time until job k departs is $D_k - s$, while if $s \notin [\tilde{A}_k, D_k)$, job k is not in service so it cannot have any remaining service.

At time s, the number of departures is $D(s)$. Thus, $D(s)+1$ is the first job to depart after time s. The departure time of this job is $D_{D(t)+1}$, hence the remaining service time at time s is $D_{D(s)+1} - s$, provided this job is in service.

s.2.10.14. It is clear that the time-average $Y(t)/t \to \mathsf{E}[S_r]$. Moreover, $X_k = Y(D_k) - Y(D_{k-1})$ is the area under the triangle in Fig. 13. Thus, $X = \lim_{n\to\infty} \lim_{k=1}^{n} S_k^2/2 = \mathsf{E}\left[S^2\right]/2$. Finally, $\delta = \lambda$ by rate-stability.

The rest is plain algebra, see 2.10.15.

s.2.10.15. Use that

$$\frac{\mathsf{E}\left[S^2\right]}{(\mathsf{E}[S])^2} = \frac{(\mathsf{E}\left[S^2\right] - (\mathsf{E}[S])^2) + (\mathsf{E}[S])^2}{(\mathsf{E}[S])^2} = \frac{\mathsf{V}[S] + (\mathsf{E}[S])^2}{(\mathsf{E}[S])^2} = C_s^2 + 1.$$

Then

$$\lambda\,\mathsf{E}\left[S^2\right] = \frac{\mathsf{E}\left[S^2\right]}{(\mathsf{E}[S])^2}\lambda(\mathsf{E}[S])^2 = \frac{\mathsf{E}\left[S^2\right]}{(\mathsf{E}[S])^2}\rho\,\mathsf{E}[S] = (1 + C_s^2)\rho\,\mathsf{E}[S].$$

s.2.10.16. By the PASTA property we know that ρ is the probability to find the server busy upon arrival; hence $1-\rho$ is the probability that the server is idle upon arrival. Then,

$$\mathsf{E}[S_r] = \rho\,\mathsf{E}[S_r \mid S_r > 0] + (1-\rho)\,\mathsf{E}[S_r \mid S_r = 0] = \rho\,\mathsf{E}[S_r \mid S_r > 0],$$

since, evidently, $\mathsf{E}[S_r \mid S_r = 0] = 0$.

s.2.10.17. $\mathsf{E}[S_r \mid S_r > 0] = \mathsf{E}[S]$ for the $M/M/1$ queue, and $\mathsf{E}[S_r] = \rho\,\mathsf{E}[S_r \mid S_r > 0]$ for the $M/G/1$ queue, it follows that

$$\mathsf{E}[S_r] = \rho\,\mathsf{E}[S_r \mid S_r > 0] = \rho\,\mathsf{E}[S].$$

s.2.10.18. From (2.10.5) and the sentences above this equation, we have that

$$\mathsf{E}[S_r] = \rho\,\mathsf{E}[S_r \mid S_r > 0] = \lambda\,\mathsf{E}[S]\,\mathsf{E}[S_r \mid S_r > 0].$$

Hence,

$$\lambda\,\mathsf{E}[S_r \mid S_r > 0] = \frac{\mathsf{E}[S_r]}{\mathsf{E}[S]} = \lambda\frac{\mathsf{E}\left[S^2\right]}{2\,\mathsf{E}[S]}.$$

s.2.10.19. When S is deterministic $\mathsf{E}\left[S^2\right] = (\mathsf{E}[S])^2$. Now use (2.10.6) to see that it is true.

s.2.10.20. By 2.10.19, it is half the service time.

s.2.10.21. When S is uniform on $[0, \alpha]$,

$$\mathsf{E}[S] = \alpha^{-1}\int_0^\alpha x\,\mathrm{d}x = \frac{\alpha^2}{2\alpha} = \frac{\alpha}{2}, \qquad \mathsf{E}\left[S^2\right] = \alpha^{-1}\int_0^\alpha x^2\,\mathrm{d}x = \frac{\alpha^2}{3}.$$

Thus, with (2.10.6), $\mathsf{E}[S_r \mid S_r > 0] = (\alpha^2/3)/(2\alpha/2) = \alpha/3$.

s.2.10.22.

$$\begin{aligned}
\mathsf{E}[S] &= \mu\int_0^\infty xe^{-\mu x}\,\mathrm{d}x = 1/\mu,\\
\mathsf{E}\left[S^2\right] &= \mu\int_0^\infty x^2e^{-\mu x}\,\mathrm{d}x = -\,x^2e^{-\mu x}\Big|_0^\infty + 2\int_0^\infty xe^{-\mu x}\,\mathrm{d}x\\
&= -\,2\frac{x}{\mu}e^{-\mu x}\Big|_0^\infty + \frac{2}{\mu}\int_0^\infty e^{-\mu x}\,\mathrm{d}x = \frac{2}{\mu^2}.
\end{aligned}$$

Now use (2.10.6).

s.2.10.23. With the hint,

$$\begin{aligned}
\mathsf{E}[S] &= \mathsf{E}[\mathbb{1}_{T=1}S_1] + \mathsf{E}[\mathbb{1}_{T=2}S_2] \\
&= \mathsf{E}[\mathbb{1}_{T=1}]\,\mathsf{E}[S_1] + \mathsf{E}[\mathbb{1}_{T=2}]\,\mathsf{E}[S_2], \text{ by the independence of } T, \\
&= \mathsf{P}(T=1)/\mu_1 + \mathsf{P}(T=2)/\mu_2 \\
&= p/\mu_1 + q/\mu_2 \\
&= p\,\mathsf{E}[S_1] + q\,\mathsf{E}[S_2].
\end{aligned}$$

(The next derivation may seem a bit long, but the algebra is standard. I include all steps so that you don't have to use pen and paper yourself if you want to check the result.) Next, using that

$$\mathbb{1}_{T=1}\,\mathbb{1}_{T=2} = 0 \text{ and } \mathbb{1}_{T=1}^2 = \mathbb{1}_{T=1},$$

we get

$$\begin{aligned}
\mathsf{V}[S] &= \mathsf{E}\left[S^2\right] - (\mathsf{E}[S])^2 \\
&= \mathsf{E}\left[(\mathbb{1}_{T=1}S_1 + \mathbb{1}_{T=2}S_2)^2\right] - \left(\frac{p}{\mu_1} + \frac{q}{\mu_2}\right)^2 \\
&= \mathsf{E}\left[\mathbb{1}_{T=1}S_1^2 + \mathbb{1}_{T=2}S_2^2\right] - \left(\frac{p}{\mu_1} + \frac{q}{\mu_2}\right)^2 \\
&= p\,\mathsf{E}\left[S_1^2\right] + q\,\mathsf{E}\left[S_2^2\right] - \left(\frac{p}{\mu_1} + \frac{q}{\mu_2}\right)^2 \\
&= p\,\mathsf{V}[S_1] + p(\mathsf{E}[S_1])^2 + q\,\mathsf{V}[S_2] + q(\mathsf{E}[S_2])^2 - \left(\frac{p}{\mu_1} + \frac{q}{\mu_2}\right)^2 \\
&= p\,\mathsf{V}[S_1] + \frac{p}{\mu_1^2} + q\,\mathsf{V}[S_2] + \frac{q}{\mu_2^2} - \left(\frac{p}{\mu_1} + \frac{q}{\mu_2}\right)^2 \\
&= p\,\mathsf{V}[S_1] + q\,\mathsf{V}[S_2] + \frac{p}{\mu_1^2} + \frac{q}{\mu_2^2} - \frac{p^2}{\mu_1^2} - \frac{q^2}{\mu_2^2} - \frac{2pq}{\mu_1\mu_2} \\
&= p\,\mathsf{V}[S_1] + q\,\mathsf{V}[S_2] + \frac{p(1-p)}{\mu_1^2} + \frac{q(1-q)}{\mu_2^2} - \frac{2pq}{\mu_1\mu_2} \\
&= p\,\mathsf{V}[S_1] + q\,\mathsf{V}[S_2] + \frac{pq}{\mu_1^2} + \frac{qp}{\mu_2^2} - \frac{2pq}{\mu_1\mu_2} \\
&= p\,\mathsf{V}[S_1] + q\,\mathsf{V}[S_2] + pq(\mathsf{E}[S_1] - \mathsf{E}[S_2])^2.
\end{aligned}$$

2.11 $M^X/M/1$ QUEUE LENGTH DISTRIBUTION

Theory and Exercises

In Sections 2.9 and 2.10 we established the Pollaczek-Khinchine formula for the waiting times of the $M^X/M/1$ queue and the $M/G/1$ queue, respectively. To compute more difficult performance measures, for instance, the loss probability $\mathsf{P}(L > n)$, we need expressions for the stationary distribution $\pi(n) = \mathsf{P}(L = n)$ of the number of jobs in the system. Here we present a numerical, recursive, scheme to compute these probabilities.

To find $\pi(n)$, $n = 0, 1, \ldots$, we turn again to level-crossing arguments. However, the reasoning that led to the level-crossing equation (2.4.3) needs to be generalized. To see this, we consider

an example. If $L(t) = 3$, the system contains 3 items. (This is not necessarily the same as 3 batches.) Since the server serves single items, down-crossings of level $n = 3$ occur in single units. However, due to the batch arrivals, when a job arrives it typically brings multiple items to the queue. For instance, suppose that $L(A_k-) = 3$, i.e., job k sees 3 items in the system at its arrival epoch. If its size $B_k = 20$, then right after the kth arrival the system contains 23 items, that is, $L(A_k) = 3 + 20 = 23$. Thus, upon the arrival of job k, all levels between states 3 and 23 are crossed.

The left panel in Fig. 14 shows all up- and down-crossings of some level n. The down-crossing rate is easy: just as in Fig. 9 there is just one arrow from right to left. However, level n can be up-crossed from below from many states, in fact from any level $m \in \{0, 1, \ldots, n-1\}$. More formally, to count the number of up-crossings define

$$A(m,n,t) = \sum_{k=1}^{A(t)} \mathbb{1}_{L(A_k-)=m}\, \mathbb{1}_{B_k>n-m}$$

as the number of jobs up to time t that see m in the system upon arrival and have batch size larger than $n-m$.

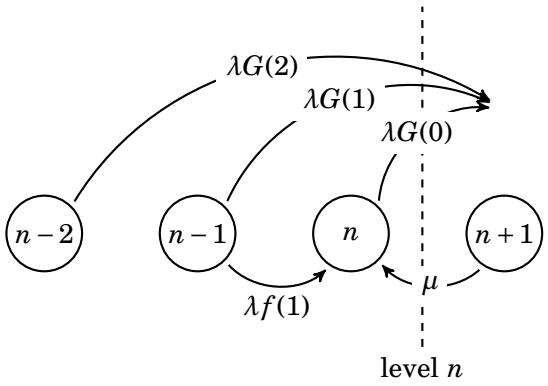


Figure 14: Level crossing of level n. Observe that when the system is in state $n-2$, the arrival of any batch larger than 2 ensures that level n is crossed from below. The rate at which such events happen is $\lambda\pi(n-2)G(2)$. Similarly, in state $n-1$, the arrival of any batch larger than one item ensures that level n is crossed, and this occurs with rate $\lambda\pi(n-1)G(1)$, and so on.

2.11.1. *Show that $A(n,n,t) = A(n,t)$, where $A(n,t)$ is defined by* (2.4.1a).

2.11.2. *Provided the limit exists, show that*

$$\lim_{t\to\infty} \frac{A(m,n,t)}{A(m,t)} = \mathsf{P}(B > n-m \,|\, L(A-) = m), \tag{2.11.1}$$

where the random variable $L(A-)$ denotes the number in the system seen by an arbitrary arrival.

2.11.3. *Show that* $\mathsf{P}(B > n-m \,|\, L(A-) = m) = \mathsf{P}(B > n-m)$.

2.11.4. *Assuming that the limits exist, show that*

$$\lim_{t\to\infty} \frac{A(m,n,t)}{t} = \lambda\pi(m)G(n-m).$$

Equating the number of up- and down-crossing gives $\sum_{m=0}^{n} A(m,n,t) \approx D(n,t)$. Then, dividing by t, taking the limit $t \to \infty$, and using 2.11.4 results in the level-crossing equation for the $M^X/M/1$ queue:

$$\lambda \sum_{m=0}^{n} \pi(m) G(n-m) = \mu \pi(n+1). \tag{2.11.2}$$

2.11.5. *Provide an interpretation of* (2.11.2) *in terms of a thinned Poisson arrival process.*

2.11.6. *Show that* (2.11.2) *reduces to* $\lambda\pi(n) = \mu\pi(n+1)$ *for the* $M/M/1$ *case.*

It is left to find the normalization constant. As the recursion (2.11.2) does not lead to a closed form expression for $\pi(n)$, such as (2.5.1), we need to use a criterion to stop this iterative procedure. It is not so easy to find general conditions when to stop, but we can use a pragmatic approach. When the demand is finite, the numbers $\{\pi(k)\}$ should decrease geometrically fast for all $k \geq N$ where N[3] is some large number. Stop when $\pi(N) \ll \pi(0)$, and take $\sum_{i=0}^{N} \pi(i)$ as the normalization constant.

Once we have $\pi(n)$, we can compute the influence on the batch size distribution, λ, and μ on the system's performance.

2.11.7. *Why is* (2.7.7)*, i.e.,* $\pi(n) = \delta(n)$*, not true for the* $M^X/M/1$ *batch queue? Provide an example.*

2.11.8. *Show that*

$$\mu \mathsf{E}[L] = \mu \sum_{n=0}^{\infty} n\pi(n) = \lambda \frac{\mathsf{E}\left[B^2\right]}{2} + \lambda \mathsf{E}[B]\,\mathsf{E}[L] + \lambda \frac{\mathsf{E}[B]}{2}. \tag{2.11.3}$$

2.11.9. *Use* (2.11.3) *and the definition* $\rho = \lambda \mathsf{E}[B]/\mu$ *to show that*

$$(1-\rho)\mathsf{E}[L] = \frac{\lambda}{\mu}\frac{\mathsf{E}\left[B^2\right]}{2} + \frac{\rho}{2}.$$

2.11.10. *Substitute recursion* (2.11.2) *for* $\pi(n)$ *into the expression* $\mathsf{E}[L] = \sum_{n=0}^{\infty} n\pi(n)$ *and derive* (2.9.2)*.*

2.11.11. *Implement the recursion* (2.11.2) *in a computer program for the case* $f(1) = f(2) = f(3) = 1/3$*. Take* $\lambda = 1$ *and* $\mu = 3$*.*

We consider the $M^X/M/1/K$ queue, i.e., a batch queue in which at most K jobs fit into the system. When customers can be blocked in a batch queue it is necessary to specify a policy that decides which items in a batch to accept. Three common rules are

1. Complete rejection: if a batch does not fit entirely into the system, it will be rejected completely.

2. Partial acceptance: accept whatever fits of a batch, and reject the rest.

3. Complete acceptance: accept all batches that arrive when the system contains K or less jobs, and reject the entire batch otherwise.

2.11.12. *Derive a set of recursions, analogous to* (2.11.2)*, to compute* $\pi(n)$ *for the* $M^X/M/1/K$ *queue with complete rejection.*

[3] An interesting question, why should it decrease monotonically after some, large, N?

2.11.13. *Derive a set of recursions, analogous to* (2.11.2)*, to compute* $\pi(n)$ *for the* $M^X/M/1/K$ *queue with complete acceptance.*

2.11.14. *Derive a set of recursions, analogous to* (2.11.2)*, to compute* $\pi(n)$ *for the* $M^X/M/1/K$ *queue with partial acceptance.*

2.11.15. *[Batch services, not obligatory] An interesting extension is a queueing process with batch services, i.e., the* $M/M^Y/1$ *queue. Constructing a recursion for the steady-state probabilities* $\pi(n)$ *for this case is not hard, in fact, mostly analogous to* (2.11.2)*. However, solving the recursion appears to be quite a bit harder.*

Hints

h.2.11.3. Realize that B and $L(A-)$ are assumed to be independent.

h.2.11.8. Substitute the recursion, and carry on with the algebra. Use also the results of the exercises of Section 2.9.

Solutions

s.2.11.1. $A(n,n,t)$ counts all jobs up to time t that see n items and bring at least $n-n+1=1$ unit of work. As each job brings at least 1 item, $A(n,n,t)$ counts all jobs that see n items at arrival.

s.2.11.2.

$$\lim_{t\to\infty}\frac{A(m,n,t)}{A(m,t)}=\lim_{t\to\infty}\frac{\sum_{k=1}^{A(t)}\mathbb{1}_{L(A_k-)=m,B_k>n-m}}{\sum_{k=1}^{A(t)}\mathbb{1}_{L(A_k-)=m}}=\mathsf{P}(B>n-m\,|\,L(A-)=m),$$

s.2.11.3.

$$\begin{aligned}\mathsf{P}(B>n-m\,|\,L(A-)=m)&=\frac{\mathsf{P}(B>n-m,L(A-)=m)}{\mathsf{P}(L(A-)=m)}\\&=\frac{\mathsf{P}(B>n-m)\mathsf{P}(L(A-)=m)}{\mathsf{P}(L(A-)=m)}\\&=\mathsf{P}(B>n-m)=G(n-m).\end{aligned}$$

s.2.11.4. For this purpose, observe that we can write

$$\frac{A(m,n,t)}{t}=\frac{A(t)}{t}\frac{A(m,t)}{A(t)}\frac{A(m,n,t)}{A(m,t)}. \tag{2.11.4}$$

By the assumptions of Section 2.7, $A(t)/t\to\lambda$ and $A(m,t)/A(t)\to\pi(m)$. Next, apply the results of 2.11.2 to 2.11.3.

s.2.11.5. (2.11.4) has the interpretation that the rate at which level n is crossed from below from state m is equal to the rate at which jobs arrive times the fraction of jobs that see m jobs in the system times the fraction of jobs with batch size larger than $n-m$. Observe that the stream of jobs with batch size larger than $n-m$ is a Poisson process thinned at rate $G(n-m)$.

s.2.11.6. The right-hand side of (2.11.2) is identical, so we only have to concentrate on the left-hand side. In the $M/M/1$ queue, all batches have size 1. Thus, $\mathsf{P}(B=1)=f(1)=1$ and $f(k)=0$ for $k\neq 1$. Thus, $G(0)=1$ and $G(1)=G(2)=\cdots=0$. Thus, $\sum_{m=0}^{n} G(n-m)\pi(m)=G(0)\pi(n)=\pi(n)$.

s.2.11.7. Using 2.7.4 that $D(n,t)/t\to\delta\delta(n)$, we see that $\lambda\sum_{m=0}^{n}\pi(m)G(n-m)=\delta\delta(n)$. With rate-stability, $\lambda=\delta$, and therefore $\sum_{m=0}^{n}\pi(m)G(n-m)=\delta(n)$. The left-hand-side is clearly not equal to $\pi(n)$.

s.2.11.8. We use that $\mu\pi(n)=\lambda\sum_{i=0}^{n-1}\pi(i)G(n-1-i)$ and the results of the exercises of Section 2.9 to see that

$$
\begin{aligned}
\mu\,\mathsf{E}[L] &= \sum_{n=0}^{\infty} n\,\mu\pi(n), \quad \text{now substitute for } \mu\pi(n) \text{ the recursion (2.11.2),}\\
&= \lambda\sum_{n=0}^{\infty} n\sum_{i=0}^{n-1}\pi(i)G(n-1-i) = \lambda\sum_{n=0}^{\infty} n\sum_{i=0}^{\infty}\mathbb{1}_{i<n}\pi(i)G(n-1-i)\\
&= \lambda\sum_{i=0}^{\infty}\pi(i)\sum_{n=0}^{\infty}\mathbb{1}_{i<n}nG(n-1-i) = \lambda\sum_{i=0}^{\infty}\pi(i)\sum_{n=i+1}^{\infty}nG(n-1-i)\\
&= \lambda\sum_{i=0}^{\infty}\pi(i)\sum_{n=0}^{\infty}(n+i+1)G(n) = \lambda\sum_{i=0}^{\infty}\pi(i)\left[\sum_{n=0}^{\infty}nG(n)+(i+1)\sum_{n=0}^{\infty}G(n)\right]\\
&= \lambda\sum_{i=0}^{\infty}\pi(i)\sum_{n=0}^{\infty}nG(n)+\lambda\,\mathsf{E}[B]\sum_{i=0}^{\infty}\pi(i)(i+1)\\
&= \lambda\sum_{i=0}^{\infty}\pi(i)\frac{\mathsf{E}\left[B^2\right]-\mathsf{E}[B]}{2}+\lambda\,\mathsf{E}[B](\mathsf{E}[L]+1)\\
&= \lambda\frac{\mathsf{E}\left[B^2\right]-\mathsf{E}[B]}{2}+\lambda\,\mathsf{E}[B]\,\mathsf{E}[L]+\lambda\,\mathsf{E}[B]\\
&= \lambda\frac{\mathsf{E}\left[B^2\right]}{2}+\lambda\,\mathsf{E}[B]\,\mathsf{E}[L]+\lambda\frac{\mathsf{E}[B]}{2}.
\end{aligned}
$$

s.2.11.9. Dividing both sides of (2.11.3) by μ and using that $\lambda\,\mathsf{E}[B]/\mu=\rho$,

$$\mathsf{E}[L]=\frac{\lambda}{\mu}\frac{\mathsf{E}\left[B^2\right]}{2}+\rho\,\mathsf{E}[L]+\frac{\rho}{2}.$$

s.2.11.10. We chop this up into two steps, see 2.11.8 and 2.11.9.

s.2.11.11. The recursion for n becomes

$$\pi(n)=\frac{\lambda}{\mu}\sum_{i=0}^{n-1}\pi(n-1-i)G(i).$$

Since $G(i)=0$ for $i\geq 3$ we rewrite this to

$$\pi(n)=\frac{\lambda}{\mu}\sum_{i=0}^{\min\{n-1,\operatorname{len} G\}}\pi(n-1-i)G(i),$$

where len G is the largest possible batch size.

The following code carries out the recursion for the $M/M/1$ queue and compares the result to the *unnormalized* probabilities of the $M/M/1$ queue. Recall, $(\lambda/\mu)^n$ are these unnormalized probabilities. With this we can test the code right away.

```
>>> import numpy as np

>>> f = [0, 1] # set f[0] = 0
>>> F = np.cumsum(f) # distribution
>>> G = np.ones_like(F) - F # survivor function

>>> labda = 1
>>> mu = 3
>>> num = 4
>>> p = np.ones(num)

>>> for n in range(1, num):
...  p[n] = labda / mu * sum(p[n - 1 - i] * G[i] for i in range(min(n, len(G))))
...  print(n, p[n], (labda / mu)**n)
...
1 0.3333333333333333 0.3333333333333333
2 0.1111111111111111 0.1111111111111111
3 0.037037037037037035 0.03703703703703703
```

The test is convincing. So now we'll move on the real problem.

```
>>> f = np.ones(4) / 3
>>> f[0] = 0
>>> F = np.cumsum(f)
>>> G = np.ones_like(F) - F
>>> num = 30 # stop at 80
>>> p = np.ones(num)
>>> for n in range(1, num):
...  p[n] = labda / mu * sum(p[n - 1 - i] * G[i] for i in range(min(n, len(G))))
...
>>> p /= p.sum() # normalize
```

Now that we have the probabilities we can do all experiments we like.

```
>>> EL = sum(n*p[n] for n in range(len(p)))
>>> EL
3.309069824793065
```

It is of interest to compare this result to (2.9.2). In fact, I spent at least one hour to get (2.9.2) correct; now I can finally check it numerically.

```
>>> EB = sum(k*fk for k,fk in enumerate(f))
>>> rho=labda*EB/mu
>>> EB2 = sum(k*k*fk for k,fk in enumerate(f))
>>> VB = EB2 - EB*EB
>>> C2 = VB/EB/EB
>>> EL = (1+C2)/2*rho/(1-rho)*EB + rho/(1-rho)/2
>>> EL
3.3333333333333326
```

So, after all, stopping at 30 is not quite OK: there is a slight difference between the two expectations. Let's run the recursion up to 100, and see what we get then.

```
>>> num = 100
>>> p = np.ones(num)

>>> for n in range(1, num):
...  p[n] = labda / mu * sum(p[n - 1 - i] * G[i] for i in range(min(n, len(G))))
...
>>> p /= p.sum() # normalize
>>> EL = sum(n*p[n] for n in range(len(p)))
>>> print(EL)
3.3333333271183934
```

This does the job.

s.2.11.12. Suppose a batch of size k arrives when the system contains n jobs. When $k+n \leq K$, the batch can be accepted since the entire batch will fit into the queue. When, however, $k+n > K$, the batch has to be rejected.

Now consider an imaginary line between states with n and $n+1$ jobs in the system. This imaginary line separates the state space into two disjoint parts: one part with states $0,1,\ldots,n$ and the other part with states $n+1,n+2,\ldots,K$. We call this imaginary line 'level n'. We call an 'up-crossing' a transition from some state $m \leq n$ to a state $l > n$. Likewise, a 'down-crossing' of level n is a transition from some state $l > n$ to some state $m \leq n$.

If the system contains n jobs, level n is crossed from below with rate $\lambda\pi(n)\mathsf{P}(B \leq K-n)$. More generally, when the system is in state $m \leq n$, we need a batch of at least $n+1-m$ to cross level n. Moreover, any batch larger than $K-m$ gets rejected. Thus, when the system contains $m \leq n$ jobs, the rate at which level n is crossed from below is

$$\lambda\pi(m)\mathsf{P}(n+1-m \leq B \leq K-m) = \lambda\pi(m)[G(n-m)-G(K-m)],$$

where we use that

$$\begin{aligned}\mathsf{P}(n+1-m \leq B \leq K-m) &= \mathsf{P}(B \leq K-m) - \mathsf{P}(B \leq n-m)\\ &= \mathsf{P}(B > n-m) - \mathsf{P}(B > K-m)\\ &= G(n-m)-G(K-m).\end{aligned}$$

Since the server serves only single items, level n can only be crossed from above from state $n+1$. This happens at rate $\mu p(n+1)$. With PASTA this is equal to $\mu\pi(n+1)$.

Finally, since in the long run the number of up- and down-crossings must be the same, the up- and down-crossing rates must match. This implies that the balance equations becomes

$$\mu\pi(n+1) = \lambda \sum_{m=0}^{n} \pi(m)[G(n-m)-G(K-m)],$$

for $n = 0,\ldots,K-1$.

Before we continue with the other acceptance rules, it is important to check this result. In general, off-by-one errors are easily, and commonly, made, so we need to test the above on simple cases.

- If $K \to \infty$, then $G(K-m) = \mathsf{P}(B > K-m) \to 0$, so we get our earlier result.
- Take $n = K$. Then $G(n-m) - G(K-m) = 0$ for all m. Then the right-hand side is 0, as it should.
- Taken $n = 0$ and $K = 1$, then $\mu\pi(1) = \lambda\pi(0)$. This also makes sense.
- Take n much smaller than K. If the batch size is maximally 2, then for small n the entire batch must fit. Let's see if this holds in the above formula. If n much smaller than K, then also m is much smaller than K (since in the right-hand size, $m \le n$). But then $G(K-m) \le G(2) = 0$, as it should. (Observe that G is a decreasing function of its argument; it's a survival function.)

s.2.11.13. The complete-acceptance policy is actually quite simple. As any batch will be accepted when $n \le K$, the queue length is not bounded. Only when the number of jobs in the system is larger than K, we do not accept jobs.

$$\mu\pi(n+1) = \begin{cases} \lambda \sum_{m=0}^{n} \pi(m) G(n-m), & \text{for } n \le K, \\ \lambda \sum_{m=0}^{K} \pi(m) G(n-m), & \text{for } n > K. \end{cases}$$

s.2.11.14. For the partial acceptance case, any job is accepted, but the system only admits whatever fits. As level $n \in 0, 1, ..., K-1$ is still up-crossed by any batch of size at least $n-m$ when the system is in state m, the formula for the up-crossing rate is identical to the case without this acceptance policy. Moreover, nothing changes to the formula for the down-crossing rate. Hence,

$$\mu\pi(n+1) = \lambda \sum_{m=0}^{n} \pi(m) G(n-m),$$

for $n = 0, 1, \ldots, K-1$.

2.12 $M/G/1$ QUEUE LENGTH DISTRIBUTION

Theory and Exercises

In Section 2.11 we used level-crossing arguments to find a recursive method to compute the stationary distribution $p(n)$ of the number of items in an $M^X/M/1$ queue. Here we apply similar arguments to find $p(n) = \mathsf{P}(L = n)$ for the $M/G/1$ queue. However, we cannot simply copy the derivation of the $M^X/M/1$ queue to the $M/G/1$ queue, because in the $M^X/M/1$ queue the service times of the items are exponential, hence memoryless, while in the $M/G/1$ this is not the case.

When job service times are not memoryless, hence do not restart at arrival times, we cannot choose any moment we like to apply level-crossing. Thus, for the $M/G/1$ queue we need to focus on moments in time in which the system 'restarts'. As we will see below, the appropriate moments are job departure epochs. All in all, the argumentation to find the recursion for $\{p(n)\}$ is quite subtle, as it uses an interplay of the PASTA property and (2.7.7) between $\pi(n)$, $p(n)$ and $\delta(n)$.

An important role below is played by the number of arrivals Y_k during the service time of the kth job. Since the service times of the jobs form a sequence of i.i.d. random variables, the elements of the sequence $\{Y_k\}$ are also i.i.d. Let Y be the common random variable with probability mass $f(j) = \mathsf{P}(Y = j)$; write $G(j) = \mathsf{P}(Y_k > j)$ for the survivor function.

2.12.1. *Explain that if the service time is constant and equal to s, then*

$$\mathsf{P}(Y_k = j \,|\, S = s) = e^{-\lambda s}\frac{(\lambda s)^j}{j!}. \tag{2.12.1}$$

2.12.2. *Explain that*

$$\mathsf{P}(Y_k = j) = \int_0^\infty e^{-\lambda x}\frac{(\lambda x)^j}{j!}\, dF(x), \tag{2.12.2}$$

where F is the distribution of the service times.

2.12.3. *If S is deterministic and equal to s, show that* (2.12.2) *reduces to* (2.12.1).

2.12.4. *If $S \sim \mathrm{Exp}(\mu)$, show that*

$$f(j) = \mathsf{P}(Y_k = j) = \frac{\mu}{\lambda+\mu}\left(\frac{\lambda}{\lambda+\mu}\right)^j. \tag{2.12.3}$$

2.12.5. *If $S \sim \mathrm{Exp}(\mu)$, show that*

$$G(j) = \sum_{k=j+1}^{\infty} f(k) = \left(\frac{\lambda}{\lambda+\mu}\right)^{j+1}. \tag{2.12.4}$$

2.12.6. *Design a suitable numerical method to evaluate* (2.12.2) *for more general distribution functions F.*

Let us concentrate on a down-crossing of level n, see Fig. 15; recall that level n lies between states n and $n+1$. For job k to generate a down-crossing of level n, two events must take place: job '$k-1$' must leave $n+1$ jobs behind after its service completion, and job k must leave n jobs behind. Thus,

$$\text{Down-crossing of level } n \iff \mathbb{1}_{L(D_{k-1})=n+1}\,\mathbb{1}_{L(D_k)=n} = 1.$$

Let us write this in another way. Observe that if $L(D_{k-1}) = n+1$ and no other jobs arrive during the service time S_k of job k, i.e., when $Y_k = 0$, it must also be that job k leaves n jobs behind. If, however, $Y_k > 0$, then $L(D_k) \geq n+1$. Thus, we see that

$$\text{Down-crossing of level } n \iff \mathbb{1}_{L(D_{k-1})=n+1}\,\mathbb{1}_{Y_k=0} = 1.$$

Consequently, the number of down-crossings of level n up to time t is

$$D(n+1,0,t) = \sum_{k=1}^{D(t)} \mathbb{1}_{L(D_{k-1})=n+1}\,\mathbb{1}_{Y_k=0}.$$

2.12.7. *Show that*

$$\lim_{t\to\infty}\frac{D(n+1,0,t)}{t} = \delta\delta(n+1)f(0),$$

where $f(0) = \mathsf{P}(Y = 0)$.

Before we deal with the up-crossing, it is important to do the next exercise.

2.12.8. *Suppose that $L(D_{k-1}) > 0$. Why is $D_k = D_{k-1} + S_k$? However, if $L(D_{k-1}) = 0$, the time between D_{k-1} and D_k is* not *equal to S_k. Why not? Can you find an expression for the distribution of $D_k - D_{k-1}$ in case $L(D_{k-1}) = 0$?*

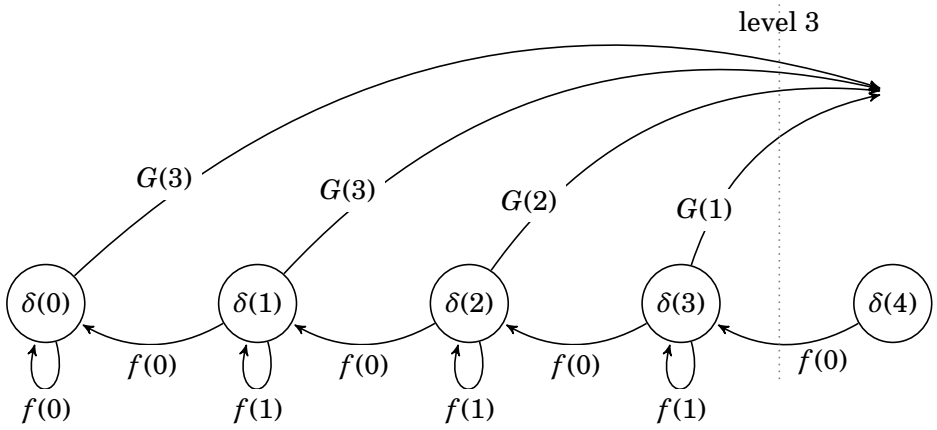


Figure 15: Level 3 is crossed from below with rate $\delta\delta(0)G(3) + \delta\delta(1)G(3) + \cdots \delta\delta(3)G(1)$ and crossed from above with rate $\delta\delta(4)f(0)$.

For the up-crossings, assume first that $L(D_{k-1}) = n > 0$. Then an up-crossing of level $n > 0$ must have occurred when $L(D_k) > n$, i.e.,

$$\mathbb{1}_{L(D_{k-1})=n}\,\mathbb{1}_{L(D_k)>n} = 1 \implies \text{Up-crossing of level } n.$$

Again, we can convert this into a statement about the number of arrivals Y_k that occurred during the service time S_k of job k. If $Y_k = 0$, then job k must leave $n-1$ jobs behind, so no up-crossing can happen. Next, if $Y_k = 1$, then job k leaves n jobs behind, so still no up-crossing occurs. In fact, level n can only be up-crossed from level n if more than one job arrives during the service of job k, i.e.,

$$\mathbb{1}_{L(D_{k-1})=n}\,\mathbb{1}_{Y_k>1} = 1 \implies \text{Up-crossing of level } n.$$

More generally, level n is up-crossed from level m, $0 < m \le n$ whenever

$$\mathbb{1}_{L(D_{k-1})=m}\,\mathbb{1}_{Y_k>n-m+1} = 1 \implies \text{Up-crossing of level } n.$$

However, if $m = 0$ (think about this),

$$\mathbb{1}_{L(D_k)>n} = \mathbb{1}_{L(D_{k-1})=0}\,\mathbb{1}_{Y_k>n} \implies \text{Up-crossing of level } n.$$

Again we define proper counting functions, divide by t, and take suitable limits to find for the up-crossing rate

$$\delta\delta(0)G(n) + \delta \sum_{m=1}^{n} \delta(m)G(n-m+1). \tag{2.12.5}$$

Equating the down-crossing and up-crossing rates and dividing by δ gives

$$f(0)\delta(n+1) = \delta(0)G(n) + \sum_{m=1}^{n} \delta(m)G(n+1-m).$$

Noting that $\pi(n) = \delta(n)$, as this holds for any rate-stable $G/G/1$ queue, cf., (2.7.7), hence in particular for the $M/G/1$ queue length process, we arrive at

$$f(0)\pi(n+1) = \pi(0)G(n) + \sum_{m=1}^{n} \pi(m)G(n+1-m). \tag{2.12.6}$$

Clearly, we have again obtained a recursion with which we can compute the state probabilities.

2.12.9. *Provide the details behind the derivation of* (2.12.5).

2.12.10. *Clearly, the $M/M/1$ queue is a special case of the $M/G/1$ queue. Check that the queue length distribution of the $M/M/1$ queue satisfies* (2.12.6).

Hints

h.2.12.1. If s is deterministic, the number of arrivals during a fixed period of time with length s must be Poisson distributed.

h.2.12.2. Use 2.12.1.

h.2.12.3. $S = s$ then $\mathsf{P}(S = s) = 1$, i.e., all probability mass lies at s. Thus, all arrivals must occur during $[0, s]$.

h.2.12.4. Use the ideas of 1.4.3 to simplify the standard integral.

h.2.12.5. Use 2.12.4.

h.2.12.6. Discretize time to a grid of points, and approximate the integral by a summation over the grid.

h.2.12.7. Use a similar derivation as in (2.11.1).

h.2.12.8. You might find some inspiration in 2.7.11.

Realize that if $L(D_{k-1}) = 0$, job $k-1$ leaves behind an empty system. Thus, before job k can leave, it has to arrive. In other words, $D_{k-1} < A_k$. Since job k arrives to an empty system, his service starts right away, to that the time between A_k and D_k is equal to the service time of job k.

h.2.12.9. Define for $m = 1, \ldots, n$

$$D(m, n, t) = \sum_{k=1}^{D(t)} \mathbb{1}_{L(D_{k-1})=m} \mathbb{1}_{Y_k > n-m+1},$$

and

$$D(0, n, t) = \sum_{k=1}^{D(t)} \mathbb{1}_{L(D_{k-1})=0} \mathbb{1}_{Y_k > n}.$$

Then, divide by $D(n, t)$ and $D(t)$ and take limits.

h.2.12.10. To simplify the computations, define shorthands such as $\alpha = \lambda/(\lambda+\mu)$, so that $1-\alpha = \mu/(\lambda+\mu)$, and $\alpha/(1-\alpha) = \lambda/\mu = \rho$. Then, with (2.12.3), $f(n) = \alpha^n(1-\alpha)$ and $G(n) = \alpha^{n+1}$. (These results do not 'come for free'. Of course, it's just algebra, but please try to derive this yourself. It's good to hone your computational skills.)

Solutions

s.2.12.1. See the hint. The period during which the arrivals occur is s.

s.2.12.2. We use a conditioning argument to arrive at this result. The probability that the service time is x units long is written in various ways in the literature: $F(\mathrm{d}x) = \mathrm{d}F(x) = \mathsf{P}(S \in \mathrm{d}x)$, but this all means the same thing, it is only the notation that differs. (As an aside, to properly define this we need measure theory). When F has a density f, then $\mathrm{d}F(x) = f(x)\mathrm{d}x$. Note that when S is discrete, it does not have a density everywhere. With this,

$$\mathsf{P}(Y_k = j) = \int_0^\infty \mathsf{P}(Y_k = j \,|\, S = x)\mathsf{P}(S \in \mathrm{d}x) = \int_0^\infty \mathsf{P}(Y_k = j \,|\, S = x)\,\mathrm{d}F(x).$$

Using the answer of the previous problem we arrive at the result.

s.2.12.3. Let us first attack this problem from a general point of view. Suppose the service time S can take values $s_1 < s_2 < \cdots < s_n$, and $\mathsf{P}(S = s_i) = \alpha_i$. Then of course we want that $\mathsf{P}(S \le s_n) = \sum_{i=1}^n \alpha_i = 1$. The distribution function F of S is in this case a step function, with steps at the points $s_1, s_2, \ldots$, and step sizes $\alpha_1, \alpha_2, \ldots$. Thus, $F(s_i) - F(s_i-) = \alpha_i$. We say that such a distribution function has *atoms* at the points $s_1, s_2, \ldots$. In this case we write

$$\int_0^\infty g(x)\,\mathrm{d}F(x) = \sum_{i=1}^n g(s_i)\alpha_i.$$

Thus, the integral of g with respect to F is the sum of g at the points at which F makes a jump times the weight of F at these points.

As an example, in case $S \equiv 10$ (the service time is always 10 time units long) the distribution function F makes just one jump at $s_1 = 10$ of size $\alpha_1 = F(10) - F(10-) = 1$, i.e., F has the form

$$F(x) = \begin{cases} 0, & x < 10, \\ 1, & x \ge 10. \end{cases}$$

With this,

$$\int_0^\infty g(x)\,\mathrm{d}F(x) = \sum_{i=1}^n g(s_i)\alpha_i = g(s_1)\alpha_1 = g(10)\cdot 1 = g(10).$$

Thus, the integral of g with respect to this distribution F is $g(10)$. More generally, when F puts all probability mass at the single point s (rather than at the point 10), then

$$\int_0^\infty g(x)\,\mathrm{d}F(x) = g(s).$$

Let us now copy the formula of the previous problem:

$$\mathsf{P}(Y_k = j) = \int_0^\infty \mathsf{P}(Y_k = j \,|\, S = x)\,\mathrm{d}F(x).$$

We see that here the integrand is the function $g(x) = \mathsf{P}(Y_k = j \,|\, S = x)$. It is given in the question that F puts all mass on the point s. Therefore,

$$\mathsf{P}(Y_k = j) = \int_0^\infty \mathsf{P}(Y_k = j \,|\, S = x)\,\mathrm{d}F(x) = \int_0^\infty g(x)\,\mathrm{d}F(x) = g(s). \qquad \star$$

Now we also know that if the service time takes precisely x time units, the number of arrivals is Poisson distributed. Therefore

$$\mathsf{P}(Y_k = j \mid S = x) = e^{-\lambda x}\frac{(\lambda x)^j}{j!}.$$

Hence,

$$g(x) = \mathsf{P}(Y_k = j \mid S = x) = e^{-\lambda x}\frac{(\lambda x)^j}{j!}.$$

Finally, using $(\star)$, and taking $x = s$ in the above expression of g, we get

$$\mathsf{P}(Y_k = j) = \int_0^\infty \mathsf{P}(Y_k = j \mid S = x)\,\mathrm{d}F(x) = g(s) = e^{-\lambda s}\frac{(\lambda s)^j}{j!}.$$

s.2.12.4. Use conditional probability to see that

$$\begin{aligned}
\mathsf{P}(Y_n = j) &= \int_0^\infty e^{-\lambda x}\frac{(\lambda x)^j}{j!}\,\mathrm{d}F(x) = \int_0^\infty e^{-\lambda x}\frac{(\lambda x)^j}{j!}\mu e^{-\mu x}\,\mathrm{d}x \\
&= \frac{\mu}{j!}\lambda^j\int_0^\infty e^{-(\lambda+\mu)x}x^j\,\mathrm{d}x = \frac{\mu}{j!}\left(\frac{\lambda}{\lambda+\mu}\right)^j\int_0^\infty e^{-(\lambda+\mu)x}((\lambda+\mu)x)^j\,\mathrm{d}x \\
&= \frac{\mu}{j!}\left(\frac{\lambda}{\lambda+\mu}\right)^j\frac{j!}{\lambda+\mu}.
\end{aligned}$$

In the last integral, we use the hint. Specifically,

$$\begin{aligned}
\int_0^\infty e^{-\alpha x}(\alpha x)^j\,\mathrm{d}x &= \alpha^{-1}\int_0^\infty e^{-x}x^j\,\mathrm{d}x = \alpha^{-1}\left(-e^{-x}x^j|_0^\infty + j\int_0^\infty e^{-x}x^{j-1}\,\mathrm{d}x\right) \\
&= \alpha^{-1}j\int_0^\infty e^{-x}x^{j-1}\,\mathrm{d}x = \alpha^{-1}j(j-1)\int_0^\infty e^{-x}x^{j-2}\,\mathrm{d}x \\
&= \alpha^{-1}j!\int_0^\infty e^{-x}\,\mathrm{d}x = j!/\alpha.
\end{aligned}$$

In hindsight, this result could have been derived in another way, in fact by using the result of 1.4.16 in which we analyzed a merged Poisson process. Consider the Poisson process with rate $\lambda+\mu$ that arises when the arrival and service process are merged. The probability that an arrival corresponds to an epoch of the merged process is $\lambda/(\lambda+\mu)$ and the probability that a departure corresponds to an epoch of the merged process is $\mu/(\lambda+\mu)$. The probability that j arrivals occur before a service occurs, is the same as the probability that a geometrically distributed random variable with success probability $\mu/(\lambda+\mu) = 1-p$ takes the value j.

s.2.12.5. Take $\alpha = \lambda/(\lambda+\mu)$ so that $f(j) = (1-\alpha)\alpha^j$.

$$\begin{aligned}
G(j) &= \sum_{k=j+1}^\infty f(k) = (1-\alpha)\sum_{k=j+1}^\infty \alpha^k \\
&= (1-\alpha)\sum_{k=0}^\infty \alpha^{k+j+1}, \text{ by change of variable} \\
&= (1-\alpha)\sum_{k=0}^\infty \alpha^k\alpha^{j+1} = (1-\alpha)\alpha^{j+1}\sum_{k=0}^\infty \alpha^k \\
&= (1-\alpha)\alpha^{j+1}\frac{1}{1-\alpha} = \alpha^{j+1}.
\end{aligned}$$

s.2.12.6. A simple numerical method is as follows. Make a grid of size $\mathrm{d}x$, for some small number $\mathrm{d}x$, e.g. $\mathrm{d}x = 1/100$, and write $f_i = \mathsf{P}(S \in (i\,\mathrm{d}x, (i+1)\mathrm{d}x]) = F((i+1)\mathrm{d}x) - F(i\,\mathrm{d}x)$. Then

$$\mathsf{P}(Y_k = j) = \int_0^\infty e^{-\lambda x} \frac{(\lambda x)^j}{j!}\,\mathrm{d}F(x) \approx \sum_{i=1}^{\infty} e^{-\lambda i\,\mathrm{d}x} \frac{(\lambda i\,\mathrm{d}x)^j}{j!} f_i\,\mathrm{d}x.$$

Let's try a numerical experiment.

```
>>> import numpy as np

>>> labda = 3
>>> mu = 4
>>> j = 5
>>> dx = 1 / 100


>>> def F(x):
...  return 1 - np.exp(-mu * x)
...

>>> def f(x):
...  return F(x + dx) - F(x)
...

>>> def term(i):
...  res = np.exp(-labda * i * dx)
...  res *= (labda * i * dx)**j / np.math.factorial(j)
...  res *= f(i*dx) * dx
...  return res
...

>>> print(sum(term(i) for i in range(50)))
1.1158835007226524e-05
>>> print(sum(term(i) for i in range(500)))
8.098807710055904e-05
>>> print(sum(term(i) for i in range(5000)))
8.098807712728956e-05
```

Since I don't know when to stop the integral I just try a few values; of course stopping the integration at $x = 50$ is too small, since $50\,\mathrm{d}x = 50/100 = 1/2$, but I include it for illustrative purposes. Stopping at 500 seems OK, since the results of the last two integrals are nearly the same. This also suggests that $\mathrm{d}x = 1/100$ is sufficiently small. In general, however, one must take care and try various values for $\mathrm{d}x$ and the integration limits.

For more complicated situations it is best to use a numerical library to compute the above integral. These methods have been designed to produce good and reliable results, and, typically, it is very hard to improve these methods. Thus, let's try a real number cruncher.

```
>>> from scipy.integrate import quad
```

```
>>> def g(x):
...  return np.exp(-labda*x) * (labda*x)**j/np.math.factorial(j) * f(x)
...
>>> print(quad(g, 0, np.inf))
(8.098807712760667e-05, 3.4086429822163874e-11)
```

This is the same as our earlier answer.

s.2.12.7. By using the definitions and limits developed in Sections 2.1 and 2.7,

$$\begin{aligned}\lim_{t\to\infty}\frac{D(n+1,0,t)}{t} &= \lim_{t\to\infty}\frac{D(t)}{t}\frac{D(n+1,t)}{D(t)}\frac{D(n+1,0,t)}{D(n+1,t)}\\ &= \delta\delta(n+1)\lim_{t\to\infty}\frac{D(n+1,0,t)}{D(n+1,t)}\\ &= \delta\delta(n+1)\mathsf{P}(Y=0)\\ &= \delta\delta(n+1)f(0),\end{aligned}$$

where the last limit follows from the independence of Y_k and $L(D_{k-1})$.

s.2.12.8. When $L(D_{k-1})>0$, job k is already in the system when job $k-1$ finishes its service and leaves. Thus, at the departure time D_{k-1} of job $k-1$, the service of job k can start right away at D_{k-1}. Then, $D_k = D_{k-1}+S_k$.

When job $k-1$ leaves an empty system behind, $D_k = A_k + S_k$, since job k sees an empty system, hence its service can start right away after its arrival time at time A_k. Since the arrival process is Poisson by assumption, the time to the next arrival after D_{k-1} is exponentially distributed with rate λ. (Recall the memoryless property of the inter-arrival times.) Thus, $A_k - D_{k-1}$ has the same distribution as X_k, so that $\mathsf{P}(D_k - D_{k-1}\le x) = \mathsf{P}(X_k + S_k \le x)$.

s.2.12.9. With the definition of the hint,

$$\begin{aligned}\lim_{t\to\infty}\frac{D(0,n,t)}{t} &= \lim_{t\to\infty}\frac{D(t)}{t}\frac{D(0,t)}{D(t)}\frac{D(0,n,t)}{D(0,t)}\\ &= \delta\delta(0)\lim_{t\to\infty}\frac{D(0,n,t)}{D(0,t)}\\ &= \delta\delta(0)\mathsf{P}(Y>n)\\ &= \delta\delta(0)G(n).\end{aligned}$$

$$\begin{aligned}\lim_{t\to\infty}\frac{D(m,n,t)}{t} &= \lim_{t\to\infty}\frac{D(t)}{t}\frac{D(m,t)}{D(t)}\frac{D(m,n,t)}{D(m,t)}\\ &= \delta\delta(m)\lim_{t\to\infty}\frac{D(m,n,t)}{D(m,t)}\\ &= \delta\delta(m)\mathsf{P}(Y>n-m+1)\\ &= \delta\delta(m)G(n-m+1).\end{aligned}$$

s.2.12.10. Use the shorthands of the hint. We work in steps. First we check whether $\pi(0)=1-\rho$ of the $M/M/1$ queue satisfies (2.12.6), then we check $\pi(1)$, and finally we check the general case with $n\ge 1$.

Take $n = 0$, so that $f(0) = 1 - \alpha$. Hence, the left-hand side becomes $f(0)\pi(1) = (1-\alpha)\pi(1)$. The right-hand side is $\pi(0)G(0) = \pi(0)\alpha$. Thus, $\pi(1) = \pi(0)\alpha/(1-\alpha) = \rho\pi(0)$. And indeed, for the $M/M/1$ queue we know that this is true.

Now we check it for $n = 1$,

$$\begin{aligned}(1-\alpha)\pi(2) &= \pi(0)G(1) + \pi(1)G(1) = \pi(0)G(1)(1+\rho)\\ &= \pi(0)\alpha^2(1+\rho) = \pi(0)\alpha\alpha(1+\rho) = \pi(0)\alpha\rho.\end{aligned}$$

Dividing by $1-\alpha$, we get

$$\pi(2) = \pi(0)\rho^2.$$

Finally, we fill in $\pi(n) = \rho^n\pi(0)$ of the $M/M/1$ queue for $n \geq 1$. For ease, we divide both sides by $\pi(0)$ first. We are left with checking that

$$\begin{aligned}(1-\alpha)\rho^{n+1} &= \alpha^{n+1} + \sum_{m=1}^{n} \rho^m \alpha^{n-m+2}\\ &= \alpha^{n+1} + \alpha^{n+2}\sum_{m=1}^{n}(\rho/\alpha)^m\\ &= \alpha^{n+1} + \alpha^{n+1}\rho\sum_{m=0}^{n-1}(\rho/\alpha)^m\\ &= \alpha^{n+1} + \alpha^{n+1}\rho\frac{1-(\rho/\alpha)^n}{1-\rho/\alpha}\\ &= \alpha^{n+1} - \alpha^{n+1}(1-(\rho/\alpha)^n), \quad \text{as } \rho/\alpha = 1+\rho,\\ &= \alpha^{n+1}(\rho/\alpha)^n = \alpha\rho^n.\end{aligned}$$

Since $\rho = \alpha/(1-\alpha)$ we see that the left- and right-hand sides are the same.

Thus we get that, by using PASTA, $p(n) = \pi(n) = \rho^n\pi(0) = \rho^n(1-\rho)$; a result we obtained earlier for the $M/M/1$ queue.

2.13 GRAPHICAL SUMMARIES

We finish this chapter with providing two summaries in graphical form to clarify how all concepts developed in this chapter relate.

2.14 OLD EXAM QUESTIONS

2.14.1 *Multiple-choice Questions*

2.14.1. *[201703] Let $L(s)$ be the number of items in the system at time s. Define*

$$\alpha = \lim_{n\to\infty}\frac{1}{n}\sum_{k=1}^{n} L(A_k).$$

For the $M/G/1$ queue we can use PASTA to see that $1-\rho = \alpha$.

2.14.2. *[201703] For the $G/G/1$ queue,*

$$\frac{D(n-1,t)}{t} = \frac{D(n-1,t)}{Y(n,t)}\frac{Y(n,t)}{t} \to \mu(n)p(n), \tag{2.14.1}$$

if $n \geq 1$.

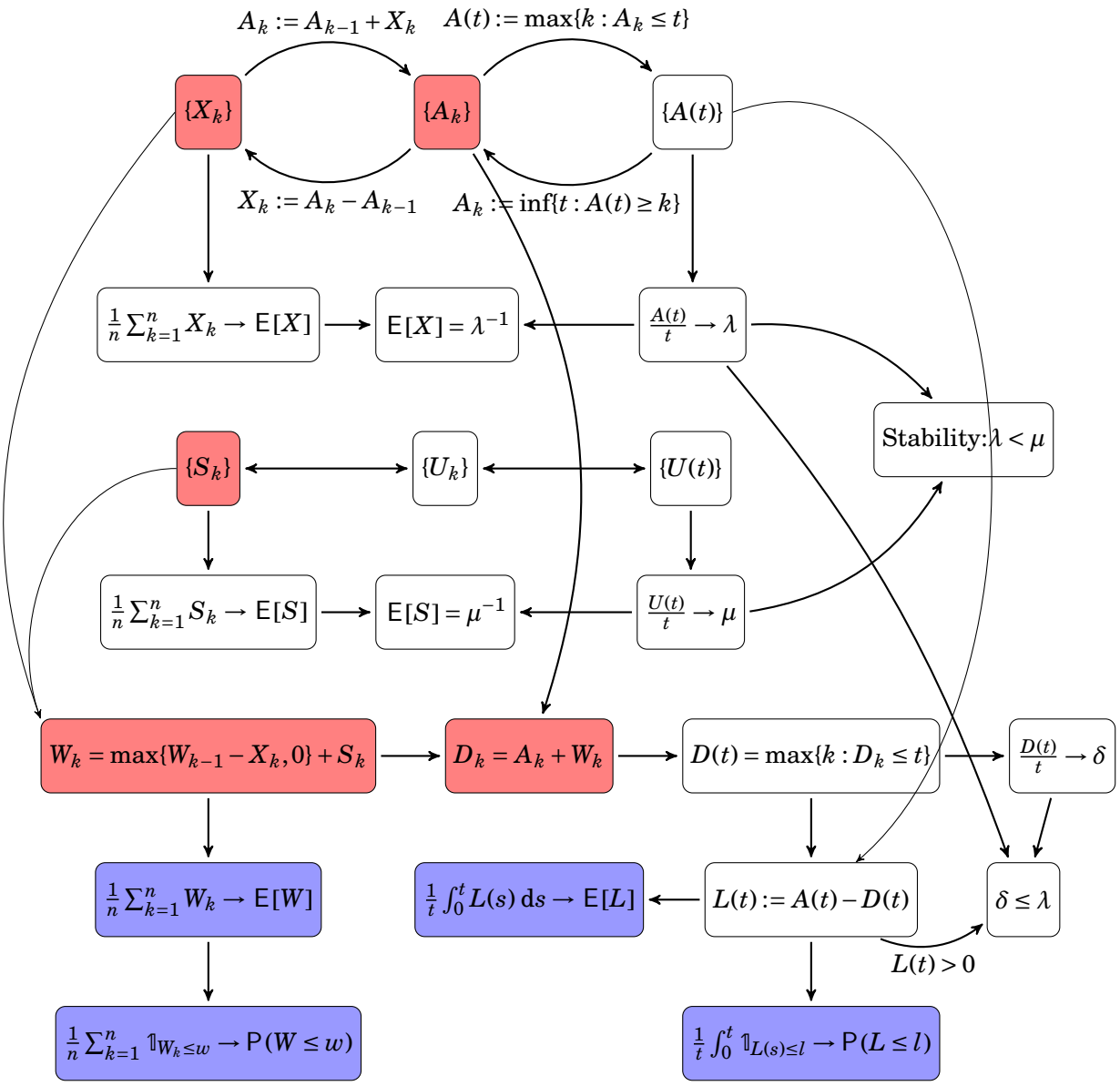


Figure 16: Here we sketch the relations between the construction of the $G/G/1$ queue from the primary data, i.e., the inter-arrival times $\{X_k; k \ge 0\}$ and the service times $\{S_k; k \ge 0\}$, and different performance measures.

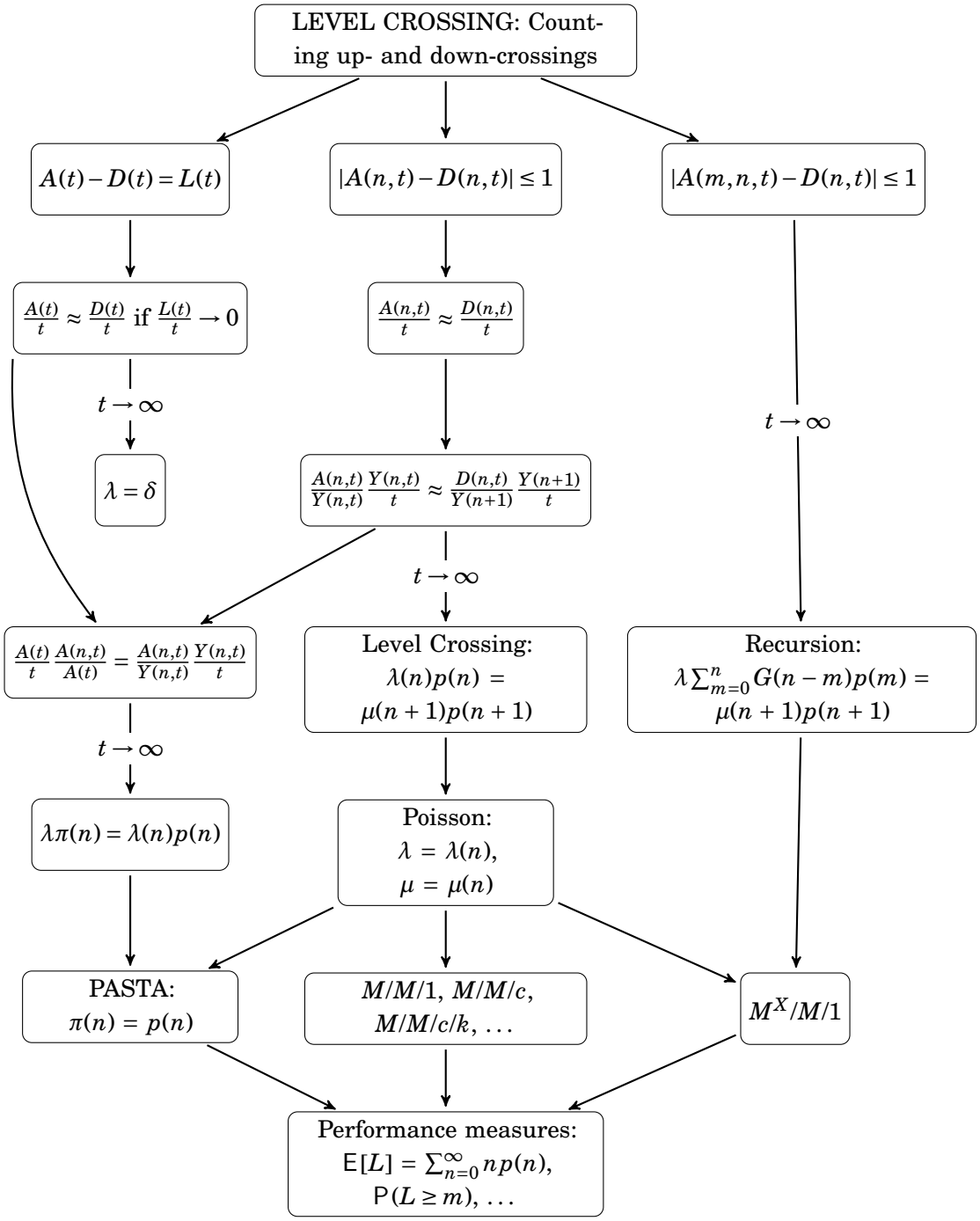


Figure 17: With level-crossing arguments we can derive a number of useful relations. This figure presents an overview of these relations that we derive in this and the next sections.

2.14.3. *[201703] For the $M/M/1$ queue with $\lambda = 3$, $\mu = 5$, $\mathsf{E}\left[L_Q\right] \leq 1$*

2.14.4. *[201703]*

$$\begin{aligned}\sum_{n=0}^{\infty} n^2\rho^n &= \sum_{n=0}^{\infty}\left(\sum_{i=1}^{\infty} 2i\,\mathbb{1}_{i\leq n} - n\right)\rho^n = \sum_{n=0}^{\infty}\sum_{i=0}^{\infty} 2i\,\mathbb{1}_{i\leq n}\rho^n - \sum_{n=0}^{\infty} n\rho^n \\ &= \sum_{i=0}^{\infty} 2i\sum_{n=i}^{\infty}\rho^n - \frac{\mathsf{E}[L]}{1-\rho} = \sum_{i=0}^{\infty} 2i\rho^i\sum_{n=0}^{\infty}\rho^n - \frac{\mathsf{E}[L]}{1-\rho} \\ &= \frac{2}{1-\rho}\sum_{i=0}^{\infty} i\rho^i - \frac{\mathsf{E}[L]}{1-\rho} = \frac{2}{(1-\rho)^2}\mathsf{E}[L] - \frac{\mathsf{E}[L]}{1-\rho} \\ &= \frac{\mathsf{E}[L]}{1-\rho}\left(\frac{2}{1-\rho} - 1\right) = \frac{\mathsf{E}[L]}{1-\rho}\frac{1+\rho}{1-\rho} \\ &= \frac{\rho}{1-\rho}\frac{1+\rho}{(1-\rho)^2}.\end{aligned}$$

2.14.5. *[201703] For the $M/G/1$ queue with $G = U[0,A]$, i.e., the uniform distribution on $[0,A]$:*

$$C_s^2 = \frac{1}{3}.$$

2.14.6. *[201704]Consider the following queueing process. At times $0,2,4,\ldots$ customers arrive, each customer requires 1 unit of service, and there is one server. Then, for $t \in [0,3)$*

$$Y(1,t) = \int_0^t \mathbb{1}_{L(s)=1}\,ds = \begin{cases} t & t \in [0,1), \\ 1 & t \in [1,2), \\ 1+(t-2) & t \in [2,3), \end{cases}$$

2.14.7. *[201704]For the $M/M/c$ queue with $\rho = c\lambda/\mu$,*

$$p(n+1) = \frac{\Pi_{k=1}^{n+1}\min\{c,k\}^{n+1}}{\rho}\quad p(0).$$

2.14.8. *[201704]Consider a queueing system in which each customer requires precisely 59 minutes of service. At the start of each hour, one customer arrives. Then $\pi(0) = 1/60$.*

2.14.9. *[201704]For the $G/G/1$ queue, when T is a moment in time in which the system is empty, we have*

$$\begin{aligned}\int_0^T L(s)\,ds &= \int_0^T \sum_{k=1}^{A(T)} 1\{A_k \leq s < D_k\}\,ds \\ &= \sum_{k=1}^{A(T)}\int_0^T 1\{A_k \leq s < D_k\}\,ds = \sum_{k=1}^{A(T)} W_k.\end{aligned}$$

In words, the area between the graphs of $A(s)$ and $D(s)$ must be equal to the total waiting time spent by all jobs in the system until T.

2.14.10. *[201704]For the number of jobs in the system L for the $M/G/1$ queue we have that*

$$\phi(z) = \mathsf{E}\left[z^L\right] = \sum_{n=0}^{\infty} z^n p(n) = (1-\rho)\sum_{n=0}^{\infty}(\rho z)^n = \frac{1-\rho}{1-\rho z}.$$

Then

$$\mathsf{E}[L] = \frac{d}{dz}\phi(z)\bigg|_{z=0}.$$

2.14.11. *[201704]For the $M^X/M/1$ queue,*

$$\frac{\lambda}{\mu}\mathsf{E}\left[B^2\right] = \rho\frac{\mathsf{V}[B]}{(\mathsf{E}[B])^2}\mathsf{E}[B] = \rho C_s^2\mathsf{E}[B].$$

2.14.12. *[201704]For the $M^X/M/1$ queue with $G(n) = \mathsf{P}(B > n)$,*

$$\begin{aligned}\sum_{n=0}^{\infty} G(n) &= \sum_{n=0}^{\infty}\mathsf{P}(B>n) = \sum_{n=0}^{\infty}\sum_{i=n+1}^{\infty}\mathsf{P}(B=i)\\ &= \sum_{n=0}^{\infty}\sum_{i=0}^{\infty}1\{n<i\}\mathsf{P}(B=i) = \sum_{i=0}^{\infty}\sum_{n=0}^{\infty}1\{n<i\}\mathsf{P}(B=i)\\ &= \sum_{i=0}^{\infty} i\mathsf{P}(B=i) = \mathsf{E}[B].\end{aligned}$$

2.14.13. *[201704]We model a workstation with just one machine as a $G/G/1$ queue. The coefficient of variation of the interarrival times of the jobs is 1 and the arrival rate $\lambda = 3/8$ per hour. The average service time $\mathsf{E}[S] = 2$ hours, $C_s^2 = 1/2$. Then, $\mathsf{E}\left[W_Q\right] \in [4,5]$ hours.*

2.14.14. *[201802] Let A_1 be the arrival time of the first job at a queueing system. Assume $L(A_1-) = 0$. Suppose that the $n+1$th job is the first job after A_1 that sees an empty system. Thus, $L(D_n) = 0$. The fraction of time that the server has been busy during $[A_1, A_{n+1})$ is*

$$\frac{\sum_{i=1}^{n} S_i}{A_{n+1} - A_1}$$

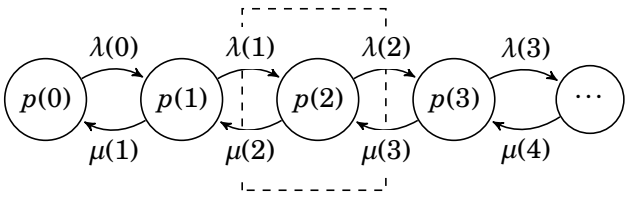
2.14.15. *[201802] The number of arrivals $A(t)$ up to time t is equal to* $\sup\{k : A_k \le t\}$.

2.14.16. *[201802] In a discrete-time queueing model, L_k is the number of jobs in the system at the end of period k. Thus, $\sum_{k=1}^{n} \mathbb{1}_{L_k > m}$ is the number of jobs that see more than m jobs in the system upon arrival.*

2.14.17. *[201802] In a level-crossing analysis of a queueing system, the departure rate from state n is*

$$\mu(n) = \lim_{t\to\infty}\frac{D(n-1,t)}{Y(n,t)},$$

2.14.18. *[201802] To obtain the* balance equations *we do not count the number of up- and down crossings of a level. Instead we count how often a box around a state, such as state 2 in the figure below, is crossed from inside and outside.*



2.14.19. *[201802] The process $L(t)$ that counts the number of jobs in system is* right continuous.

2.14.20. *[201803] If the limit exists,*

$$\frac{1}{t}\sum_{k=1}^{A(t)} \mathbb{1}_{W_k \le x} \to \mathsf{P}(W \le w),$$

as $t \to \infty$.

2.14.21. *[201803] Using the definitions of the book, for the $M/M/1$ queue and $t > 0$,*

$$\left| \frac{A(n,t)}{Y(n,t)} \frac{Y(n,t)}{t} - \frac{D(n,t)}{Y(n+1,t)} \frac{Y(n+1,t)}{t} \right| \leq 1.$$

2.14.22. *[201803] For the $M/M/1$ queue, the following reasoning leads to the expected number of jobs in the system.*

$$\begin{aligned} M_L(s) = \mathsf{E}\left[e^{sL}\right] &= \sum_{n=0}^{\infty} e^{sn} p(n) = (1-\rho) \sum_{n} e^{sn} \rho^n \\ &= \frac{1-\rho}{1-e^s \rho}, \end{aligned}$$

where we assume that s is such that $e^s \rho < 1$. Then,

$$M_L'(s) = (1-\rho) \frac{1}{(1-e^s \rho)^2} e^s \rho.$$

Hence, $\mathsf{E}[L] = M_L'(0) = \rho/(1-\rho)$.

2.14.23. *[201803] Customers of fast-food restaurants prefer to be served from stock. For this reason such restaurants often use a 'produce-up-to' policy: When the on-hand inventory I is equal or lower than some threshold $S-1$, the company produces items until the inventory level equals S again. The level S is known as the order-up-to level, and $S-1$ as the reorder level.*

Suppose that customers arrive as a Poisson process with rate λ and the production times of single items are i.i.d. and exponentially distributed with parameter μ. Assume also that customers who cannot be served from on-hand stock are backlogged, that is, they wait until their item has been produced.

The average on-hand inventory level is S minus the average number of jobs at the cook. i.e., $\mathsf{E}[I] = \sum_{i=0}^{S} (S-i) p(i)$.

2.14.24. *[201803] A queueing system with balking, at level b say, behaves the same as a queueing system with finite calling population of size b.*

2.14.25. *[201803] Consider the $M/G/1$ queue. By the PASTA property, a fraction ρ, $\rho < 1$, of the arrivals sees the server occupied, while a fraction $1-\rho$ sees a free server.*

2.14.26. *[201803] For the $M/G/1$ queue we can use the PASTA property to see that the expected waiting time in the system is equal to $\mathsf{E}[W] = \sum_{n=0}^{\infty} \mathsf{E}\left[W_Q \,\middle|\, N = n\right] \pi(n) + \mathsf{E}[S]$.*

2.14.27. *[201803] The following computation is correct for the $M/M/1$ queue:*

$$\begin{aligned} \sum_{n=0}^{\infty} n^2 \rho^n &= \sum_{n=0}^{\infty} \left(\sum_{i=1}^{\infty} 2i \, \mathbb{1}_{i \leq n} - n \right) \rho^n = \sum_{n=0}^{\infty} \sum_{i=0}^{\infty} 2i \, \mathbb{1}_{i \leq n} \rho^n - \sum_{n=0}^{\infty} n \rho^n \\ &= \sum_{i=0}^{\infty} 2i \sum_{n=i}^{\infty} \rho^n - \frac{\mathsf{E}[L]}{1-\rho} = \sum_{i=0}^{\infty} 2i \rho^i \sum_{n=0}^{\infty} \rho^n - \frac{\mathsf{E}[L]}{1-\rho} \\ &= \frac{2}{1-\rho} \sum_{i=0}^{\infty} i \rho^i - \frac{\mathsf{E}[L]}{1-\rho} = \frac{2}{(1-\rho)^2} \mathsf{E}[L] - \frac{\mathsf{E}[L]}{1-\rho} \\ &= \frac{\mathsf{E}[L]}{1-\rho} \left(\frac{2}{1-\rho} - 1 \right) = \frac{\mathsf{E}[L]}{1-\rho} \frac{1+\rho}{1-\rho} \\ &= \frac{\rho}{1-\rho} \frac{1+\rho}{(1-\rho)^2}. \end{aligned}$$

2.14.28. *[201803] For the $M/M/1$ queue, if $\mathsf{E}[L] = \rho/(1-\rho)$ then $\mathsf{E}[W] = \lambda\,\mathsf{E}[L]$.*

2.14.29. *[201803] For the $M/M/1$ queue, $\mathsf{P}(L \le n) = \sum_{k=0}^{n} \rho^k$.*

2.14.30. *[201803] For the $M/M/c$ queue, $\mathsf{E}\left[L_Q\right] = \sum_{n=0}^{\infty} \max\{n-c,0\}p(n)$.*

2.14.31. *[201803] The load of the $M^X/M/1$ queue is $\rho = \lambda\,\mathsf{E}[S]$ where $\mathsf{E}[S]$ is the service time of a single item in a batch.*

2.14.32. *[201803] For the $M/G/1$ queue the following is true:*

$$\lambda\,\mathsf{E}\left[S^2\right] = (1+C_s^2)\,\mathsf{E}[S], \quad \text{where } C_s^2 = \frac{\mathsf{V}[S]}{(\mathsf{E}[S])^2} \tag{2.14.2}$$

is the square coefficient of variation.

2.14.33. *[201803] If a job arrives at time A, and $\{L(t)\}$ the queue length process, then the random variable $L(A)$ denotes the number in the system seen by this job upon arrival.*

2.14.34. *[201803] For the $M^X/M/1$ queue, define*

$$A(m,n,t) = \sum_{k=1}^{A(t)} \mathbb{1}_{L(A_k-)=m}\,\mathbb{1}_{B_k>n-m}$$

as the number of jobs up to time t that see m in the system upon arrival and have batch size larger than $n-m$. Then, $A(n-1,n,t)$ is the number of batches that arrived up to time t.

2.14.35. *[201803] Consider the $M^X/M/1$ queue with partial acceptance: the system can contain at most K jobs, so that when a batch arrives, accept whatever fits in the queue, and reject the rest. The level-crossing equations are then as follows:*

$$\mu\pi(n+1) = \lambda \sum_{m=0}^{n} \pi(m)G(n-m),$$

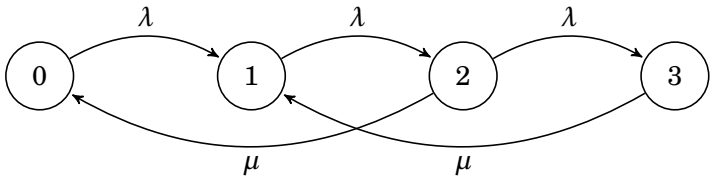
for $n = 0,1,\ldots,K-1$, where $G(n-m) = \mathsf{P}(B > n-m)$ is the survivor function of the random batch size B.

2.14.36. *[201804] For the $G/G/1$ queue, the average number of jobs in the system as seen by arrivals is given by*

$$\frac{1}{t}\int_0^t L(s)\,ds = \frac{1}{t}\int_0^t (A(s)-D(s))\,ds, \tag{2.14.3}$$

where we use that $L(s) = A(s) - D(s) + L(0)$ is the total number of jobs in the system at time s and $L(0) = 0$.

2.14.37. *[201804] Consider the $M^2/M^2/1/3$ queue. The graph below shows all relevant transitions.*



2.14.38. *[201804] For the $M/M/1$ queue, when the server is busy at time 0, the time to the next departure has density $f_D(t) = \mu e^{-\mu t}$.*

2.14.39. *[201804] When $X \sim \text{Exp}(\lambda)$ and $S \sim \text{Exp}(\mu)$, and X and S are independent, their joint density is $f_{X,S}(x,y) = \lambda\mu e^{-\lambda x - \mu y}$. With this,*

$$\begin{aligned}
\mathsf{P}(X+S \le t) &= \lambda\mu \int_0^\infty \int_0^\infty e^{-\lambda x - \mu y} \mathbb{1}_{x+y\le t}\, dx\, dy \\
&= \lambda\mu \int_0^t e^{-\lambda x} \int_0^{t-x} e^{-\mu y}\, dy\, dx \\
&= \lambda \int_0^t e^{-\lambda x}(1 - e^{-\mu(t-x)})\, dx \\
&= \lambda \int_0^t e^{-\lambda x}\, dx - \lambda e^{-\mu t} \int_0^t e^{(\mu-\lambda)x}\, dx
\end{aligned}$$

2.14.40. *[201804] For the $G/G/1$ queue and the definitions of the book, consider state n, i.e., the system contains n jobs. If we count the transitions into and out of state n, the following is true. The number of transitions into state n during $[0,t]$ are given by $A(n,t) + D(n-1,t)$, and the number of transitions out of state n up during $[0,t]$ is given $A(n-1,t) + D(n,t)$.*

2.14.41. *[201804] For the $M^X/M/1$ queue, if B_r is the number of items of the batch currently at the server and $L_{Q,b}$ the number of batches in queue, then*

$$\mathsf{E}[L] = \mathsf{E}\left[L_{Q,b}\right] \mathsf{E}[B] + \mathsf{E}[B_r].$$

2.14.42. *[201804] For the $M^X/M/1$ queue, Let $\tilde{A}_k$ be the moment the kth batch moves to the server and D_k its departure time. When $S_{k,i}$ is the service time of the ith item of batch k,*

$$\int_{\tilde{A}_k}^{D_k} \mathbb{1}_{L_S(s)=i}\, ds = S_{k,i}\, \mathbb{1}_{B_k \ge i}.$$

2.14.43. *[201804] For the $M/G/1$ queue and $S \sim \text{Exp}(\mu)$,*

$$\begin{aligned}
\mathsf{E}\left[S^2\right] &= \mu \int_0^\infty x^2 e^{-\mu x}\, dx = x^2 e^{-\mu x}\Big|_0^\infty + 2\int_0^\infty x e^{-\mu x}\, dx \\
&= 2\frac{x}{\mu} e^{-\mu x}\Big|_0^\infty + \frac{2}{\mu}\int_0^\infty e^{-\mu x}\, dx = \frac{2}{\mu^2}.
\end{aligned}$$

2.14.44. *[201804] For the $M/G/1$ queue, define for $m = 1, \ldots, n$,*

$$D(m,n,t) = \sum_{k=1}^{D(t)} \mathbb{1}_{L(D_{k-1})=m}\, \mathbb{1}_{Y_k > n-m+1},$$

with Y_k the number of arrivals during the service time of the kth job. Then,

$$\begin{aligned}
\lim_{t\to\infty} \frac{D(m,n,t)}{t} &= \lim_{t\to\infty} \frac{D(t)}{t} \frac{D(m,n,t)}{D(t)} \\
&= \delta \lim_{t\to\infty} \frac{D(m,n,t)}{D(t)} \\
&= \delta \mathsf{P}(Y > n-m+1) \\
&= \delta G(n-m+1),
\end{aligned}$$

where G is the survivor function of Y and δ the departure rate.

2.14.45. *[201807] In the level-crossing analysis of the $M(n)/M(n)/1$ queue we claim it is necessary that the interarrival times of jobs are i.i.d.*

2.14.46. *[201807] To prove Little's law for any input-output system we claim that we need for all $T \geq 0$ the property*

$$\int_0^T L(s)\,ds = \sum_{k=1}^{A(T)} W_k.$$

2.14.47. *[201807] For the $M/M/1$ queue we claim that $\mathsf{E}\left[L_Q\right] = \sum_{n=1}^{\infty}(n-1)\pi(n)$.*

2.14.48. *[201807] A repair/maintenance facility would like to determine how many employees should be working in its tool crib. The service time is exponential, with mean 4 minutes, and customers arrive by a Poisson process with rate 28 per hour. With one employee we claim that the system is not rate stable.*

2.14.49. *[201807] In the notes we derived that*

$$\frac{\mathsf{E}\left[L(M^X/M/1)\right]}{\mathsf{E}[L(M/M/1)]} = \frac{\mathsf{E}\left[B^2\right]}{2\mathsf{E}[B]} + \frac{1}{2},$$

when the loads in both queueing systems are the same. We claim that this implies for such systems that $\mathsf{E}\left[L(M^X/M/1)\right] \geq \mathsf{E}[L(M/M/1)]$.

2.14.50. *[201807] For the $G/G/1$ the difference between the number of 'out transitions' and the number of 'in transitions' is at most 1 for all t. As a consequence,*

$$\begin{aligned}
&\text{transitions out} \approx \text{transitions in} \iff \\
&A(n,t) + D(n-1,t) \approx A(n-1,t) + D(n,t) \iff \\
&\frac{A(n,t) + D(n-1,t)}{t} \approx \frac{A(n-1,t) + D(n,t)}{t} \iff \\
&\frac{A(n,t)}{t} + \frac{D(n-1,t)}{t} \approx \frac{A(n-1,t)}{t} + \frac{D(n,t)}{t}.
\end{aligned}$$

Thus, under proper technical assumptions (which you can assume to be satisfied) this becomes for $t \to \infty$,

$$(\lambda(n) + \mu(n))p(n) = \lambda(n-1)p(n-1) + \mu(n+1)p(n+1).$$

We claim that if we specialize this result for the $M/D/1$ queue we have that $\lambda(n) = \lambda$ and $\mu(n) = \mu$, hence using PASTA,

$$(\lambda + \mu)\pi(n) = \lambda\pi(n-1) + \mu\pi(n+1).$$

2.14.51. *[201807] For the $M^X/M/1$ queue we have shown in the notes that*

$$\mu\,\mathsf{E}[L] = \lambda\frac{\mathsf{E}\left[B^2\right]}{2} + \lambda\,\mathsf{E}[B]\,\mathsf{E}[L] + \lambda\frac{\mathsf{E}[B]}{2},$$

With a proper definition for the load ρ we claim that it can be rewritten to

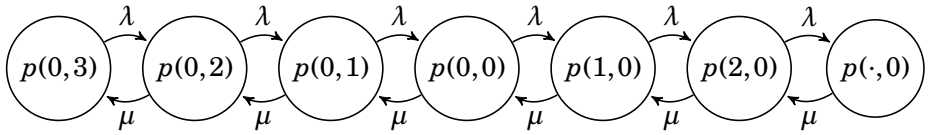
$$(1-\rho)\mathsf{E}[L] = \frac{\rho}{2}\left(\frac{\mathsf{E}\left[B^2\right]}{\mathsf{E}[B]} + 1\right).$$

2.14.52. *[201807] For the $M/G/1$ queue, let us concentrate on a down-crossing of level n; recall that level n lies between states n and $n+1$. We claim that job k only generates a down-crossing of level n this job leaves n jobs behind right after its service completion.*

2.14.53. *[201807] In the notes we derive a recursion to be satisfied by the queue length distribution of the $M/G/1$ queue. To check whether this recursion holds for the $M/M/1$ we are lead to the computation below. Given that $\alpha = \rho/(1+\rho)$, we claim that the computation below entirely correct.*

$$\begin{aligned}
\alpha^{n+1} + \sum_{m=1}^{n} \rho^m \alpha^{n-m+2} &= \alpha^{n+1} + \alpha^{n+2} \sum_{m=1}^{n} (\rho/\alpha)^m \\
&= \alpha^{n+1} + \alpha^{n+1} \rho \sum_{m=0}^{n-1} (\rho/\alpha)^m \\
&= \alpha^{n+1} + \alpha^{n+1} \rho \frac{1-(\rho/\alpha)^n}{1-\rho/\alpha} \\
&= \alpha^{n+1} - \alpha^{n+1} \frac{\rho}{\alpha}(1-(\rho/\alpha)^n).
\end{aligned}$$

2.14.54. *[201807] (Hall 5.22). At a large hotel, taxi cabs arrive at a rate of 15 per hour, and parties of riders arrive at the rate of 12 per hour. Whenever taxicabs are waiting, riders are served immediately upon arrival. Whenever riders are waiting, taxicabs are loaded immediately upon arrival. A maximum of three cabs can wait at a time (other cabs must go elsewhere). Let $p(i,j)$ be the steady-state probability of there being i parties of riders and j taxicabs waiting at the hotel. Claim: the transitions are modeled by the graph below.*



2.14.55. *[201807] Just assume that the figure in the previous question is correct. Claim: the balance equations are as follows:*

$$\begin{aligned}
\lambda p(0,3) &= \mu p(0,2) \\
(\lambda+\mu)p(0,2) &= \mu p(0,1) + \lambda p(0,3) \\
(\lambda+\mu)p(0,1) &= \mu p(0,0) + \lambda p(0,2) \\
(\lambda+\mu)p(0,0) &= \mu p(1,0) + \lambda p(0,1) \\
(\lambda+\mu)p(i,0) &= \mu p(i+1,0) + \lambda p(i-1,0)
\end{aligned}$$

for $i \geq 1$

2.14.56. *[201902] A machine produces items, but a fraction p of the items produced in each period turns out to be faulty. Faulty items have to be repaired. The service time of faulty items is just as long as entirely new items. Repaired items are always ok, in other words, they cannot be faulty again. To keep the system stable, the average service capacity must satisfy*

$$\lim_{n\to\infty} n^{-1} \sum_{i=1}^{n} c_i > \lambda(1+p)$$

where λ is the arrival rate of requests for items.

2.14.57. *[201902] Define $A(A_n-) = \lim_{h\downarrow 0} A(A_n - h)$. Then $A(A_n-) = n-1$.*

2.14.58. *[201902] The number of jobs in the system at time t is equal to*

$$L(t) = A(t) - D(t) = \sum_{k=1}^{\infty} \mathbb{1}_{D_k < t < A_k}.$$

2.14.59. *[201902] Let $N_{\lambda+\mu}$ be a Poisson process with rate $\lambda+\mu$. If $\{a_k\}$ is an i.i.d. sequence of Bernoulli random variables such that $\mathsf{P}(a_k = 1) = \lambda/(\lambda+\mu) = 1 - \mathsf{P}(a_k = 0)$, the random variable*

$$N(t) = \sum_{k=1}^{\infty} a_k \, \mathbb{1}_{k \le N_{\lambda+\mu}(t)},$$

is Poisson distributed with rate μt.

2.14.60. *[201902] We consider the $M/G/1$ queue such that $\mathsf{E}[X] > \mathsf{E}[S]$. In general the expected busy time of the server is $\mathsf{E}[B] = \rho$. (Recall, ρ is the long-run fraction of time the server is busy.)*

2.14.61. *[201902] Consider a $G/G/1$ queue that is rate-stable. The distribution of the waiting times at arrival times can be sensibly defined as*

$$\mathsf{P}(W \le x) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{W_k \le x}.$$

2.14.62. *[201903] Consider the $G/G/1$ queue of the previous exercise. Define*

$$U(t) = \max\{n : D_n \le t\}.$$

The service rate is

$$\mu = \lim_{t\to\infty} \frac{U(t)}{t}.$$

2.14.63. *[201903] Consider the $G/G/1$ queue of the previous exercise. Then $(\mathsf{E}[X] - \mathsf{E}[S])/\mathsf{E}[X]$ is the fraction of time the server is idle.*

2.14.64. *[201903] For the $G/G/1$ queue define*

$$A(n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \le t} \, \mathbb{1}_{L(A_k-)=n},$$

where $L(s)$ is the number of customers in the system at time s. Then $A(n,t)$ counts the number of arrivals up to time t that saw n customers in the system at their arrival.

2.14.65. *[201903] The data below specifies the arrival time of customers, e.g., Jan arrives at time 21. The following code is guaranteed to print the customers in order of arrival*

```
from heapq import heappop, heappush


stack = []

heappush(stack, (21, "Jan"))
heappush(stack, (20, "Piet"))
heappush(stack, (18, "Klara"))
heappush(stack, (25, "Cynthia"))

print(stack)
```

2.14.66. *[201903] For the $M/M/c$ queue we can take*

$$\lambda(n) = \lambda,$$
$$\mu(n) = \begin{cases} n\mu, & \text{if } n \le c, \\ c\mu, & \text{if } n \ge c. \end{cases}$$

Then $p(n) = p(0)(c\rho)^n/n!$ for all n.

2.14.67. *[201903] We can use the PASTA property to conclude that $\sum_{n=0}^{\infty} np(n) = \sum_{n=0}^{\infty} n\pi(n)$ for any $G/M/1$ queue.*

2.14.68. *[201903] When $\lambda > \delta$, then $\pi(n) < \delta(n)$.*

2.14.69. *[201903] For the $M/G/1$ queue,*

$$\mathsf{E}\left[W_Q\right] = \mathsf{E}[L]\,\mathsf{E}[S],$$

that is, the expected time in queue is the expected number of customers in the system times the expected service time of these customers.

2.14.70. *[201903] For the $M/M/1$ queue we have that $\mathsf{P}(L = n) = (1-\rho)\rho^n$. We can use the relation $\sum_{i=1}^{n} i = n(n+1)/2$ to see that $n^2 = -n + 2\sum_{i=1}^{n} i$. Using this result it follows that*

$$\begin{aligned} \mathsf{E}\left[L^2\right] &= (1-\rho)\sum_{n=0}^{\infty} n^2\rho^n \\ &= (1-\rho)\sum_{n=0}^{\infty}\left(\sum_{i=1}^{\infty} 2i\,\mathbb{1}_{i\le n} - n\right)\rho^n \\ &= (1-\rho)\sum_{n=0}^{\infty}\sum_{i=0}^{\infty} 2i\,\mathbb{1}_{i\le n}\rho^n - \sum_{n=0}^{\infty} n\rho^n. \end{aligned}$$

2.14.71. *[201903] A single-server queueing system is known to have Poisson arrivals and exponential service times. However, the arrival rate and service time are state dependent. The manager observes that when the queue becomes longer, servers work faster, and the arrival rate declines. The following choice for $\lambda(n)$ and $\mu(n)$ are consistent with the manager's observation: $\lambda(0) = 5$, $\lambda(1) = 3$, $\lambda(2) = 2$, $\lambda(n) = 0, n \ge 3$, $\mu(0) = 0$, $\mu(1) = 2$, $\mu(2) = 3$, $\mu(n) = 4, n \ge 3$.*

2.14.72. *[201903] Consider the $G/G/2$ queue with $P(0) = 0.4$, $P(1) = 0.3$, $P(2) = 0.2$, $P(3) = 0.05$, $P(4) = 0.05$. (Here, $P(n)$ is the fraction of time the system contains n jobs.) Then with the following code we can compute the (time) average number of jobs in queue.*

```
>>> P = [0.4, 0.3, 0.2, 0.05, 0.05]
>>> ELQ = sum(n*P[n] for n in range(len(P)))
```

2.14.73. *[201903] Consider a queueing system in which we normally have 1 server working at rate $\mu = 4$. When the queue becomes longer than a threshold at 20, we hire one extra server that also works at rate $\mu = 4$, and when the queue is empty again, we send the extra servers home, until the queue hits 20 again, and so on. The following code implements this behavior of the extra server in a correct way.*

```
import numpy as np

from scipy.stats import poisson

labda = 3
mu = 4
a = poisson(labda).rvs(100000)
Q = np.zeros_like(a)
d = np.zeros_like(a)

threshold = 20

for i in range(1, len(a)):
 if Q[i-1] < threshold:
 c = poisson(mu).rvs()
 elif Q[i-1] >= threshold:
 c = poisson(2*mu).rvs()
 d[i] = min(Q[i-1],c)
 Q[i] = Q[i-1] + a[i] - d[i]
```

2.14.74. *[201903] Consider the $M^X/M/1$ queue with B denoting the batch size of an arriving batch. Then*

$$\frac{\mathsf{E}\left[B^2\right]}{\mathsf{E}[B]} = (1+C_s^2)\mathsf{E}[B], \quad \textit{where } C_s^2 = \frac{\mathsf{V}[B]}{(\mathsf{E}[B])^2},$$

2.14.75. *[201903] Consider the $M/G/1$ queue. Denote by $\tilde{A}_k$ the time job k starts service and by D_k its departure time, $k = 1,\ldots,n$. Then, the expression*

$$\sum_{k=1}^{n} \int_0^{D_n} (D_k - s)\mathbb{1}_{\tilde{A}_k \le s < D_k}\, ds$$

computes the total remaining service time up to time $t = D_n$.

2.14.76. *[201904] A machine serves two types of jobs. The processing time of jobs of type i, $i = 1,2$, is exponentially distributed with parameter μ_i. The type T of a job is random and independent of anything else, and such that $\mathsf{P}(T = 1) = p = 1 - q = 1 - \mathsf{P}(T = 2)$. Then,*

$$\mathsf{E}[S] = p/\mu_1 + q/\mu_2.$$

2.14.77. *[201904] The $M/G/c/K$ shorthand means that jobs arrive as a Poisson process, job service times are exponentially distributed, and there are c servers.*

2.14.78. *[201904] Consider the server of the $G/G/1$ queue as a system by itself. The time jobs stay in this system is $\mathsf{E}[S]$, and jobs arrive at rate λ. It follows from Little's law that the fraction of time the server is busy is $\lambda\mathsf{E}[S]$.*

2.14.79. *[201904] If $A(n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \le t}\mathbb{1}_{L(A_k-)=n}$, is $A(t) = \sum_{n=0}^{\infty} A(n,t)$?*

2.14.80. *[201904] If $\lambda > \delta$ it can happen that $\lim_{t\to\infty} A(n,t)/t > 0$ for some (finite) n.*

2.14.81. *[201904] Let*

$$D(n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{D_k \le t} \mathbb{1}_{L(D_k)=n}, \qquad Y(n,t) = \int_0^t \mathbb{1}_{L(s)=n}\, ds$$

denote the number of departures up to time t that leave n customers behind and the total time the system contains n jobs during $[0,t]$. Then, the departure rate from state $n+1$ is

$$\mu(n+1) = \lim_{t\to\infty} \frac{D(n+1,t)}{Y(n+1,t)},$$

2.14.82. *[201904] As $K \to \infty$, the performance measures of the $M/M/1/K$ converge to those of the $M/M/1$ queue.*

2.14.83. *[201904] To model the $M/M/c/c+K$ queue as an $M(n)/M(n)/1$ queue we need to take $\lambda(n) = \lambda$ for all n.*

2.14.84. *[201904] Take*

$$\pi(n) = \lim_{t\to\infty} \frac{A(n,t)}{A(t)}, \qquad \delta(n) = \lim_{t\to\infty} \frac{D(n,t)}{D(t)}.$$

Then, $\lambda \neq \delta \implies \pi(n) > \delta(n)$.

2.14.85. *[201904] Consider a* $M/D/1$ *queue with* $\lambda = 1$ *and* $\mathsf{E}[S] = 0.10$. *Then the SCV of its departure process is smaller than* 0.5.

2.14.86. *[201904] For a given single-server queueing system the average number of customers in the system is* $\mathsf{E}[L] = 10$, *customers arrive at rate* $\lambda = 4$ *per hour and are served at rate* $\mu = 5$ *per hour. At the moment you join the system, the number of customers in the system is 10. Your expected time in the system is, by Little's law,* $\mathsf{E}[W] = \mathsf{E}[L]/\lambda = 2.5$ *hour.*

2.14.87. *[201904] When* $\mathsf{V}[S] = 0$, *it follows for the remaining service time* S_r *that*

$$\mathsf{E}[S_r \,|\, S_r > 0] = \frac{\mathsf{E}[S^2]}{2\,\mathsf{E}[S]} \implies \mathsf{E}[S_r \,|\, S_r > 0] = \frac{\mathsf{E}[S]}{2}$$

2.14.88. *[201907] For the* $M/G/1$ *queue with rate* $\lambda = 1$ *per hour,* $\mathsf{P}(A_k = k \text{ for all } k) > 0$.

2.14.89. *[201907] For the computation of the waiting time of the single-server queue we assume that all random variables in the sequences* $\{X_k\}$ *and* $\{S_k\}$ *are independent. This a necessary condition to compute the set of waiting times* $\{W_k\}$.

2.14.90. *[201907] Let* $N_{\lambda+\mu}$ *be a Poisson process with rate* $\lambda + \mu$. *If* $\{a_k\}$ *is an i.i.d. sequence of Bernoulli random variables such that* $\mathsf{P}(a_k = 1) = \mu/(\lambda+\mu) = 1 - \mathsf{P}(a_k = 0)$, *the random variable*

$$N(t) = \sum_{k=1}^{\infty} a_k \, \mathbb{1}_{k \leq N_{\lambda+\mu}(t)},$$

has a Poisson distribution with rate λt.

2.14.91. *[201907] Take the stable* $M(n)/M/1$ *queue with* $\lambda(15) = 0$. *Suppose that the queue length starts at* 100, *i.e.,* $Q(0) = 100$. *Then* $\pi(90) > 0$.

2.14.92. *[201907] If* $L(t)/t \to 0$ *as* $t \to \infty$ *it can still be true that* $0 < \mathsf{E}[L] < \infty$.

2.14.93. *[201907] Consider the (stable)* $M/G/1$ *queue. The density* f_D *of the interdeparture times is equal to the density* f_S *of the service times.*

2.14.94. *[201907] Suppose there are 10 jobs present at the* $M/M/1$ *queue with arrival rate* $\lambda = 3$ *and service rate* $\mu = 4$ *per hour. The time to clear the system follows from Little's law and is* $3 \cdot 10 = 30$ *hours.*

2.14.95. *[201907] For the* $M^X/M/1$ *we have the recursion*

$$\pi(n) = \frac{\lambda}{\mu} \sum_{i=0}^{n-1} \pi(n-1-i) G(i).$$

The following code can be used to compute the unnormalized *probabilities for* $p(1), \ldots, p(4)$ *of the* $M/M/1$ *queue. (Recall that* `range[1,5]` *goes up to, but does not include,* 5.*)*

```
p[0] = 1
G = [1, 0]
for n in range(1, 5): # this goes up to, but does not include, 5
 p[n] = labda / mu * sum(p[n - 1 - i] * G[i] for i in range(min(n, len(G))))
```

2.14.96. *[201907] For the $M/G/1$ queue the up-crossing rate of level n is equal to*

$$\delta\delta(0)G(n)+\delta\sum_{m=1}^{n}\delta(m)G(n-m+1), \tag{2.14.4}$$

where $G(j)=\mathsf{P}(Y>j)$ is the survivor function of the number of arrivals Y during a service time, δ is the long-run departure rate and $\delta(n)$ the probability to leave n jobs behind.

2.14.97. *[201907] Consider a single-server queueing in discrete time, $k=0,1,\ldots$. The machine can switch between a high and a slow production speed. When the queue is larger than or equal to M at the start of period k, the machine switches to a high speed c_+; when the queue becomes smaller or equal to m, $0\le m<M$, the machine switches to the low speed $c_-<c_+$, otherwise the machine's speed remains the same. The variable I_k keeps track of the state of the server and satisfies*

$$I_{k+1}=c_+\,\mathbb{1}_{Q_k\ge M}+I_k\,\mathbb{1}_{m<Q_k<M}+c_-\,\mathbb{1}_{Q_k\le m}.$$

2.14.98. *[201907] Consider the (stable) $M/D/1$ queue. The SCV of the departure process is always less than 1.*

2.14.2 *Open Questions*

2.14.99. *[201704] Show with a level-crossing argument that*

$$\lambda\pi(n)=\mu(n+1)p(n+1) \tag{2.14.5}$$

for a queueing system in which jobs arrive and depart in single units.

2.14.100. *[201704] What condition should be satisfied in the above Equation (2.14.5) so that PASTA holds?*

2.14.101. *[201704] The server of an $M/M/1$ queue fails, with constant failure rate, once per 10 days. The repair times are exponentially distributed with a mean of one day. The job service times without failures have a mean of 2 days. Jobs arrive with rate $\lambda=1/3$ per day. What is $\mathsf{E}[W]$?*

2.14.102. *[201704]Consider the $M^X/M/1$ queue with $G(n)=\mathsf{P}(B>n)$. With level-crossing arguments we* nearly *get*

$$\lambda\sum_{m=0}^{n}G(n-1-m)\pi(m)=\mu\pi(n+1).$$

What is wrong with this formula, and repair it.

2.14.103. *[201706] Show that an exponentially distributed random variable is memoryless.*

2.14.104. *[201706] Suppose an $M/M/1$ queue contains 5 jobs at time some $t>0$. Jobs arrive at rate λ and have average service time μ^{-1}. What is the distribution of the time until the next event (arrival or departure epoch)?*

2.14.105. *[201706] Suppose an $M/M/4$ queue contains 5 jobs at time some $t > 0$. Jobs arrive at rate λ and have average service time μ^{-1}. What is the distribution of the time until the next event (arrival or departure epoch)?*

2.14.106. *[201706] If the inter-arrival times $\{X_i\}$ are i.i.d. and exponentially distributed with mean $1/\lambda$, prove that the number $N(t)$ of arrivals during interval $[0,t]$ is Poisson distributed. Use that $\mathsf{P}(A_k \leq t) = \int_0^t \lambda e^{-\lambda s} \frac{(\lambda s)^{k-1}}{(k-1)!}\, ds$, where A_k is the arrival time of the kth arrival.*

2.14.107. *[201706] Consider a single-server queueing in discrete time, $k = 0, 1, \ldots$. The machine can switch between a high and a low production speed. When the queue is larger than or equal to M at the start of period k, the machine switches to the high speed c_+; when the queue becomes smaller or equal to m, $0 \leq m < M$, the machine switches to the low speed $c_- < c_+$, otherwise the machine's speed remains the same. The arrival process is given by $\{a_k\}$. Assuming the arrivals a_k in period k cannot be served on day k, establish a set of recursions to enable a simulation of this system. Assume that $Q_0 = 0$.*

2.14.108. *[201706] For the previous problem, provide a formula to compute (count) the number of times the machine switches to the fast state during the first n periods.*

2.14.109. *[201706] Related to the previous problem, assuming that the average arrival rate of jobs $a = \lim_{n\to\infty} 1/n \sum_k^n a_k$ is such that $c_- < a < c_+$. What is the average cycle time, i.e., the average time between two moments the machine switches to the fast state? (More specifically, let τ_n be the nth time the machine switches to the fast state. What is $\lim_{n\to\infty} n^{-1}\mathsf{E}[\tau_n]$?)*

2.14.110. *[201706] Provide a real-world example for the queueing model of the previous three problems.*

2.14.111. *[201706] Can you make an arrival process such that $\frac{A(t)}{t}$ as $t \to \infty$ does not have a limit?*

2.14.112. *[201706] Consider the $M/M/1$ queue with arrival rate $\lambda = 3$ per hour and average service times $\mathsf{E}[S] = 10$ minutes. What is the value of*

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{L(A_k-)=3}?$$

2.14.113. *[201706] State explicitly all relevant assumptions you needed to make in the previous problem to obtain an answer.*

2.14.114. *[201706] The queueing system at a fast-food stand behaves in a peculiar fashion. When there is no one in the queue, people are reluctant to use the stand, fearing that the food is unsavory. People are also reluctant to use the stand when the queue is long. This yields the following arrival rates (in numbers per hour): $\lambda(0) = 10$, $\lambda(1) = 15$, $\lambda(2) = 15$, $\lambda(3) = 10$, $\lambda(4) = 5$, $\lambda(n) = 0, n \geq 5$. The stand has two servers, each of which can operate at 5 customers per hour. Service times are exponential, and the arrival process is Poisson. Calculate the steady-state probabilities.*

2.14.115. *[201706] For the previous question, customers spend 10 Euro on average on an order. What is the rate at which the stand makes money? What theorem(s) do you need to use to compute this?*

2.14.116. *[201804] Show with a level-crossing argument that for the $M/M/1$ queue*

$$\pi(n) = \delta(n). \tag{2.14.6}$$

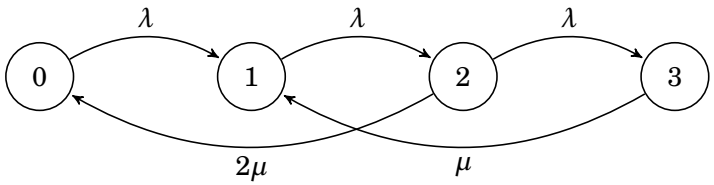
2.14.117. *[201804] What condition should be satisfied in* (2.14.6) *so that it holds?*

2.14.118. *[201804] Does* (2.14.6) *also hold for the $G/G/1$ queue (motivate).*

2.14.119. *[201804] Does* (2.14.6) *imply the PASTA property (motivate).*

2.14.120. *[201804] Use the Pollack-Khintchine equation to show for the $M/M/1$ queue that $\mathsf{E}[L] = \rho/(1-\rho)$.*

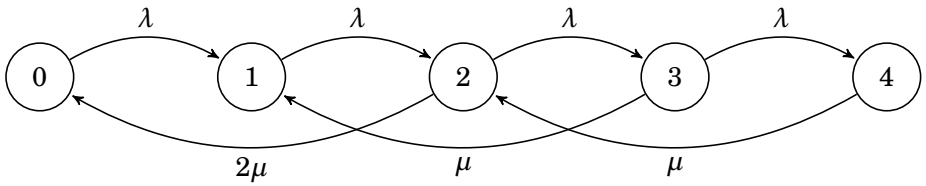
2.14.121. *[201807] Consider the queueing system below. Jobs arrive as a Poisson process with rate λ. Service times are exponential service with mean $(2\mu)^{-1}$ when there are two jobs in the system, and mean μ^{-1} when there are 3 jobs. What is $p(2)$ if $\lambda = \mu = 1$?*



2.14.122. *[201807] What is the fraction of lost jobs?*

2.14.123. *[201807] Suppose the manager is unsatisfied with the loss rate. Writing $x = \lambda/\mu$ for ease, what condition should x satisfy such that the loss probability is less than some threshold α? (Just show the condition on x; you do not have to solve for x.)*

2.14.124. *[201807, Continuation of previous exercise, 1] Assume again that $\lambda = \mu = 1$. Would the loss be reduced if the manager extends the system to four 4 positions like this:*



2.14.125. *[201807] Consider the $G/G/n/K$ queue (specifically, the system can contain at most K jobs). Let λ be the arrival rate, μ the service rate, β the long-run fraction of customers lost, and ρ the average number of busy/occupied servers Show that*

$$\beta = 1 - \rho\frac{\mu}{\lambda}. \tag{2.14.7}$$

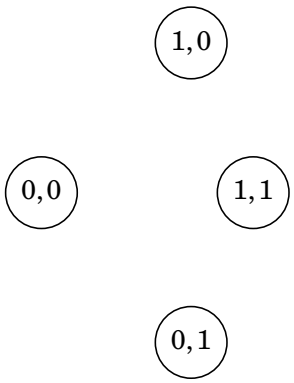
2.14.126. *[201807] Explain why* (2.14.7) *not applicable for the queueing system of question 2.14.121.*

2.14.3 *Parallel servers*

In the $M/M/c$ queue it is assumed that all servers work at the same rate. However, in practice, this is not always the case. In this section, from 2.14.127–1.2.8, we make a model by which we can analyze the effects of servers with different speeds.

A station has two parallel servers, the service times of the first are $\text{Exp}(\mu_1)$ and of the second $\text{Exp}(\mu_2)$. Jobs arrive as a Poisson process with rate λ. The queue cannot contain a job, hence any job arriving when both servers are busy is rejected. When the system is empty, jobs are routed to the first server.

2.14.127. *[201904] Explain that the figure below contains all the relevant states.*



2.14.128. *[201904] Add arrows to the figure of the state space to indicate the possible transitions and add the rates.*

2.14.129. *[201904] Find an expression for the long-run fraction of time the system is empty. (You might want to use balance equations here.)*

2.14.130. *[201904] Express the long-run fraction of lost jobs in terms of one of the probabilities $p(0,0)$, $p(1,0)$, $p(0,1)$, $p(1,1)$. Explain your reasoning.*

2.14.131. *[201904] By making the simplifying assumption that $\mu_1 = \mu_2$ the above system reduces to a simpler queueing system for which we have closed-form solutions for the state probabilities. What is this simpler queueing system?*

Suppose we can extend the system such that one job can be queued.

2.14.132. *[201904] Make a sketch of the state space and include the transitions and rates.*

2.14.133. *[201904] Show how you can use level-crossing arguments to express the probability to find a job in queue in terms of one of the probabilities $p(0,0)$, $p(1,0)$, $p(0,1)$, $p(1,1)$.*

2.14.134. *[201904] Suppose now that the queue is infinite so that jobs are never lost. Approximate the queueing system by a suitable $G/G/c$ queue and use Sakasegawa's formula to estimate $\mathsf{E}\left[W_Q\right]$. Take $\mu_1 = 6$, $\mu_2 = 3$, $\lambda = 8$.*

2.14.135. *[201904] For the $M^X/M/1$ queue we derived the expression that $p(n)$, i.e., the fraction of time the system contains n jobs, must satisfy*

$$\lambda \sum_{m=0}^{n} G(n-m)p(m) = \mu p(n+1), \tag{2.14.8}$$

where $G(k) = \mathsf{P}(B > k)$. Explain this formula, by a drawing or in words (or both).

2.14.136. *[201904] For the $M^X/M/1$ queue we have that*

$$\mu\,\mathsf{E}[L] = \mu \sum_{n=0}^{\infty} n\pi(n) = \lambda \frac{\mathsf{E}\left[B^2\right]}{2} + \lambda\,\mathsf{E}[B]\,\mathsf{E}[L] + \lambda \frac{\mathsf{E}[B]}{2}. \tag{2.14.9}$$

Show that this reduces to $\mathsf{E}[L] = \rho/(1-\rho)$ for the $M/M/1$ queue.

2.14.137. *[201904] Explain (briefly) why checks such as in the previous exercise are important.*

Solutions

s.2.14.1. Answer = B. We should define

$$\alpha = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} L(A_k -).$$

s.2.14.2. Answer = A.

s.2.14.3. Answer = A.

$$\mathsf{E}\left[L_Q\right] = \frac{\rho^2}{1-\rho}.$$

s.2.14.4. Answer = A.

s.2.14.5. Answer = A.

s.2.14.6. Answer = A.

s.2.14.7. Answer = B. $\rho = \lambda/(c\mu)$, and the rest of the formula is also wrong:

$$p(n+1) = \frac{\lambda}{\mu(n+1)} p(n) = \frac{\lambda}{\min\{c, n+1\}\mu} p(n)$$

etc.

s.2.14.8. Answer = B. $\pi(0) = 1$.

s.2.14.9. Answer = A.

s.2.14.10. Answer = B.

$$\mathsf{E}[L] = \frac{d}{dz}\phi(z)\bigg|_{z=1} = \frac{\rho}{1-\rho}.$$

s.2.14.11. Answer = B. We have

$$\begin{aligned}\frac{\lambda}{\mu}\mathsf{E}\left[B^2\right] &= \frac{\lambda\,\mathsf{E}[B]}{\mu}\frac{\mathsf{E}\left[B^2\right]}{(\mathsf{E}[B])^2}\mathsf{E}[B] = \rho\frac{\mathsf{E}\left[B^2\right]}{(\mathsf{E}[B])^2}\mathsf{E}[B] \\ &= \rho\frac{(\mathsf{E}[B])^2 + \mathsf{V}[B]}{(\mathsf{E}[B])^2}\mathsf{E}[B] = \rho(1+C_s^2)\mathsf{E}[B].\end{aligned}$$

s.2.14.12. Answer = A.

s.2.14.13. Answer = A.
$\rho = \lambda\,\mathsf{E}[S] = 6/8 = 3/4$. Hence, $\rho/(1-\rho) = 3$.

$$\mathsf{E}\left[W_Q\right] = \frac{1+1/2}{2}3\,\mathsf{E}[S] = 3/4 \cdot 3 \cdot 2 = 9/2 = 4.5$$

s.2.14.14. Answer = A.

s.2.14.15. Answer = A.

s.2.14.16. Answer = B. When $a_k = 0$, there are no job arrivals, but still L_k can be positive.

s.2.14.17. Answer = A.

s.2.14.18. Answer = A.

s.2.14.19. Answer = A.

s.2.14.20. Answer = B.

$$\frac{1}{t}\sum_{k=1}^{A(t)} \mathbb{1}_{W_k \le x} = \frac{A(t)}{t}\frac{1}{A(t)}\sum_{k=1}^{A(t)} \mathbb{1}_{W_k \le x} \to \lambda \mathsf{P}(W \le x).$$

Moreover, the last w in the equation in the exercise is also wrong, and should be an x.

s.2.14.21. It is wrong. Suppose that the system starts empty and only one job arrives at $t = 1/3$. Consider now a time $t = 2/3$. Then $Y(0,t) = 1/3$, $Y(1,t) = 1/3$, $A(0,t) = 1$ and $D(n,t) = 0$. Then the expression at the left-hand side becomes 3/2. Hence, the statement is not correct for all $t > 0$;

s.2.14.22. Answer = A.

s.2.14.23. Answer = A.

s.2.14.24. Answer = B. There were some questions about this. So lets explain this a bit better. In a system with finite calling population, b say, the arrival rate is $\lambda(n) = \lambda(b-n)$ when there are n jobs in service or in queue. For a queueing system with balking *at* level b, $\lambda(n) = \lambda$ for all $n < b$, and $\lambda(n) = 0$ for $n \geq b$. Thus, the arrival processes are different, hence the queueing systems must behave differently.

s.2.14.25. Answer = A.

s.2.14.26. Answer = A.

s.2.14.27. Answer = A.

s.2.14.28. Answer = B.

s.2.14.29. Answer = B. $\mathsf{P}(L \leq n) = (1-\rho)\sum_{k=0}^{n} \rho^k$. It's evident in the question that the normalization misses.

s.2.14.30. Answer = A.

s.2.14.31. Answer = B.

s.2.14.32. Answer = B.

$$\lambda \mathsf{E}\left[S^2\right] = (1 + C_s^2)\rho \mathsf{E}[S], \quad \text{where } C_s^2 = \frac{\mathsf{V}[S]}{(\mathsf{E}[S])^2}. \tag{2.14.10}$$

s.2.14.33. Answer = B. It should be $L(A-)$.

s.2.14.34. Answer = B. It should be $A(n, n, t)$.

s.2.14.35. Answer = A.

s.2.14.36. Answer = B. The expressions become (in the limit $t \to \infty$) to the time-average of the number of jobs in the system. In general, this time-average is not what the jobs see upon arrival.

s.2.14.37. Answer = B. The figure sketches the $M/M^2/1/3$ queue.

s.2.14.38. Answer = A. When the server is busy at time 0, we need to wait until the job in service departs, as this is the next departure. As service times are Exp(μ) and memoryless, the expression follows.

s.2.14.39. Answer = A.

s.2.14.40. Answer = B. It's precisely the other way around.

s.2.14.41. Answer = A.

s.2.14.42. I decided to accept any answer to this exercise, as it is possible to read it in two ways, which I did not realize at first. One way is that the ith item is the one such that still i items remain (in which case the equation is correct). The other is that $i-1$ items have been served, and that the server is processing the ith item in the sequence, in which case the system contains $B-i-1$.

s.2.14.43. Answer = B. It should be this:

$$\begin{aligned}\mathsf{E}\left[S^2\right] &= \mu\int_0^\infty x^2 e^{-\mu x}\,\mathrm{d}x = -\,x^2 e^{-\mu x}\big|_0^\infty + 2\int_0^\infty x e^{-\mu x}\,\mathrm{d}x \\ &= -\,2\frac{x}{\mu}e^{-\mu x}\Big|_0^\infty + \frac{2}{\mu}\int_0^\infty e^{-\mu x}\,\mathrm{d}x \\ &= -\frac{2}{\mu}\,e^{-\mu x}\big|_0^\infty = \frac{2}{\mu^2}.\end{aligned}$$

Note the minus signs.

s.2.14.44. Answer = B.

s.2.14.45. Answer = B.

s.2.14.46. Answer = B. This only holds at times T at which the system is empty.

s.2.14.47. Answer = A.

s.2.14.48. Answer = A.

s.2.14.49. Answer = A.

s.2.14.50. Answer = B.

s.2.14.51. Answer = A.

s.2.14.52. Answer = B. In fact, two events must be satisfied:

$$\text{Down-crossing of level } n \iff \mathbb{1}_{L(D_{k-1})=n+1}\,\mathbb{1}_{L(D_k)=n} = 1.$$

s.2.14.53. Answer = B. The last line should be

$$\alpha^{n+1} - \alpha^{n+1}(1-(\rho/\alpha)^n).$$

s.2.14.54. Answer = A.

s.2.14.55. Answer = A.

s.2.14.56. Answer = A.

s.2.14.57. Answer = A.

s.2.14.58. Answer = B. $D_k \geq A_k$ always.

$$L(t) = A(t) - D(t) \sum_{k=1}^{\infty} \mathbb{1}_{A_k \leq t} - \sum_{k=1}^{\infty} \mathbb{1}_{D_k \leq t}.$$

s.2.14.59. Answer = B. It is Poisson distributed with rate λt.

s.2.14.60. Answer = B. The server utilization is ρ. Check exercise 1.7.10.

s.2.14.61. Answer = A.

s.2.14.62. Answer = B.

s.2.14.63. Answer = A.

s.2.14.64. Answer = A.

s.2.14.65. Answer = B.

```
>>> from heapq import heappop, heappush


>>> stack = []

>>> heappush(stack, (21, "Jan"))
>>> heappush(stack, (20, "Piet"))
>>> heappush(stack, (18, "Klara"))
>>> heappush(stack, (25, "Cynthia"))

>>> print(stack)
[(18, 'Klara'), (21, 'Jan'), (20, 'Piet'), (25, 'Cynthia')]
```

s.2.14.66. Answer = B.

s.2.14.67. Answer = B. In the $G/M/1$ queue jobs don't arrive as a Poisson process.

s.2.14.68. Answer = B.
In fact, if $\lambda > \delta$, then $p(n) = 0 = \delta(n)$ for all n.

s.2.14.69. Answer = B. In the $M/G/1$ queue the remaining service time of the job in service (if there is any), is not $\mathsf{E}[S]$.

s.2.14.70. Answer = B.

$$\begin{aligned}
\mathsf{E}\left[L^2\right] &= (1-\rho)\sum_{n=0}^{\infty} n^2\rho^n \\
&= (1-\rho)\sum_{n=0}^{\infty}\left(\sum_{i=1}^{\infty} 2i\,\mathbb{1}_{i\le n} - n\right)\rho^n \\
&= (1-\rho)\sum_{n=0}^{\infty}\sum_{i=0}^{\infty} 2i\,\mathbb{1}_{i\le n}\rho^n - (1-\rho)\sum_{n=0}^{\infty} n\rho^n.
\end{aligned}$$

s.2.14.71. Answer = A.

s.2.14.72. Answer = B. It's the number of jobs in the system, not in queue.

s.2.14.73. Answer = B. The extra server switches off when the queue length becomes below 20.

s.2.14.74. Answer = A.

s.2.14.75. Answer = A.

s.2.14.76. Answer = A, 2.10.23.

s.2.14.77. Answer = B. 1.4.8

s.2.14.78. Answer = A, 2.8.6.

s.2.14.79. Answer = A, 2.4.2

s.2.14.80. Answer = B, 2.4.4

s.2.14.81. Answer = B, 2.4.8

s.2.14.82. Answer = A, 2.6.2

s.2.14.83. Answer = B, 2.6.11

s.2.14.84. Answer = B, 2.7.3.

s.2.14.85. Answer = B.

```
>>> labda = 1
>>> ES = 0.10
>>> Ca = 1
>>> Cs = 0
>>> rho = labda*ES
>>> rho
0.1
>>> Cd = (1-rho*rho)*Ca + rho*rho*Cs
>>> Cd
0.99
```

s.2.14.86. Answer = B, 2.8.8. Depending on the service distribution, the expected time can be 10/2, but it can also be something else. This depends on the distribution of the remaining service time of the job in service at the moment you arrive.

s.2.14.87. Answer = A, 2.10.19

s.2.14.88. Answer = B. $\mathsf{P}(A_k = k \text{ for all } k) = 0$.

s.2.14.89. Answer = B. Of course this is not necessary. For instance, in a simulation these random variables are not independent.

s.2.14.90. Answer = B, 1.7.2

s.2.14.91. Answer = B. It is zero. Note that the services are described by an M, hence the service rate is μ at all stations. Since the queue is stable, it is necessary that $\mu > 0$.

s.2.14.92. Answer = A, 2.3.2.

s.2.14.93. Answer = B, 2.7.14.

s.2.14.94. Answer = B.

s.2.14.95. Answer = A.

s.2.14.96. Answer = A, see (2.12.5).

s.2.14.97. Answer = A.

s.2.14.98. Answer = A.

s.2.14.99. $|A(n,t) - D(n,t)| \le 1$. Hence

$$\frac{A(t)}{t}\frac{A(n,t)}{A(t)} \approx \frac{D(n+1,t)}{Y(n+1,t)}\frac{Y(n+1,t)}{t}.$$

Now take the limit $t \to \infty$.

Just saying that level-crossing implies $\lambda\pi(n) = \mu(n+1)p(n+1)$ is of course not a good answer.

s.2.14.100.

$$\frac{A(n,t)}{t} = \frac{A(t)}{t}\frac{A(n,t)}{A(t)}\frac{A(n,t)}{Y(n,t)}\frac{Y(n,t)}{t}.$$

Now take the limit $t \to \infty$ to get

$$\lambda\pi(n) = \lambda(n)p(n).$$

Hence, when $\lambda = \lambda(n)$ for all n, $\pi(n) = p(n)$, which is the PASTA property.

Some students said that $p(n) = \pi(n)$, but this is the meaning of PASTA, not the condition for PASTA. (For the curious, there is an Anti-PASTA theorem.)

If you would just have said something like 'Poisson arrivals', or 'exponential interarrival times', I gave one point. The answer is not complete.

s.2.14.101. The availability is $A = 9/10$. Hence, $\mathsf{E}[S_e] = 2/(9/10)) = 20/9$. We also have

$$C_e^2 = C_0^2 + 2A(1-A)\frac{m_r}{\mathsf{E}[S_0]} = 1 + 2\frac{9}{10}\frac{1}{10}\frac{2}{.}$$

Now,

$$\rho = \lambda\,\mathsf{E}[S_e] = \frac{1}{3}\frac{20}{9} = \frac{20}{27}$$

and

$$\mathsf{E}\left[W_Q\right] = \frac{1+C_e^2}{2}\frac{\rho}{1-\rho}\mathsf{E}[S] = \frac{1+C_e^2}{2}\frac{20/27}{7/27}\frac{20}{9}.$$

Finally, $EW = \mathsf{E}\left[W_Q\right] + \mathsf{E}[S_e]$.

It is essential that you should that both $\mathsf{E}[S]$ and C_e^2 are affected by failures. If you forgot to compensate in either of the two, I subtracted one point.

s.2.14.102.

$$\lambda \sum_{m=0}^{n} G(n-m)\pi(m) = \mu\pi(n+1).$$

s.2.14.103. See the book.

No points if you do not explicitly use the exponential distribution.

s.2.14.104. See the book. $\mathsf{P}(\min\{X,S\} > x) = \exp -(\mu+\lambda)x$.

A number of students don't seem to understand the difference between time and number. The time to the next event is exponentially distributed. The fact that the number of jobs arriving in a certain amount of time is Poisson distributed is not the same as that the time to the next event is Poisson distributed.

Stating that the time is $\min\{X,Y\}$ where X and Y are exp. distr. random variables gives 1/2 point.

s.2.14.105. See the book. $\mathsf{P}(\min\{X,S_1,\ldots,S_4\} > x) = \exp -(4\mu+\lambda)x$.

s.2.14.106. See the book.

We want to show that

$$\mathsf{P}(N(t)=k) = e^{-\lambda t}\frac{(\lambda t)^k}{k!}.$$

Now observe that $\mathsf{P}(N(t)=k) = \mathsf{P}(A_k \le t) - \mathsf{P}(A_{k+1} \le t)$. Using the density of A_{k+1} as obtained previously and applying partial integration leads to

$$\begin{aligned}
\mathsf{P}(A_{k+1} \le t) &= \lambda \int_0^t \frac{(\lambda s)^k}{k!} e^{-\lambda s}\,\mathrm{d}s \\
&= \lambda \frac{(\lambda s)^k}{k!}\frac{e^{-\lambda s}}{-\lambda}\Big|_0^t + \lambda \int_0^t \frac{(\lambda s)^{k-1}}{(k-1)!} e^{-\lambda s}\,\mathrm{d}s \\
&= -\frac{(\lambda t)^k}{k!} e^{-\lambda t} + \mathsf{P}(A_k \le t)
\end{aligned}$$

We are done.

Half point for stating that we are looking for $\mathsf{P}(A_k \le t) - \mathsf{P}(A_{k+1} \le t)$. Some students reversed these two probabilities. It is simple to see that that is wrong.

For a full point I also wanted to see the algebra that leads to the final answer.

s.2.14.107. First we need to implement the switching policy. For this we need an extra variable to keep track of the state of the server. Let $I_k = c_+$ if the machine is working fast in period k and $I_k = c_-$ if it is working slowly. Then $\{I_k\}$ must satisfy the relation

$$I_{k+1} = \begin{cases} c_+ & \text{if } Q_k \ge M, \\ c_- & \text{if } Q_k <= m, \\ I_k & \text{else.} \end{cases}$$

and assume that $I_0 = c_+$ at the start, i.e., the machine if off. Thus, we can write:

$$I_{k+1} = c_+ \mathbb{1}_{Q_k \ge M} + I_k \mathbb{1}_{m<Q_k<M} + c_- \mathbb{1}_{Q_k \le m}.$$

With I_k it follows that $d_k = \min\{Q_{k-1}, I_k\}$, from which Q_k follows, and so on.

I want to see an update rule for the state of the machine. If that is missing or wrong, 1/2 point.

s.2.14.108.

$$\sum_{k=1}^{n} \mathbb{1}_{I_{k-1}=c_-, I_k=c_+}.$$

The expression

$$\sum_{k=1}^{n} \mathbb{1}_{Q_k>m}$$

counts too many periods.

This,

$$\sum_{k=1}^{n} (\mathbb{1}_{I_{k-1}=c_-} - \mathbb{1}_{I_k=c_+})^2$$

counts all changes in speed.

s.2.14.109. The time T_1 to move from slow to fast satisfies $(a-c_-)T_1 = M-m$, and the time to move from fast to slow satisfies $(c_+-a)T_2 = M-m$. The total cycle time is T_1+T_2.

s.2.14.110. A machine that needs to be warm/hot to produce. When there are no jobs, it is better to switch it off ($m=0$ case); only switch it on when the queue length in front of it is longer than M. If $m>0$, assume that the machine can work at two speeds, but the higher speed requires more power.

s.2.14.111. For instance, let a_k be the number of arrivals in period k. Then take $a_1=1$, $a_2=a_3=0$, $a_4=a_5=1$, and then we have 3 period with no arrivals, and 3 periods with 1 arrivals and then 4 without and 4 with 1 arrival, and so on.

I accepted the following suggestion $A(t)=t^2$. However, formally the limit does exist, it is $\lim_{t\to\infty} t^2/t = \infty$.

s.2.14.112. By the PASTA property, arrivals of an $M/G/n$ queue see the time average stationary distribution. Thus, we can focus on the time-average probability that the system contains 3 jobs. This is $(1-\rho)\rho^3$. Here, $\rho = 3\cdot 10/60 = 1/2$. Clearly, $\rho<1$, hence the system is stable.

s.2.14.113. See the previous problem: pasta + stability. Just mentioning PASTA is ok.

s.2.14.114. See the book. Note that the rate from state 1 to 0 is 5, not 10.

s.2.14.115. see the book. The rate at which customers are accepted is $\sum_n \lambda_n p_n$. Multiply this by 10 to get the rate at which the system makes money. Note that this is not $10\,\mathsf{E}[L]$. Customers pay to get served, not to stay in the line...

s.2.14.116. $A(n,t)\approx D(n,t)$. Hence

$$\frac{A(t)}{t}\frac{A(n,t)}{A(t)} \approx \frac{D(n,t)}{D(t)}\frac{D(t)}{t}.$$

Now take the limit $t\to\infty$ to get $\lambda\pi(n)=\delta(n)\delta$.

Using $D(n,t)/Y(n,t)\cdot Y(n,t)/t$ is plain wrong (check the book). $-1/2$.

s.2.14.117. Rate stability, i.e., $\lambda=\delta$.

s.2.14.118. Sure, to derive (2.14.6), we only used counting arguments, but nothing in particular about the interarrival times.

Some students say no, because it is not true in case of batch arrivals. Ok. but recall, by assumption we are dealing here with $G/G/1$ queue...

s.2.14.119. No, for PASTA we need more.

$$\frac{A(t)}{t}\frac{A(n,t)}{A(t)} = \frac{A(n,t)}{Y(n,t)}\frac{Y(n,t)}{t}$$

Taking limits gives $\lambda\pi(n) = \lambda(n)p(n)$. When $\lambda(n) = \lambda$, arrivals see time averages. In particular, when the arrival process is Poisson, $\lambda(n) = \lambda$.

Some students seem to think that if you use PASTA to prove that $\pi(n) = \delta(n)$ (which is a bit convoluted), that this $\pi(n) = \delta(n)$ then implies PASTA. This, however, is a simple logical failure: note that $p \implies q$ is not the same as $q \implies p$. (A dog is an animal with four legs, an animal with four legs is not necessarily a dog...)

s.2.14.120.

$$\mathsf{E}\left[W_q\right] = \frac{1+C_a^2}{2}\frac{\rho}{1-\rho}\mathsf{E}[S] = \frac{\rho}{1-\rho}\mathsf{E}[S],$$

since for the $M/M/1$ queue, $C_a^2 = 1$. Next, $\mathsf{E}[W] = \mathsf{E}\left[W_q\right] + \mathsf{E}[S]$. Thus, with Little's law,

$$\mathsf{E}[L] = \lambda\,\mathsf{E}[W] = \frac{\rho^2}{1-\rho} + \rho.$$

The result now follows.

Not using the PK-formula: no point.

s.2.14.121. The level-crossing equations are like this:

$$\begin{aligned}\lambda p(0) &= 2\mu p(2)\\ \lambda p(1) &= 2\mu p(2) + \mu p(3)\\ \lambda p(2) &= \mu p(3).\end{aligned}$$

Since $\lambda = \mu$, $p(0) = 2p(2)$, $p(2) = p(3)$. But then, from the second equation: $p(1) = 3p(2)$. Finally, since $\sum p(i) = 1$, we get that $p(2)(2+3+1+1) = 7$. Hence $p(2) = 1/7$.

s.2.14.122. First use PASTA to see that $\pi(n) = p(n)$. Then, conclude that $\pi(3) = p(3) = 1/7$.

You need to mention that you used PASTA. It's not evident that $p(n) = \pi(n)$.

s.2.14.123. Now,

$$p(2) = \frac{2}{x}p(1) \qquad\qquad p(3) = xp(2) = \frac{x^2}{2}p(0)$$
$$p(1) = \frac{2}{x}p(2) + \frac{1}{x}p(3) = p(0)(1+\frac{x}{2}).$$

Hence

$$p(0)(1+1+\frac{x}{2}+\frac{x}{2}+\frac{x^2}{2}) = 1.$$

With this we have, for given x, found $p(0)$. Then we want x to be such that

$$p(3) = \frac{x^2/2}{2+x+x^2/2} > \alpha.$$

s.2.14.124. Yes, because we add an extra state that can be reached, while the transition rates between states $0,1,2$ and 3 do not change. Hence, since $p(4) > 0$, and $p(4) = p(3)$, the fraction of time spent in state 3 must be smaller in the presence of a state 4 than without this extra state.

In simple terms, if there is a waiting room with 3 seats, and you add an extra seat, than the fraction of people that have to stand becomes less.

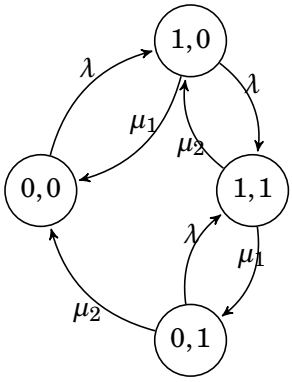
s.2.14.125. Jobs arrive at rate λ. The fraction accepted is $(1-\beta)$. Hence, the rate at which jobs enter is $\lambda(1-\beta)$. The average time jobs stay in the system is 1
mu. Thus, by Little's law, the average number of jobs in the system $\rho = \lambda(1-\beta)/\mu$.

s.2.14.126. The service times are not i.i.d., they depend on the number of jobs in the system, in other words, the service rate in state 2 is 2μ, while in state 1 it is 0 and in state 3 it is μ. It's simply not a $G/G/n/K$ queue, even though there is loss.

Interestingly, quite a number of students mess up the concepts of $G/G/1$ and so on. Note that the the $M/M/1$ queue is a special case of the $G/G/1$ queue, and this in turn is a special case of the $G/G/n$ queue. Check out the definitions in the book.

In the $G/G/n/K$ queue, it is not true that $\rho = \lambda/\mu$. This would imply with (2.14.7) that $\beta = 0$. But this ridiculous: the fraction of lost customers is not 0 in the $G/G/n/K$ queue in general, while (2.14.7) holds for any such queueing system.

s.2.14.127. $(0,0)$ both servers are empty, $(1,0)$ first server is busy, $(0,1)$ second server is busy, $(1,1)$ both servers are busy. As no jobs can be in queue, these are all possible states.



s.2.14.128.

There cannot be an arrow from $(1,1)$ to $(0,0)$, nor from $(0,0)$ to $(0,1)$ (Besides that this is not in line with the problem specification, why would you join the slow server if the fast is free?)

s.2.14.129. For ease, call $p(0,0) = a$, $p(1,0) = b$, $p(1,1) = c$, $p(0,1) = d$. Balance equations:

$$\begin{aligned}
\lambda a &= \mu_1 b + \mu_2 d \\
(\lambda + \mu_1) b &= \lambda a + \mu_2 c \\
(\mu_1 + \mu_2) c &= \lambda b + \lambda d \\
(\lambda + \mu_2) d &= \mu_1 c.
\end{aligned}$$

Thus, from (4) we have d as function of c. With (3) we get c as function of b. With (2) we get b as function of a. Then normalize to get a.

This is NOT a Jackson network...

s.2.14.130. Jobs arrive at rate λ. By PASTA $\lambda p(1,1)$ is the rate at which jobs are lost. Hence, the loss fraction is $p(1,1)$. Note, that $\lambda p(1,1) \neq p(1,1)$, but somehow, many students give the answer $\lambda p(1,1)$. Even by checking the units (number per unit time versus just number) it is apparent that $\lambda p(1,1)$ must be wrong. You need to mention that you used PASTA. It's not evident that it is $p(1,1)$.

s.2.14.131. If $\mu_2 = \mu_1$ and the queue can contain no job, then the above system reduces to the $M/M/2/2$ queue. It is not the $M/M/1$ or $M/M/2$ or $M/M/1/2$ queue.

s.2.14.132. Add a state (1) to the right of $(1,1)$. There is an arrow λ from $(1,1)$ to (1), and an arrow $\mu_1 + \mu_2$ in the other direction.

Some students made two queues, one for each server. But why would you wait for a server when another server is free?

Also, this is not an $M/M/2/3$ queue, because the servers are not identical.

s.2.14.133. $p(1) = p(1,1)\lambda/(\mu_1 + \mu_2)$.

s.2.14.134. $C_a^2 = 1$. $c = 2$. However, $C_s^2 \neq 1$ (you should realize this as soon as you read the problem.). As an approximation

$$\mathsf{E}[S] = \frac{1}{3}\frac{1}{3} + \frac{2}{3}\frac{1}{6},$$
$$\mathsf{E}\left[S^2\right] = \frac{1}{3}\frac{1}{3}^2 + \frac{2}{3}\frac{1}{6}^2.$$

This gives $\mathsf{V}[S]$, and then C_s^2.

The real problem is more interesting than this. The fraction of orders served by the fast server should depend on λ. Think about this.

Some students think that $\mathsf{E}[S] = 1/(3+6)$.

s.2.14.135. See the section in the book.

Some students say that $A(m,n,t) \approx D(n,t)$. Others say that $\lambda G(n-m)p(m)$ is the rate at which state $n+1$ is entered from state m. Why is this wrong? Yet others just say that up-crossings are down-crossings, and leave it at that.

s.2.14.136. $B = 1$, hence $\mathsf{E}[B] = \mathsf{E}\left[B^2\right] = 1$. Substitute and simplify.

Some students show that $\mathsf{E}[L] = \sum_n np(n) = \rho/(1-\rho)$, but that is not the answer to the question.

s.2.14.137. Formulas of general model should reduce to formulas for special cases of the general model. If not, the formula for the general model is wrong.

Some students say that such checks prove that the general formula is correct. In other words, they reverse the logic. Others say that it is important to 'connect the systems'. This is of course not the answer to the question.

3

APPROXIMATE MODELS

In this chapter we first consider the very useful formula of Sakasegawa to approximate the average waiting time in queue for the $G/G/c$ queue. We then illustrate how to use this formula to estimate waiting time in three examples in which the service process is interrupted. In the first case the server has to produce jobs from different families, and there is a change-over time required to switch from one production family to another. As such setups reduce the time the server has available to serve jobs, the load must increase. In fact, to reduce the load, the server produces in batches of fixed sizes. In the second case, the server sometimes requires small adjustments, for instance, to prevent the production quality to degrade below a certain level. Clearly, such adjustments are typically not required during a job's service; however, they can occur at arbitrary moments in time. Thus, this is different from batch production in which the batch sizes are constant. In the third example, quality problems or break downs can occur during a job's service. For each case we develop a model to analyze the influence of the interruptions on average job sojourn times.

3.1 $G/G/c$ QUEUE: APPROXIMATIONS

Theory and Exercises

In manufacturing settings it is quite often the case that the arrival process at a station is not a Poisson process. For instance, if processing times at a station are nearly constant, and the jobs of this station are sent to a second station for further processing, the inter-arrival times at the second station must be more or less equal too. Hence, in this case, the SCV of the arrivals at the second station $C_{a,2}^2$ is most probably smaller than 1. As a second, trivial case, if the inter-arrival times of jobs are 1 hour always and service times 59 minutes always, there simply cannot be a queue. Thus, the $M/G/1$ waiting time formula should not be naively applied to approximate the average waiting time of the $G/G/1$ queue.

While there is no closed-form expression available to compute the expected waiting time in queue for the $G/G/c$ queue, Sakasegawa's formula provides a reasonable approximation. This takes the form

$$\mathsf{E}\left[W_Q\right] = \frac{C_a^2 + C_s^2}{2} \frac{\rho^{\sqrt{2(c+1)}-1}}{c(1-\rho)} \frac{\mathsf{E}[S]}{c}, \tag{3.1.1}$$

where

$$\rho = \frac{\lambda \mathsf{E}[S]}{c}$$

is the load of the station (but not of the individual machines) and $C_a^2 = \mathsf{V}[X]/(\mathsf{E}[X])^2$ is the SCV of the inter-arrival times. This formula is reasonably accurate; for related expressions we refer to Bolch et al. [2006] and Hall [1991].

It is crucial to memorize the *scaling* relations that can be obtained from the $G/G/c$ waiting time formula. Even though the above results are only approximations, they prove to be exceedingly useful when designing queueing systems and analyzing the effect of certain changes, in particular changes in capacity, variability and service times. Roughly:

1. $\mathsf{E}[W_Q] \sim (1-\rho)^{-1}$. The consequence is that the waiting time increases *very steeply* when ρ is large. Hence, the waiting time is very sensitive to the actual value of ρ when ρ is large.

2. $\mathsf{E}[W_Q] \sim C_a^2$ and $\mathsf{E}[W_Q] \sim C_s^2$. Hence, reductions of the variation of the inter-arrival and service times do affect the waiting time, but only linearly.

3. $\mathsf{E}[W_Q] \sim \mathsf{E}[S]/c$. Thus, working in smaller job sizes reduces the waiting time as well. The average queue length does not decrease by working with smaller batches, but jobs are more 'uniformly spread' over the queue. This effect lies behind the idea of 'lot-splitting', i.e., rather than process large jobs, split jobs into multiple small jobs (assuming that setup times are negligible), so that the waiting time per job can be reduced.

These insights prove very useful when trying to reduce waiting times in any practical situation. First try to reduce the load (by blocking demand or increasing the capacity), then try to reduce the variability (e.g., by planning the arrival times of jobs), and finally, attempt to split jobs into multiple smaller jobs and use the resulting freedom to reschedule jobs in the queue.

3.1.1. *Show that the approximation* (3.1.1) *reduces to the result known for the M/M/*1 *and M/G/*1 *queues.*

3.1.2. *Is* (2.10.2) *also valid for the G/G/*1 *queue? Why (not)?*

3.1.3. *Consider a queue with c servers, with generally distributed inter-arrival times, generally distributed service times, and the system can contain at most K customers, i.e., the G/G/c/K queue. Let λ be the arrival rate, μ the service rate, β the long-run fraction of customers lost, and ρ the average number of busy/occupied servers. Show that*

$$\beta = 1 - \rho\frac{\mu}{\lambda}.$$

3.1.4. *Consider a single-server queue at which every minute a customer arrives, precisely at the first second. Each customer requires precisely* 50 *seconds of service. What are ρ, $\mathsf{E}[L]$, C_a^2, and C_S^2?*

3.1.5. *Consider the same single-server system as in 3.1.5, but now the customer service time is stochastic: with probability* 1/2 *a customer requires* 1 *minute and* 20 *seconds of service, and with probability* 1/2 *the customer requires only* 20 *seconds of service. What are ρ, C_a^2, and C_S^2? It is crucial to remember from the above exercises that knowledge of the utilization is not sufficient to characterize the average queue length.*

3.1.6. *For the G/G/*1 *queue, prove that the fraction of jobs that see n jobs in the system is the same as the fraction of departures that leave n jobs behind. What condition have you used to prove this?*

3.1.7. *(Hall 5.19) When a bus reaches the end of its line, it undergoes a series of inspections. The entire inspection takes 5 minutes on average, with a standard deviation of 2 minutes. Buses arrive with inter-arrival times uniformly distributed on* [3,9] *minutes.*

*As a first case, assuming a single server, estimate $\mathsf{E}[W_Q]$ with the G/G/*1 *waiting time formula. As a second case, compare this result to an M/G/*1 *system with arrival rate 10 per hour and the same service time distribution. Explain why your previous answer is smaller.*

Clearly, Sakasegawa's equation requires an estimate of the SCV C_a^2 of the inter-arrival times and the SCV C_s^2 of the service times. Now it is not always easy in practice to determine the actual service time distribution, one reason being that service times are often only estimated by a planner, but not actually measured. Similarly, the actual arrival moments of jobs are often not registered, mostly just the date or the hour, perhaps, that a customer arrived. Hence, it is often not possible to estimate C_a^2 and C_s^2 from the information that is available. However, when for instance the number of arrivals per day has been logged for some time so that we know $\{a_n, n = 1,\ldots,N\}$ for some N, we can use this information instead of the inter-arrival times $\{X_k\}$ to obtain insight into C_a^2. The relation we present here to compute C_a^2 from $\{a_n\}$ can of course also be applied to estimate C_s^2.

Theorem 3.1.1. The SCV of the inter-arrival times can be estimated with the formula

$$C_a^2 \approx \frac{\tilde{\sigma}^2}{\tilde{\lambda}},$$

where

$$\tilde{\lambda} = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} a_i, \quad \tilde{\sigma}^2 = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} a_i^2 - \tilde{\lambda}^2.$$

In words, $\tilde{\lambda}$ is the average number of arrivals per period, e.g., per day, and $\tilde{\sigma}^2$ is the variance of the number of arrivals per period.

Proof. The proof is based on an argument in Cox [1962]. We use quite a bit of the notation developed in Section 2.1. Let $\{A(t), t \geq 0\}$ be the number of arrivals that occur up to (and including) time t. We assume that $\{A(t)\}$ is a renewal process such that the inter-arrival times $\{X_k, k = 1,2,\ldots\}$ with $X_k = A_k - A_{k-1}$, are i.i.d. with mean $1/\lambda$ and standard deviation σ. (Observe that σ is not the same as $\tilde{\sigma}$ above.) Note that C_a^2 is defined in terms of λ and σ as:

$$C_a^2 = \frac{\mathsf{V}[X_i]}{(\mathsf{E}[X_i])^2} = \frac{\sigma^2}{1/\lambda^2} = \lambda^2\sigma^2.$$

Next, let A_k be the arrival time of the kth arrival. The following useful relation between $A(t)$ and A_k enables us to prove our result (recall that we used a similar relation in the derivation of the Poisson process):

$$\mathsf{P}(A(t) < k) = \mathsf{P}(A_k > t).$$

Since the inter-arrival times have finite mean and second moment by assumption, we can apply the central limit law to obtain that, as $k \to \infty$,

$$\frac{A_k - k/\lambda}{\sigma\sqrt{k}} \to N(0,1),$$

where $N(0,1)$ is a standard normal random variable with distribution $\Phi(\cdot)$. Similarly,

$$\frac{A(t) - \lambda t}{\alpha\sqrt{t}} \to N(0,1)$$

for an α that is yet to be determined. Thus, $\mathsf{E}[A(t)] = \lambda t$ and $\mathsf{V}[A(t)] = \alpha^2 t$.

Using that $\mathsf{P}(N(0,1) \leq y) = \mathsf{P}(N(0,1) > -y)$ (and $\mathsf{P}(N(0,1) = y) = 0$) we have that

$$\Phi(y) \approx \mathsf{P}\left(\frac{A_k - k/\lambda}{\sigma\sqrt{k}} \leq y\right)$$

$$= \mathsf{P}\left(\frac{A_k - k/\lambda}{\sigma\sqrt{k}} > -y\right)$$
$$= \mathsf{P}\left(A_k > \frac{k}{\lambda} - y\sigma\sqrt{k}\right).$$

Define for ease

$$t_k = \frac{k}{\lambda} - y\sigma\sqrt{k}.$$

We can use the above relation between the distributions of $A(t)$ and A_k to see that $\mathsf{P}(A_k > t_k) = \mathsf{P}(A(t_k) < k)$. With this we get,

$$\begin{aligned}\Phi(y) &\approx \mathsf{P}(A_k > t_k)\\ &= \mathsf{P}(A(t_k) < k)\\ &= \mathsf{P}\left(\frac{A(t_k) - \lambda t_k}{\alpha\sqrt{t_k}} < \frac{k - \lambda t_k}{\alpha\sqrt{t_k}}\right).\end{aligned}$$

Since $(A(t_k) - \lambda t_k)/\alpha\sqrt{t_k} \to N(0,1)$ as $t_k \to \infty$, the above implies that

$$\frac{k - \lambda t_k}{\alpha\sqrt{t_k}} \to y,$$

as $t_k \to \infty$. Using the above definition of t_k, the left-hand side of this equation can be written as

$$\frac{k - \lambda t_k}{\alpha\sqrt{t_k}} = \frac{\lambda\sigma\sqrt{k}}{\alpha\sqrt{k/\lambda + \sigma\sqrt{k}}}y.$$

Since $t_k \to \infty$ is implied by (and implies) $k \to \infty$, we therefore want that α is such that

$$\frac{\lambda\sigma\sqrt{k}}{\alpha\sqrt{k/\lambda + \sigma\sqrt{k}}}y \to y,$$

as $k \to \infty$. This is precisely the case when

$$\alpha = \lambda^{3/2}\sigma.$$

Finally, for t large (or, by the same token k large),

$$\frac{\sigma_k^2}{\lambda_k} = \frac{\mathsf{V}[A(t)]}{\mathsf{E}[A(t)]} \approx \frac{\alpha^2 t}{\lambda t} = \frac{\alpha^2}{\lambda} = \frac{\lambda^3\sigma^2}{\lambda} = \lambda^2\sigma^2 = C_a^2,$$

where the last equation follows from the above definition of C_a^2. The proof is complete. □

Hints

h.3.1.1. Take $c = 1$.

Solutions

s.3.1.1. If $c = 1$, as is the case for the $M/M/1$ queue, $\sqrt{2(c+1)} - 1 = 2 - 1 = 1$, so that (3.1.1) reduces to $\mathsf{E}[W_Q] = \rho/(1-\rho)\mathsf{E}[S]$. Recall that $C_a^2 = C_s^2 = 1$ for the $M/M/1$.

s.3.1.2. Not necessarily. Since jobs do not arrive in general as a Poisson process, we cannot use the PASTA property to conclude that the time average queue length $\mathsf{E}\left[L_Q\right]$ is the same as the average queue length as observed by customers. (Can you provide an example that shows this difference? Hint, we did this earlier.)

s.3.1.3. This follows from the observation that $\lambda(1-\beta)$ is the net arrival rate, as jobs are lost at a rate $\lambda\beta$. Hence, the load must be $\lambda(1-\beta)/\mu$. As the load must be equal to ρ, it follows that $\rho = \lambda(1-\beta)/\mu$, from which the other result immediately follows.

s.3.1.4. $\rho = \lambda\,\mathsf{E}[S] = 1/60 \cdot 50 = 5/6$. Since job arrivals do not overlap any job service, the number of jobs in the system is 1 for 50 seconds, then the server is idle for 10 seconds, and so on. Thus $\mathsf{E}[L] = 1 \cdot 5/6 = 5/6$. There is no variance in the inter-arrival times, and also not in the service times, thus $C_a^2 = C_s^2 = 0$. Also $\mathsf{E}\left[W_Q\right] = 0$ since $\mathsf{E}\left[L_Q\right] = 0$. Interestingly, we get the same from Kingman's formula.

s.3.1.5. Again $\mathsf{E}[S]$ is 50 seconds, so that $\rho = 5/6$. Also $C_a^2 = 0$. For the C_S^2 we have to do some work.

$$\begin{aligned}
\mathsf{E}[S] &= \frac{20}{2} + \frac{80}{2} = 50 \\
\mathsf{E}\left[S^2\right] &= \frac{400}{2} + \frac{6400}{2} = 3400 \\
\mathsf{V}[S] &= \mathsf{E}[S]^2 - (\mathsf{E}[S])^2 = 3400 - 2500 = 900 \\
C_S^2 &= \frac{\mathsf{V}[S]}{(\mathsf{E}[S])^2} = \frac{900}{2500} = \frac{9}{25}.
\end{aligned}$$

s.3.1.6. All follows straightaway from the definitions in the main text. In the $G/G/1$ queue jobs arrive and depart in single units. Hence, $|A(n,t) - D(n,t)| \leq 1$, cf., (2.4.1a). Then,

$$\frac{A(t)}{t}\frac{A(n,t)}{A(t)} \approx \frac{D(t)}{t}\frac{D(n,t)}{D(t)}.$$

The left-hand side goes to $\lambda\pi(n)$ as $t \to \infty$, and the right-hand side to $\delta\delta(n)$. Use the fact that we always assume, implicitly, that the system is stable, so that $\lambda = \delta$. As a consequence $\delta(n) = \pi(n)$.

s.3.1.7. First the $G/G/1$ case. Observe that in this case, the inter-arrival time $X \sim U[3,9]$, that is, never smaller than 3 minutes, and never longer than 9 minutes.

```
>>> a = 3.
>>> b = 9.
>>> EX = (b+a)/2. # expected inter-arrival time
>>> EX
6.0
>>> labda = 1./EX # per minute
>>> labda
0.16666666666666666
>>> VA = (b-a)*(b-a)/12.
>>> CA2 = VA/(EX*EX)
>>> CA2
0.08333333333333333
```

```
>>> ES = 5.
>>> sigma = 2
>>> VS = sigma*sigma
>>> CS2 = VS/(ES*ES)
>>> CS2
0.16

>>> rho = labda*ES
>>> rho
0.8333333333333333

>>> Wq = (CA2+CS2)/2. * rho/(1.-rho) * ES
>>> Wq
3.041666666666665
```

Now the $M/G/1$ case.

```
>>> Wq = (1.+CS2)/2. * rho/(1.-rho) * ES
>>> Wq
14.499999999999993
```

The arrival process with uniform inter-arrival times is much more regular than a Poisson process. In the first case, bus arrivals are spaced in time at least with 3 minutes.

3.2 SETUPS AND BATCH PROCESSING

Theory and Exercises

In this section, we focus on the effect of change-over, or setup, times on the average sojourn time of jobs. Consider, for instance, a machine that paints red and blue bikes. When the machine requires a color change, a clean-up time is necessary. As we will see it is necessary in such situations to produce in batches. Other examples are ovens that need warm-up or cool-down times when different item types require different production temperatures. Yet another example can be seen in service operations settings: when servers have to move from one part of a building to another, the time spent moving cannot be spent on serving customers. Thus, the servers have to process jobs (or customers) in batches in one building, then move to another (part of the) building, serve a batch, and so on.

In this section, we first make a model and provide a list of elements required to compute the expected sojourn time of an item. Then we illustrate how to use these elements in a concrete case. Finally, in an exercise we ask you to derive the formulas yourself.

Specifically, we analyze the following queueing situation. There are two job families, e.g., red and blue, each served by the same single server. Jobs arrive at rate λ_r and λ_b, respectively, so that the arrival rate of jobs is $\lambda = \lambda_b + \lambda_r$. The setup times $\{R_i\}$ form an i.i.d. sequence with common random variable R and are independent of S_0.

The main idea is to convert the effects of the setup times into job service times and then use the $G/G/1$ waiting time formula. For this reason we define the *net processing time* S_0 of a

job as the regular processing time without setups, and we define the *effective processing time* S as the time the server is occupied with processing a job including a potential setup. We assume that S_0 has the same distribution for all job families. A bit of thought. cf. 3.2.4, will reveal that

$$\mathsf{E}[S] = \mathsf{E}[S_0] + \frac{\mathsf{E}[R]}{B}. \tag{3.2.1}$$

With this the load takes the form

$$\rho = \lambda \left(\mathsf{E}[S_0] + \frac{\mathsf{E}[R]}{B} \right).$$

We next consider all steps that make up a sojourn time. First, jobs of each color are assembled into batches of size B, which we assume to be the same for both colors. Once a batch is complete, the batch enters a queue (of batches). After some time the batch reaches the head of the queue. Then the machine performs a setup, and starts processing each job individually until the batch is complete. Finally, once a job is finished, it can leave the system; as a consequence, it does not have to wait for other jobs in the same batch to finish.

The average time needed to form (or assemble) a batch is

$$\mathsf{E}[W_r] = \frac{B-1}{2\lambda_r}. \tag{3.2.2}$$

To apply Sakasegawa's formula to the queue with batches, observe that the service time of a batch is $\mathsf{E}[R] + B\,\mathsf{E}[S_0]$. The load is given above. The SCV of the inter-arrival times of the batches is given by

$$C^2_{a,B} = \frac{C^2_a}{B}, \tag{3.2.3}$$

recall that jobs are first assembled into batches, and then these batches are sent to the queue. The SCV $C^2_{s,B}$ of the service times of the batches is

$$C^2_{s,B} = \frac{B\,\mathsf{V}[S_0] + \mathsf{V}[R]}{(B\,\mathsf{E}[S_0] + \mathsf{E}[R])^2}. \tag{3.2.4}$$

With these elements we can compute the average time a batch spends in queue.

Finally, after a batch is taken into service, we need a rule to determine when the job can leave the system. If the job has to wait until all jobs in the batch are served, the time a job spends at the server is $\mathsf{E}[R] + B\,\mathsf{E}[S_0]$. Here we assume that jobs can leave right after being served, consequently,

$$\mathsf{E}[R] + \frac{B+1}{2}\,\mathsf{E}[S_0]. \tag{3.2.5}$$

3.2.1. *Show that the requirement $\rho < 1$ leads to the following constraint on the minimal batch size B*

$$B > \frac{\lambda\,\mathsf{E}[R]}{1 - \lambda\,\mathsf{E}[S_0]}.$$

3.2.2. *What important insights do the above formulas provide about setting proper batch sizes?*

3.2.3. *Jobs arrive at $\lambda = 3$ per hour at a machine with $C^2_a = 1$; service times are exponential with an average of 15 minutes. Assume $\lambda_r = 0.5$ per hour, hence $\lambda_b = 3 - 0.5 = 2.5$ per hour. Between any two batches, the machine requires a cleanup of 2 hours, with a standard deviation of 1 hour, during which it is unavailable for service. What is the smallest batch size that can be allowed?*

What is the average time a red job spends in the system in case $B = 30$ jobs? Finally, observe that there is B that minimizes the average sojourn time.

3.2.4. *Derive* (3.2.1)–(3.2.5).

3.2.5. *Derive* (3.2.2).

3.2.6. *Derive* (3.2.1).

3.2.7. *Explain that the load is*

$$\rho = \lambda_B(B\,\mathsf{E}[S_0] + \mathsf{E}[R]),$$

where λ_B *is the arrival rate of batches.*

3.2.8. *Derive* (3.2.3).

3.2.9. *Derive* (3.2.4).

3.2.10. *Derive* (3.2.5).

Hints

h.3.2.5. Show that the total time to form a red batch is $(B-1)/\lambda_r$.

h.3.2.6. What fraction of the setup time $\mathsf{E}[R]$ 'belongs' to one job?

h.3.2.9. What is the variance of a batch service time?

Solutions

s.3.2.1. We require that the load is less than one, in other words, that $\lambda(\mathsf{E}[S_0] + \mathsf{E}[R]/B) < 1$. This is equivalent to $\mathsf{E}[R]/B < 1/\lambda - \mathsf{E}[S_0]$, hence $B > \mathsf{E}[R]/(1/\lambda - \mathsf{E}[S_0])$. Multiplying with λ gives the result.

s.3.2.2. Overall, batch sizes need to be tuned to minimize average sojourn times. When the batch sizes are small, the load ρ is near to one (in other words, the server spends a relatively large fraction of its time on setups), so that the queueing times are long, but the times to form a batch are small, as the times to form and process batches are linear functions of the batch size B. If, however, the batch sizes are large, the queueing times will be relatively short, but the times to form and unpack batches will be large.

s.3.2.3. First check the load.

```
>>> labda = 3 # per hour
>>> ES0 = 15./60 # hour
>>> ES0
0.25
>>> ER = 2.
>>> B = 30
>>> ESe = ES0+ ER/B
>>> ESe
0.31666666666666665

>>> rho = labda*ESe
>>> rho
0.95
```

Evidently, the load is smaller than 1.

The minimal batch size is

```
>>> Bmin = labda*ER/(1-labda*ES0)
>>> Bmin
24.0
```

So, with $B = 30$ we are on the safe side.

The time to form a red batch is

```
>>> labda_r = 0.5
>>> EWf = (B-1)/(2*labda_r)
>>> EWf # in hours
29.0
```

Now the time a batch spends in queue

```
>>> Cae = 1.
>>> CaB = Cae/B
>>> CaB
0.03333333333333333
>>> Ce = 1 # SCV of service times
>>> VS0 = Ce*ES0*ES0
>>> VS0
0.0625
>>> VR = 1*1. # Var setups is sigma squared
>>> VSe = B*VS0 + VR
>>> VSe
2.875
>>> ESb = B*ES0+ER
>>> ESb
9.5
>>> CeB = VSe/(ESb*ESb)
>>> CeB
0.03185595567867036
>>> EWq = (CaB+CeB)/2 * rho/(1-rho) * ESb
>>> EWq
5.8833333333333275
```

The time to unpack the batch, i.e., the time at the server.

```
>>> ES = ER + (B-1)/2 * ES0 + ES0
>>> ES
5.875
```

The overall time red jobs spend in the system.

```
>>> total = EWf + EWq + ES
>>> total
40.758333333333326
```

s.3.2.4. See 3.2.5 to 3.2.4.

s.3.2.5. Suppose a batch is just finished. The first job of a new batch needs to wait, on average, $B-1$ inter-arrival times until the batch is complete, the second $B-2$ inter-arrival times, and so on. The last job does not have to wait at all. Thus, the total time to form a batch is $(B-1)/\lambda_r$.

An arbitrary job can be anywhere in the batch, hence the average time a job must wait until the batch is complete is half the total time.

s.3.2.6. The total service time spent on a batch of size B is $B\,\mathsf{E}[S_0]+\mathsf{E}[R]$. The effective time per job is then the average, i.e., $(B\,\mathsf{E}[S_0]+\mathsf{E}[R])/B$.

s.3.2.7. It is evident that the rate at which batches arrive is

$$\lambda_B = \frac{\lambda}{B},$$

since both job colors have the same batch size. Then the equality has the interpretation of the batch arrival rate times the work per batch.

s.3.2.8. The variance of the inter-arrival time of batches is B times the variance of job inter-arrival times. The inter-arrival times of batches is also B times the inter-arrival times of jobs. Thus,

$$C^2_{a,B} = \frac{B\,\mathsf{V}[X]}{(B\,\mathsf{E}[X])^2} = \frac{\mathsf{V}[X]}{(\mathsf{E}[X])^2}\frac{1}{B} = \frac{C_a^2}{B}.$$

s.3.2.9. The variance of a batch is $\mathsf{V}\left[\sum_{i=1}^{B} S_{0,i}+R\right] = B\,\mathsf{V}[S_0]+\mathsf{V}[R]$, since the normal service times $S_{0,i}, i=1,\ldots,B$ of the job are independent, and also independent of the setup time R of the batch.

s.3.2.10. First, wait until the setup is finished, then wait (on average) for half of the batch (minus the job itself) to be served, and then the job has to be served itself, that is, $\mathsf{E}[R]+\frac{B-1}{2}\mathsf{E}[S_0]+\mathsf{E}[S_0]$.

3.3 NON-PREEMPTIVE INTERRUPTIONS, SERVER ADJUSTMENTS

Theory and Exercises

In Section 3.2 we studied the effect of setup times between job batches with a fixed size B. However, other types of interruptions can occur, such as a machine requiring random adjustments that can occur between any two jobs. This type of outages is *non-preemptive* as the outages do not interrupt the processing of a job in service. In this section we develop a simple model to understand the impact of such outages on job sojourn times; we use the same notation as in Section 3.2 and follow the same line of reasoning.

We assume that adjustments $\{R_i\}$ occur geometrically distributed between any two jobs with a mean of B jobs between any two adjustments. Consequently, the probability of an outage between any two jobs is $p = 1/B$. Observe that geometrically distributed random variables satisfy the memoryless property in discrete time. Hence, our assumption implies that the occurrence of an adjustment between jobs i and $i+1$ has no effect on the probability that an adjustment is necessary between jobs $i+1$ and $i+2$.

With the model, we can obtain quantitative insights into the effects of reducing adjustment times or the variability of these adjustments times. For instance, we might decide to do fewer adjustments, so that p decreases, but at the expense of larger average outage times. Now we can analyze the consequences of such decisions without needing to actually do the experiments in real life.

Contrary to the batch processing case of Section 3.2, we now only need to find the mean and variance of the effective processing times. They are given by

$$\mathsf{E}[S] = \mathsf{E}[S_0] + \frac{\mathsf{E}[R]}{B}, \tag{3.3.1}$$

$$\mathsf{V}[S] = \mathsf{V}[S_0] + \frac{\mathsf{V}[R]}{B} + (B-1)\left(\frac{\mathsf{E}[R]}{B}\right)^2. \tag{3.3.2}$$

Thus, the effective server load including down-times is $\rho = \lambda\,\mathsf{E}[S]$, and we can compute SCV of the effective job processing times. We have now all elements to fill in the $G/G/1$ waiting time formula!

3.3.1. *A machine requires an adjustment with an average of* 5 *hours and a standard deviation of* 2 *hours. Jobs arrive as a Poisson process with rate* $\lambda = 9$ *per working day. The machine works two* 8 *hour shifts a day. Work not processed on a day is carried over to the next day. Job service times are 1.5 hours, on average, with standard deviation of* 0.5 *hours. Interruptions occur on average between* 30 *jobs. Compute the average waiting time in queue.*

3.3.2. *Show that the average effective processing time is given by* (3.3.1).

3.3.3. *Show that*

$$\mathsf{E}\left[S^2\right] = \mathsf{E}\left[S_0^2\right] + 2\frac{\mathsf{E}[S_0]\,\mathsf{E}[R]}{B} + \frac{\mathsf{E}\left[R^2\right]}{B}.$$

3.3.4. *Derive* $V(S)$ *in* (3.3.2).

Hints

h.3.3.1. Get the units right. First compute the load, and then compute the rest.

h.3.3.2. If there is no failure, the service time is S_0. If there is a failure, the service of a job is $R + S_0$, since we add the outage time to the service time of the job.

h.3.3.4. First compute $\mathsf{E}\left[S^2\right]$. See 3.3.3.

Solutions

s.3.3.1. First we determine the load.

```
>>> B=30
>>> ES0 = 1.5
>>> labda = 9./(2*8) # arrival rate per hour
>>> ER=5.
>>> ESe=ES0+ER/B
>>> ESe
```

```
1.6666666666666667
>>> rho = labda*ESe
>>> rho
0.9375
```

So, at least the system is stable.

```
>>> VS0 = 0.5*0.5
>>> VR = 2.*2.
>>> VSe = VS0 + VR/B + (B-1)*(ER/B)**2
>>> VSe
1.1888888888888887
>>> Ce2 = VSe/(ESe*ESe)
>>> Ce2
0.4279999999999999
```

And now we can fill in the waiting time formula.

```
>>> Ca2=1 # Poisson arrivals
>>> EW = (Ca2+Ce2)/2 * rho/(1-rho) * ESe
>>> EW
17.849999999999998
```

s.3.3.2.

$$\mathsf{E}[S] = (1-p)\,\mathsf{E}[S_0] + p(\mathsf{E}[R] + \mathsf{E}[S_0]) = \mathsf{E}[S_0]\frac{B-1}{B} + (\mathsf{E}[S_0] + \mathsf{E}[R])\frac{1}{B},$$

since $p = 1/B$.

s.3.3.3.

$$\begin{aligned}\mathsf{E}\left[S^2\right] &= (1-p)\,\mathsf{E}\left[S_0^2\right] + p\,\mathsf{E}\left[(S_0+R)^2\right] \\ &= (1-p)\,\mathsf{E}\left[S_0^2\right] + p\,\mathsf{E}\left[S_0^2\right] + 2p\,\mathsf{E}[S_0]\,\mathsf{E}[R] + p\,\mathsf{E}\left[R^2\right].\end{aligned}$$

Simplify and substitute $p = 1/B$.

s.3.3.4.

$$\begin{aligned}\mathsf{V}[S] &= \mathsf{E}\left[S^2\right] - (\mathsf{E}[S])^2 \\ &= \mathsf{E}\left[S_0^2\right] + 2\,\mathsf{E}[S_0]\,\mathsf{E}[R]\frac{1}{B} + \mathsf{E}\left[R^2\right]\frac{1}{B} \\ &\quad - (\mathsf{E}[S_0])^2 - 2\,\mathsf{E}[S_0]\,\mathsf{E}[R]\frac{1}{B} - (\mathsf{E}[R])^2\frac{1}{B^2} \\ &= \mathsf{V}[S_0] + ((\mathsf{E}\left[R^2\right] - (\mathsf{E}[R])^2)\frac{1}{B} + (\mathsf{E}[R])^2\left(\frac{1}{B} - \frac{1}{B^2}\right).\end{aligned}$$

3.4 PREEMPTIVE INTERRUPTIONS, SERVER FAILURES

Theory and Exercises

In Sections 3.2 and 3.3 we assumed that servers are never interrupted while serving a job. However, in many situations this assumption is not satisfied: a person might receive a short

phone call while working on a job, a machine may fail in the midst of processing, and so on. In this section, we develop a model to compute the influence on the mean waiting time of such *preemptive outages*, i.e., interruptions that occur *during* a service. As in Section 3.3, to use the $G/G/1$ waiting time formula it suffices to find expressions for $\mathsf{E}[S]$ and $\mathsf{V}[S]$. Thus, this will be our task for the rest of the section. We remark in passing that the results and the derivation are of general interest.

Supposing that N interruptions occur during the net service time S_0 of a job, the effective service time will be

$$S = S_0 + \sum_{i=1}^{N} R_i.$$

A common assumption is that the time between two interruptions is $\mathrm{Exp}(\lambda_f)$, hence is memoryless. Consequently, the number of interruptions N that occur during the net service time S_0 is Poisson distributed with mean $\mathsf{E}[N] = \lambda_f \mathsf{E}[S_0]$. Define the *availability* as

$$A = \frac{m_f}{m_f + m_r},$$

where m_f is the mean time to fail and m_r the mean time to repair. With this,

$$\mathsf{E}[S] = \frac{\mathsf{E}[S_0]}{A}. \tag{3.4.1}$$

Finally, by assuming that repair times are exponentially distributed with mean $\mathsf{E}[R]$, we can find that

$$C_s^2 = C_0^2 + 2A(1-A)\frac{\mathsf{E}[R]}{\mathsf{E}[S_0]}, \tag{3.4.2}$$

where C_0^2 is the SCV of S_0, i.e., the service time without interruptions. It is important to realize that

$$\rho = \lambda \mathsf{E}[S] = \lambda \frac{\mathsf{E}[S_0]}{A},$$

hence the server load increases due to failures.

Before deriving the above expressions, let us see how to apply them.

3.4.1. *Suppose we have a machine with memoryless failure behavior, with a mean-time-to-fail of 3 hours. Regular service times are deterministic with an average of 10 minutes, jobs arrive as a Poisson process with rate of 4 per hour. Repair times are exponential with a mean duration of 30 minutes. What is the average sojourn time?*

3.4.2. *Suppose we could buy another machine that never fails. What is the average sojourn time?*

3.4.3. *Provide an intuitive explanation for* $\mathsf{E}[S] = \mathsf{E}[S_0]/A$.

3.4.4. *Show that* $\mathsf{E}[S] = \mathsf{E}[S_0] + \mathsf{E}[R]\,\mathsf{E}[N]$*; then show* (3.4.1).

3.4.5. *Suppose that* $N = n$*, show that* $\mathsf{E}[S_n] = n\,\mathsf{E}[R]$.

3.4.6. *Use 3.4.5 to show that* $\mathsf{E}[S_N] = \mathsf{E}[R]\,\mathsf{E}[N]$*. This result is known as* Wald's equation.

3.4.7. *Show that for our model of interruptions,*

$$A = \frac{1}{1 + \lambda_f \mathsf{E}[R]}.$$

3.4.8. *Show that* $\mathsf{E}[S] = \mathsf{E}[S_0](1+\lambda_f \mathsf{E}[R])$. *Conclude that* $\mathsf{E}[S] = \mathsf{E}[S_0]/A$.

3.4.9. *The derivation of* C_s^2 *is a bit more involved. To see this, explain that* $\mathsf{V}[S] \neq \mathsf{V}[S_0] + \mathsf{V}\left[\sum_{i=0}^{N} R_i\right]$.

3.4.10. *Show that*

$$\mathsf{E}\left[S^2\right] = \mathsf{E}\left[S_0^2\right] + 2\mathsf{E}\left[S_0 \sum_{i=1}^{N} R_i\right] + \mathsf{E}\left[\sum_{i=1}^{N} R_i^2\right] + \mathsf{E}\left[\sum_{i=1}^{N} \sum_{j\neq i} R_i R_j\right].$$

3.4.11. *To simplify the result of 2.7.1, we assume at first that* S_0 *is known, so that the number of failures that occur during a service time* S_0 *is Poisson distributed, i.e.,* $N \sim P(\lambda_f S_0)$. *Show that* $\mathsf{E}\left[S_0 \sum_{i=1}^{N} R_i \,\middle|\, S_0\right] = \lambda_f S_0^2 \mathsf{E}[R]$.

3.4.12. *Now show that* $\mathsf{E}\left[\sum_{i=1}^{N} R_i^2 \,\middle|\, S_0\right] = \lambda_f S_0 \mathsf{E}\left[R^2\right]$.

3.4.13. *Show that* $\mathsf{E}\left[\sum_{i=1}^{N} \sum_{j\neq i} R_i R_j \,\middle|\, S_0\right] = \lambda_f^2 S_0^2 (\mathsf{E}[R])^2$.

3.4.14. *Combine the above to see that* $\mathsf{E}\left[S^2 \,\middle|\, S_0\right] = \frac{S_0^2}{A^2} + \lambda_f \mathsf{E}\left[R^2\right] S_0$. *From this,*

$$\mathsf{E}\left[S^2\right] = \frac{\mathsf{E}\left[S_0^2\right]}{A^2} + \lambda_f \mathsf{E}\left[R^2\right] \mathsf{E}[S_0].$$

3.4.15. *Show that*

$$\mathsf{V}[S] = \frac{\mathsf{V}[S_0]}{A^2} + \lambda_f \mathsf{E}\left[R^2\right] \mathsf{E}[S_0].$$

3.4.16. *Show that*

$$C_s^2 = \frac{\mathsf{V}[S]}{(\mathsf{E}[S])^2} = C_0^2 + \frac{\lambda_f \mathsf{E}\left[R^2\right] A^2}{\mathsf{E}[S_0]},$$

3.4.17. *With the above assumption on the distribution of* R, *show that*

$$C_s^2 = C_0^2 + 2A(1-A)\frac{\mathsf{E}[R]}{\mathsf{E}[S_0]}.$$

Hints

h.3.4.1. Mind to work in a consistent set of units, e.g., hours. It is easy to make mistakes.

h.3.4.5. Is it relevant for the expectation of S_n that $R_1, \ldots, R_n$ are mutually independent?

h.3.4.6. Use (1.1.2a).

h.3.4.7. Observe that $m_f = 1/\lambda_f$ and $m_r = \mathsf{E}[R]$.

h.3.4.8. Realize that $\mathsf{E}[N] = \lambda_f \mathsf{E}[S_0]$. Then use 3.4.7.

h.3.4.12. Use Wald's equation, which we derived in 3.4.6.

h.3.4.16. Just realize that $\mathsf{E}[S] = \mathsf{E}[S_0]/A$, and use the above.

Solutions

s.3.4.1. Let's first compute the load. If $\rho > 1$ we are in trouble.

```
>>> labda = 4.
>>> ES0 = 10./60 # in hours
>>> labda_f = 1./3
>>> ER = 30./60 # in hours
>>> A = 1./(1+labda_f*ER)
>>> A
0.8571428571428571
>>> ES = ES0/A
>>> ES
0.19444444444444445
>>> rho = labda*ES
>>> rho
0.7777777777777778
```

As $\rho < 1$, the system is not in overload. Now for the queueing time.

```
>>> Ca2 = 1.
>>> C02 = 0. # deterministic service times
>>> Ce2 = C02 + 2*A*(1-A)*ER/ES0
>>> Ce2
0.7346938775510207
>>> EW = (Ca2+Ce2)/2 * rho/(1-rho) * ES
>>> EW
0.5902777777777779
>>> EW + ES # sojourn time
0.7847222222222223
```

s.3.4.2. Now we don't need to take availability into account: the machine never fails so $A = 1$.

```
>>> labda = 4.
>>> ES0 = 10./60 # in hours
>>> A = 1
>>> ES = ES0/A
>>> rho = labda*ES
>>> rho
0.6666666666666666
>>> Ca2 = 1.
>>> C02 = 0. # deterministic service times
>>> EW = (Ca2+C02)/2 * rho/(1.-rho) * ES
>>> EW
0.16666666666666663
>>> EW + ES # sojourn time
0.33333333333333326
```

The average time in queue reduces from ≈ 0.6 to ≈ 0.17 hours, a reduction by about a factor 3.

s.3.4.3. An intuitive way to obtain this result is by noting that A is the fraction of time the server is working. As the total service time of a job is $\mathsf{E}[S]$, the net work done is $A\,\mathsf{E}[S]$. But this must be the time needed to do the real job, hence $A\,\mathsf{E}[S] = \mathsf{E}[S_0]$.

s.3.4.4. Write $S_N = \sum_{i=1}^{N} R_i$ for the total duration of the interruptions, so that the total job duration becomes $S = S_0 + S_N$.

s.3.4.5. The expectation of the sum of random variables is the same as the sum of the expectations; independence is irrelevant. Hence,

$$\mathsf{E}[S_n] = \mathsf{E}\left[\sum_{i=1}^{n} R_i\right] = n\,\mathsf{E}[R],$$

since by assumption $\mathsf{E}[R_1] = \cdots = \mathsf{E}[R_n]$.

s.3.4.6. Since $\mathsf{E}[S_n] = n\,\mathsf{E}[R]$,

$$\begin{aligned}\mathsf{E}\left[\sum_{i=1}^{N} R_i\right] &= \mathsf{E}\left[\sum_{n=0}^{\infty} \mathbb{1}_{N=n}\left(\sum_{i=1}^{n} R_i\right)\right] \\ &= \sum_{n=0}^{\infty} \mathsf{E}[\,\mathbb{1}_{N=n} n\,\mathsf{E}[R]] \\ &= \mathsf{E}[R]\sum_{n=0}^{\infty} n\,\mathsf{E}[\,\mathbb{1}_{N=n}] = \mathsf{E}[R]\sum_{n=0}^{\infty} n p_n \\ &= \mathsf{E}[R]\,\mathsf{E}[N].\end{aligned}$$

s.3.4.7. The time to fail is the time in between two interruptions. We assume that these times are $\mathrm{Exp}(1/\lambda_f)$. The duration of an interruption is R, which can be interpreted as the time to repair the server, hence $m_r = \mathsf{E}[R]$. With this

$$A = \frac{m_f}{m_f + m_r} = \frac{1/\lambda_f}{1/\lambda_f + \mathsf{E}[R]}.$$

s.3.4.8.

$$\mathsf{E}[S] = \mathsf{E}[S_0] + \mathsf{E}[N]\,\mathsf{E}[R] = \mathsf{E}[S_0] + \lambda_f\,\mathsf{E}[S_0]\,\mathsf{E}[R] = \mathsf{E}[S_0](1 + \lambda_f\,\mathsf{E}[R]).$$

s.3.4.9. Observe that S_0 and N are not independent. In fact, when $S_0 = s$, the number of failures N is Poisson distributed with mean $\lambda_f s$.

s.3.4.10. Just work out the square of $S_0 + \sum_{i=1}^{N} R_i$ and take expectations. Realize that $(\sum_i R_i)^2 = \sum_i R_i^2 + \sum_i \sum_{j\neq i} R_i R_j$.

s.3.4.11. $\mathsf{E}\left[S_0 \sum_{i=1}^{N} R_i \,\middle|\, S_0\right] = S_0\,\mathsf{E}\left[\sum_{i=1}^{N} R_i \,\middle|\, S_0\right] = S_0\,\mathsf{E}[R]\,\mathsf{E}[N] = \lambda_f\,\mathsf{E}[R] S_0^2$.

s.3.4.12.

$$\begin{aligned}\mathsf{E}\left[\sum_{i=1}^{N} R_i^2 \,\middle|\, S_0\right] &= \mathsf{E}\left[R^2\right]\mathsf{E}\left[\sum_{n=0}^{\infty} n\,\mathbb{1}_{N=n} \,\middle|\, S_0\right] \\ &= \mathsf{E}\left[R^2\right]\mathsf{E}[N \mid S_0] \\ &= \lambda_f S_0\,\mathsf{E}\left[R^2\right].\end{aligned}$$

s.3.4.13. Since the $\{R_i\}$ are i.i.d.,

$$\mathsf{E}\left[\sum_{i=1}^{N}\sum_{j\neq i}R_iR_j\,\middle|\,S_0\right] = \mathsf{E}[N(N-1)|S_0](\mathsf{E}[R])^2 = (\mathsf{E}\left[N^2|S_0\right] - \mathsf{E}[N\,|\,S_0])(\mathsf{E}[R])^2.$$

Now $\mathsf{E}\left[N^2\,\middle|\,S_0\right] = \lambda_f^2 S_0^2 + \lambda_f S_0$ and $\mathsf{E}[N\,|\,S_0] = \lambda_f S_0$.

s.3.4.14. For the first equation,

$$\mathsf{E}\left[S^2\,\middle|\,S_0\right] = S_0^2 + 2\lambda_f\,\mathsf{E}[R]S_0^2 + \lambda_f\,\mathsf{E}\left[R^2\right]S_0 + \lambda_f^2(\mathsf{E}[R])^2S_0^2.$$

Assemble all terms with S_0^2 and observe that $(1/A) = 1 + \lambda_f\,\mathsf{E}[R]$. For the second, recall that we assumed at first that S_0 was fixed, which we indicated by the condition on S_0. When S_0 is a random variable, we can just take the expectation at the left and right, and obtain the second result.

s.3.4.15.

$$\mathsf{V}[S] = \mathsf{E}\left[S^2\right] - (\mathsf{E}[S])^2 = \frac{\mathsf{E}\left[S_0^2\right]}{A^2} + \lambda_f\,\mathsf{E}\left[R^2\right]\mathsf{E}[S_0] - \frac{(\mathsf{E}[S_0])^2}{A^2}.$$

s.3.4.16. Using 3.4.10–3.4.15

$$\begin{aligned}
C_s^2 &= \frac{\mathsf{V}[S]}{(\mathsf{E}[S])^2} = \frac{V(S)A^2}{(\mathsf{E}[S_0])^2} \\
&= \frac{\mathsf{E}\left[S_0^2\right] + \lambda_f\,\mathsf{E}\left[R^2\right]\mathsf{E}[S_0]A^2 - (\mathsf{E}[S_0])^2}{(\mathsf{E}[S_0])^2} \\
&= \frac{\mathsf{E}\left[S_0^2\right] - (\mathsf{E}[S_0])^2}{(\mathsf{E}[S_0])^2} + \frac{\lambda_f\,\mathsf{E}\left[R^2\right]\mathsf{E}[S_0]A^2}{(\mathsf{E}[S_0])^2} \\
&= C_0^2 + \frac{\lambda_f\,\mathsf{E}\left[R^2\right]A^2}{\mathsf{E}[S_0]}.
\end{aligned}$$

s.3.4.17. When repair times are exponentially distributed with mean $\mathsf{E}[R]$: $\mathsf{E}\left[R^2\right] = 2(\mathsf{E}[R])^2$. Since $A = 1/(1+\lambda_f\,\mathsf{E}[R])$,

$$\begin{aligned}
\lambda_f\,\mathsf{E}\left[R^2\right]A^2 &= 2\lambda_f(\mathsf{E}[R])^2A^2 = 2\lambda_f\,\mathsf{E}[R]AA\,\mathsf{E}[R] \\
&= 2\frac{\lambda_f\,\mathsf{E}[R]}{1+\lambda_f\,\mathsf{E}[R]}A\,\mathsf{E}[R] \\
&= 2\left(1 - \frac{1}{1+\lambda_f\,\mathsf{E}[R]}\right)A\,\mathsf{E}[R] = 2(1-A)A\,\mathsf{E}[R].
\end{aligned}$$

3.5 OLD EXAM QUESTIONS

3.5.1 *Multiple-choice Questions*

3.5.1. *[201807] Sakasegawa's approximation for the waiting time in a G/G/c queue is*

$$\mathsf{E}\left[W_Q\right] = \frac{C_a^2 + C_s^2}{2}\frac{\rho^{\sqrt{2(c+1)}-1}}{c(1-\rho)}\,\mathsf{E}[S].$$

We claim that it is exact for the M/M/1 queue.

3.5.2. *[201807] A production system consists of 2 stations in tandem. The first station has one machine, the second has two identical machines. Machines never fail and service times are deterministic. Jobs arrive at rate 1 per hour. The machine at first station has a service time of 45 minutes per job, a machine at the second station has a service time of 80 minutes. We claim that the second station is the bottleneck.*

3.5.3. *[201904] A job's normal service time, without interruptions, is given by S_0. The durations of interruptions are given by the i.i.d. random variables $\{R_i\}$ and have common mean $\mathsf{E}[R]$ and variance $\mathsf{V}[R]$. If N interruptions occur, the effective service time will then be*

$$S = S_0 + \sum_{i=1}^{N} R_i.$$

Then all steps in the computation below are correct:

$$\mathsf{E}\left[\sum_{i=1}^{N} R_i\right] = \mathsf{E}\left[\sum_{n=0}^{\infty} \mathbb{1}_{N=n}\right] \mathsf{E}\left[\sum_{i=1}^{n} R_i\right] = \mathsf{E}[N]\,\mathsf{E}[R]$$

3.5.4. *[201907] When a bus reaches the end of its line, it undergoes a series of inspections. The entire inspection takes 5 minutes on average, with a standard deviation of 2 minutes. Buses arrive with inter-arrival times uniformly distributed on $[3,9]$ minutes. We model the number of buses waiting to be inspected as an $G/G/1$ queue. The code below correctly computes the waiting time in queue.*

```
>>> a = 3.
>>> b = 9.
>>> EX = (b+a)/2. # expected inter-arrival time
>>> EX
6.0
>>> labda = 1./EX # per minute
>>> VA = (b-a)*(b-a)/12.
>>> CA2 = VA/(EX*EX)
>>> ES = 5.
>>> sigma = 2
>>> VS = sigma*sigma
>>> CS2 = VS/(ES*ES)
>>> rho = labda*ES
>>> Wq = (CA2+CS2)/2. * rho/(1.-rho) * ES
```

3.5.5. *[201907] A station contains 3 identical machines in parallel. Jobs arrive as a Poisson process with rate 3 per hour. Service times are exponential with a mean duration of 1/2 hour. This queueing system cannot be modeled with level-crossing arguments, hence we need Sakasegawa's formula to compute the average time in queue.*

3.5.2 *Open Questions*

3.5.6. *[201706] We have a machine that fails regularly, but we can control the way it fails. Option 1: Clean and tune the machine for precisely 15 minutes at the start of every hour. Option 2: Don't clean the machine, but wait until it fails and then do a repair. In the latter*

situation, the time between two failures is exponentially distributed with a mean of 6 hours. Repair times are i.i.d. and exponentially distributed with a mean of 1 hour. To simplify the analysis, you are allowed for Option 1 to model the time between the cleaning actions and the cleaning actions themselves as exponentially distributed.

Jobs arrive as a Poisson process with rate 2 *per hour, regular service times are exponential with a mean of* 10 *minutes. Which of the two options would you prefer?*

A station contains 3 identical machines in parallel. Jobs arrive as a Poisson process with rate 3 per hour. Service times are exponential with a mean duration of 1/2 hour.

3.5.7. *[201807] Is there a model by which we can obtain an exact answer for the average time the system contains n jobs?*

3.5.8. *[201807] Use Sakasegawa's approximation to compute the average time a job spends in queue. (You do not have to use your calculator to compute the final answer;. I just want to see the numerical values of each component of the formula, you don't have to compute the end result.)*

3.5.9. *[201807] What is the average time a job spends in the system?*

Some students add three times the average service time, but servers in parallel are not servers in tandem...

3.5.10. *[201807] Suppose now that each machine can fail between any two jobs with constant probability* $p = 1/3$*. The repair time is constant, and takes 1 hour. What is the average time a job spends in queue? (Again, it suffices if you show the numerical answer for each component in the formula.)*

3.5.3 *simulation*

In this section, we will deal with the code below.

```
from heapq import heappop, heappush
import numpy as np
from scipy.stats import expon

np.random.seed(3)

ARRIVAL = 0
DEPARTURE = 1

stack = [] # this is the event stack
queue = []
served_jobs = [] # used for statistics

num_jobs = 10
labda = 2.0
mu = 3.0
rho = labda / mu
F = expon(scale=1.0 / labda) # interarrival time distributon
```

```
19 G = expon(scale=1.0 / mu) # service time distributon
20
21
22 class Server:
23  def __init__(self):
24  self.busy = False
25
26
27 server = Server()
28
29
30 class Job:
31  def __init__(self):
32  self.arrival_time = 0
33  self.service_time = 0
34  self.departure_time = 0
35  self.queue_length_at_arrival = 0
36
37  def sojourn_time(self):
38  pass # write your code below.
39
40
41  def __repr__(self):
42  return f"{self.arrival_time}, {self.service_time}, {self.departure_time}\n"
43
44
45 def start_service(time, job):
46  server.busy = True
47  job.departure_time = time + job.service_time
48  heappush(stack, (job.departure_time, job, DEPARTURE))
49
50
51 def handle_arrival(time, job):
52  job.queue_length_at_arrival = len(queue)
53  if server.busy:
54  heappush(queue, (job.arrival_time, job))
55  else:
56  start_service(time, job)
57
58
59 def handle_departure(time, job):
60  server.busy = False
61  if queue: # queue is not empty
62  time, next_job = heappop(queue)
63  start_service(time, next_job)
64
65
```

```
66 time = 0
67 for i in range(num_jobs):
68  job = Job()
69  time += F.rvs()
70  job.arrival_time = time
71  job.service_time = G.rvs()
72  heappush(stack, (job.arrival_time, job, ARRIVAL))
73
74
75 while stack:
76  time, job, typ = heappop(stack)
77  if typ == ARRIVAL:
78  handle_arrival(time, job)
79  else:
80  handle_departure(time, job)
81  served_jobs.append(job)
```

3.5.11. *[201907] The above code simulates the G/G/1 queue. Complete the code to compute the sojourn time. (You can write your code at the correct place.)*

3.5.12. *[201907] Explain the code from lines 45 to 48.*

3.5.13. *[201907] Suppose we want to implement the LIFO queue. Which line of the above code has to change, and what should the change be? (It is ok if your answer is not perfect python code; however, your answer should show your correct understanding.)*

3.5.14. *[201907] Suppose we want to implement the shortest-processing-time-first scheduling rule. Which line of the above code has to change, and what should the change be?*

3.5.15. *[201907] What is the aim of test-driven coding?*

3.5.16. *[201907] Why is an event stack an essential concept for the discrete-time simulation of complicated systems?*

Solutions

s.3.5.1. Answer = A.

s.3.5.2. Answer = B. The second station has a utilization of $80/(2*60) = 8/12 = 2/3$, while the first has a utilization of $45/60 = 3/4$, which is higher.

s.3.5.3. Answer = B, 3.4.6

s.3.5.4. Answer = A, see 3.1.7.

s.3.5.5. Answer = B, see 3.5.7.

s.3.5.6. Note, the time between two failures is not the same as the time to failure.
Option 1. The availability is $A = 45/60 = 3/4$. Hence,

$$\mathsf{E}[S_e] = 10 \cdot 4/3 = 40/3 \approx 13 \text{ minutes}$$

We also have

$$C_e^2 = C_0^2 + 2A(1-A)\frac{m_r}{\mathsf{E}[S_0]} = 1 + 2\frac{3}{4}\frac{1}{4}\frac{15}{10}.$$

Now,

$$\rho = \lambda\,\mathsf{E}[S_e] = \frac{2}{60}\frac{40}{3}.$$

Now we can fill in

$$\mathsf{E}\left[W_Q\right] = \frac{1+C_e^2}{2}\frac{\rho}{1-\rho}\mathsf{E}[S_e].$$

Finally, $EW = \mathsf{E}\left[W_Q\right] + \mathsf{E}[S_e]$.

Option 2. The availability is $A = 50/60 = 5/6$. Hence,

$$\mathsf{E}[S_e] = 10 \cdot 6/5 = 12 \text{ minutes}$$

We also have

$$C_e^2 = C_0^2 + 2A(1-A)\frac{m_r}{\mathsf{E}[S_0]} = 1 + 2\frac{5}{6}\frac{1}{6}\frac{60}{10}.$$

and

$$\rho = \lambda\,\mathsf{E}[S_e] = \frac{2}{60}12$$

It is essential that you realize that both $\mathsf{E}[S]$ and C_e^2 are affected by failures. If you forgot to compensate in either of the two, I subtracted one point.

s.3.5.7. Yes, use the $M/M(n)/1$ queueing model, or the $M/M/3$ model (which is a subset of the $M/M(n)/1$ queue).

Note the $M/M/1$ queue with a fast server is not the same as the $M/M/3$ server. The service process is different.

s.3.5.8. $\rho = \lambda\,\mathsf{E}[S]/c = 3 * 0.5/3 = 0.5$.

$$\begin{aligned}\mathsf{E}\left[W_Q\right] &= \frac{C_a^2 + C_s^2}{2}\frac{\rho^{\sqrt{2(c+1)}-1}}{c(1-\rho)}\mathsf{E}[S] \\ &= \frac{1+1}{2}\frac{0.5^{\sqrt{7}}}{3*0.5} * 0.5 \\ &= \frac{0.5^{\sqrt{7}}}{3}.\end{aligned}$$

s.3.5.9. This is $\mathsf{E}\left[W_Q\right] + \mathsf{E}[S] = \mathsf{E}\left[W_Q\right] + 0.5$

s.3.5.10. The expected service time of a job now becomes

$$\mathsf{E}[S] = \mathsf{E}[S_0] + \mathsf{E}[T]/B = 0.5 + 1/3 = 5/6$$

Next, with the formula sheet:

$$\mathsf{V}[S] = 0.5 * 2 + \frac{1}{3} + \frac{3-1}{3^2}1 = 1.074074074074074.$$

With this,

$$C_s^{=} \frac{\mathsf{V}[S]}{(\mathsf{E}[S])^2} = \mathsf{V}[S]\frac{36}{25} = 1.3748148148148147.$$

Finally, $\rho = 3 * \mathsf{E}[S]/3 = 5/6$, and fill in Sakasegawa's formula with these values.

Mind that failures affect $\mathsf{E}[S]$ *and* C_s^2. If you forgot to compute C_s^2: -1/2.

Some students use the wrong failure model, the one with preemptive failures: -1/2.

s.3.5.11. This is the code:

```
def sojourn_time(self):
return self.departure_time - self.arrival_time
```

s.3.5.12. First we set a flag to indicate that the server is occupied. Then we compute the departure time. Finally, we push the job on the event stack such that it will popped by the simulator at its departure time.

s.3.5.13. `heappush(queue, (-job.arrival_time, job))`

s.3.5.14. `heappush(queue, (job.service_time, job))`

s.3.5.15. Write some small function. Test it on some known data. Once the function works, build a new function, test this, and so on. Then combine the tested functions to make something more difficult, test, etc.

s.3.5.16. The event stack keeps all events in the simulator ordered in time.

4

QUEUEING NETWORKS

We refer to the relevant sections of Zijm's book for background. Here we just include the solutions and repair a few typos.

4.1 OPEN SINGLE-CLASS PRODUCT-FORM NETWORKS

Theory and Exercises

The remark above Zijm.Eq.2.11 is not entirely correct. Remove the sentence: 'These visit ratios satisfy ... up to a multiplicative constant'.

I don't like the derivation of Zijm.Eq.2.20. The appearance of the visit ratios λ_i/γ seems to come out of thin air. The argument should be like this. Consider the entire queueing network as one 'box' in which jobs enter at rate $\gamma = \sum_{i=1}^{M} \gamma_i$. Assuming that there is sufficient capacity at each station, i.e., $\lambda_i < c_i \mu_i$ at each station i, the output rate of the 'box' must also be γ. Thus, by applying Little's law to the 'box', we have that

$$\mathsf{E}[L] = \gamma \mathsf{E}[W].$$

It is also evident that the average total number of jobs must be equal to the sum of the average number of jobs at each station:

$$\mathsf{E}[L] = \sum_{i=1}^{M} \mathsf{E}[L_i].$$

Applying Little's law to each station separately we get that $\mathsf{E}[L_i] = \lambda_i \mathsf{E}[W_i]$. Filling this into the above,

$$\mathsf{E}[W] = \frac{\mathsf{E}[L]}{\gamma} = \sum_{i=1}^{M} \frac{\mathsf{E}[L_i]}{\gamma} = \sum_{i=1}^{M} \frac{\lambda_i \mathsf{E}[W_i]}{\gamma},$$

where we recognize the visit ratios.

4.1.1. *[Linear algebra refresher] Can you find an example to show for two matrices A and B that $AB \neq BA$, hence $xA \neq Ax$.*

4.1.2. *[Linear algebra refresher 2] Suppose the matrix A has an eigenvalue 0. What is the geometric meaning of this fact?*

4.1.3. *Zijm.Ex.2.2.1*

4.1.4. *Zijm.Ex.2.2.2*

4.1.5. *Zijm.Ex.2.2.3*

4.1.6. *Zijm.Ex.2.2.4*

4.1.7. *Zijm.Ex.2.2.5. The problem is not entirely correctly formulated. It should be, if for at least one i, $\sum_{j=1}^{M} P_{ij} < 1$ ...*

4.1.8. *Zijm.Ex.2.2.6*

4.1.9. *Show that Zijm.Eq.2.13 and 2.14 can be written as*

$$f_i(n_i) = \frac{1}{G(i)} \frac{1}{\prod_{k=1}^{n_i} \min\{k, c_i\}} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}.$$

4.1.10. *We have a two-station single-server open network. Jobs enter the network at the first station with rate γ. A fraction α returns from station 1 to itself; the rest moves to station 2. At station 2 a fraction β_2 returns to station 2 again, a fraction β_1 goes to station 1. Compute λ. What happens if $\alpha \to 1$ or $\beta_1 \to 0$?*

4.1.11. *Zijm.Ex.2.2.8*

Hints

h.4.1.1. Let

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}.$$

Solutions

s.4.1.1.

$$AB = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} \neq \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} = BA.$$

Take $x = (1,1)$, then $xA = (1,2)$. Now, taking $x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, we get $Ax = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$. Recall, horizontal vectors are not vertical vectors. The horizontal ones are to the left of a matrix, and the vertical ones to the right.

s.4.1.2. Many students think that a matrix is just a bunch of numbers ordered in a grid. This is, in my opinion, the most unproductive way to think about matrices. A much more useful way is to see a matrix as an *operator*. For instance, take A to be a 3×3 matrix. Then it can be seen as a *mapping* from $\mathbb{R}^3$ to $\mathbb{R}^3$; it takes a vector $x \in \mathbb{R}^3$ and changes x into a new vector $Ax \in \mathbb{R}^3$. Thus, a square matrix A typically changes the length and direction of a vector x.

The next example is meant to illustrate what happens when a matrix has an eigenvalue 0. Consider the simple example with

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Clearly, A has an eigenvalue 0. Now take $v = (x, y, z)$, so that

$$Av = A \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x \\ y \\ 0 \end{pmatrix}.$$

We see that A removes any information about the z-direction from the vector v. (It projects v on the $x-y$ plane, and throws away the z component of v.) But then, for a given vector $w = (x, y, 0)$

in the $x-y$ plane, it is impossible to use A to retrieve the original vector $v=(x,y,z)$. Thus, A cannot have an inverse on all of $\mathbb{R}^3$.

So, hopefully, with this example, you can memorize that for any matrix A to have an inverse, it is essential that it has no zero eigenvalues. When the *operator* A (don't think of a matrix as a set of numbers) throws away part of the dimension of the space on which it operates (i.e., it has one or more eigenvalue(s) 0), it is impossible to retrieve the part of the space it throws away. Hence, its inverse cannot be used to get this part of the space back.

s.4.1.3. Jobs arrive at rate λ. We always require rate-stability, i.e., $\lambda<\mu_1$. Thus, (unless jobs are created at the server (which is not the case)), the departure rate from station 1 must be λ. When there are jobs present at station 1, they leave at rate μ_1. (Note that when there are no jobs at the station, there are no departures.) Likewise, the departure rate of station 2 is λ.

s.4.1.4. Observe from 2.7.8 that the inter-departure times of the $M/M/1$ queue are also independent and identically exponentially distributed with rate λ. Since the arrival process at the second station is the departure process of the first station, it must be that the arrival process at the second station is also Poisson with rate λ. Interestingly, from the perspective of the second station it is as if there is no first station.

s.4.1.5. The question is not well specified. We know from Burke's law, see 2.7.7, that the arrival *process* at the second station is Poisson. If, however, we know that station 1 is empty, then it is unlikely that a job will arrive at station 2 in the very near future.

Note that only the steady-state distributions of the queue lengths are independent. Once you have information about the state of one of the queues, then certainly this is not in 'steady-state'.

s.4.1.6. Simple algebra. (I am not going to write it out here. If you are willing to provide me the answer in LaTeX I'll include it.)

s.4.1.7. Linear algebra is quite useful here!

Observe that P_{ij} is the probability that a job, after completing its service at node i, moves to node j. Then $\sum_{j=1}^{M}P_{ij}$ is the probability that a job moves from node i to another node in the network, i.e., stays in the network, while P_{i0} is the probability that a job leaving node i departs the network, in other words, the job is finished. When $\sum_{j=1}^{M}P_{ij}<1$, then more jobs enter node i from the network than that node i sends 'back' into the network. Conceptually, node i 'leaks jobs'.

Now, consider some node k such that $P_{ki}>0$, then the probability that a job that starts at node k, moves to node i and then leaves the network is equal to $P_{ki}P_{i0}$. Thus, since $P_{ki}>0$ and $P_{i0}>0$, the probability that a job leaves the network from node k in two steps is positive. More specifically, $P^2_{k0}=\sum_{j=0}^{M}P_{kj}P_{j0}\geq P_{ki}P_{i0}>0$.

The irreducibility assumption implies that in at most M steps it is possible to reach, with positive probability, any node from any other node in the network. Thus, for any node j to any other node k there is a sequence of nodes $j_1,j_2,\ldots,j_{M-1}$ such that $P^M_{jk}\geq P_{jj_1}P_{j_1j_2}\cdots P_{j_{M-1}k}>0$.

Thus, if there is a node i such that $P_{i0}>0$, then it is possible from any node that sends jobs to node i directly to leave the network in two steps. Likewise, when node i can be reached from node k in n steps, say, then $P^{n+1}_{k0}\geq P^n_{ki}P_{i0}>0$, i.e., in at most $n+1$ steps it is possible to leave the network from such node k. This implies, in particular, that for all nodes $k=1,2,\ldots,M$, i.e., all nodes in the network, $P^{M+1}_{k0}>0$. For this reason we consider P^{M+1} in the hint.

As a final remark for students with knowledge of Markov chains, observe that the routing matrix P does not correspond to the transition matrix of a recurrent Markov chain. Since for at least one row i, $\sum_{j=1}^{N} P_{ij} < 1$, the matrix P is sub-stochastic. Hence, a Markov chain induced by P cannot be irreducible, because for this to happen, the chain must stay in some absorbing set with probability 1.

s.4.1.8. Since M is finite, and $k \leq M$, the set of numbers P_{k0}^{M+1} is finite. This, together with the fact that $P_{k0}^{M+1} > 0$ for all k, implies that there is some number $\epsilon > 0$ such that $P_{k0}^{M+1} > \epsilon$. Hence, for all entries $k = 1,2,\ldots,M$, we have that $P_{kj}^{M+1} < 1-\epsilon$. This, in turn, implies that $P_{kj}^{2(M+1)} < (1-\epsilon)^2$, and so on, so that for any n, $P_{kj}^{n(M+1)} < (1-\epsilon)^n$. This implies, in more general terms, that the entries of P^n decrease geometrically fast to 0.

It is well known that for any bounded sequence x_i and $0 \leq \alpha < 1$, $\sum_{i=0}^{\infty} x_i \alpha^i < \infty$. By applying this insight to the entries of P^n it follows that $\sum_{n=0}^{\infty} P_{jk}^n < \infty$.

Finally, applying $\lambda = \gamma + \lambda P$ recursively, we get

$$\lambda = \gamma + \lambda P = \gamma + (\gamma + \lambda P)P = \gamma(1+P) + \lambda P^2 = \gamma(1+P+P^2) + \lambda P^3 \rightarrow \gamma \sum_{n=0}^{\infty} P^n.$$

By the above reasoning this last sum is well defined, and finite. (For math aficionados: the above argument is not necessarily valid for matrices P that are infinite, since then $\inf\{P_{ik}^M\}$ need not be strictly positive.)

Another interesting way to see all this is by making the simplifying assumption that P is a diagonalizable matrix. (The argument can be generalized to include matrices reduced to Jordan normal form, but this gives optimal clutter, but does not change the line of reasoning in any fundamental way.) In that case, there exists an invertible matrix V with the (left) eigenvectors of P as its rows and a diagonal matrix Λ with the eigenvalues on its diagonal such that

$$VP = \Lambda V.$$

Hence, premultiplying with V^{-1},

$$P = V^{-1}\Lambda V.$$

But then

$$P^2 = V^{-1}\Lambda V \cdot V^{-1}\Lambda V = V^{-1}\Lambda^2 V,$$

and in general $P^n = V^{-1}\Lambda^n V$. If each eigenvalue λ_i is such that its modulus $|\lambda_i| < 1$, then $\Lambda^n \rightarrow 0$ geometrically fast, hence $P^n \rightarrow 0$ geometrically fast, hence the sequence of partial sums $\sum_{n=0}^{N} P^n$ converges to a matrix with finite elements as $N \rightarrow \infty$.

So, we are left with proving that the eigenvalues of P must have modulus less than 1. This fact follows from Gerschgorin's disk theorem, which I include for the interested student. Define the disk $B(a,r) = \{z \in \mathbb{C} || z-a| \leq r\}$, i.e., the set of complex numbers such that the distance to the center $a \in \mathbb{C}$ is less than or equal to the radius r. With this, the Gerschgorin disks of a matrix are defined as $B(a_{ii}, \sum_{j \neq i} |a_{ij}|)$, i.e., disks with center at the diagonal elements a_{ii} of A and radius equal to the sum of the (modulus of the) elements of A on the ith row except a_{ii}. Then Gerschgorin's theorem says that all eigenvalues of A lie in the union of these disks, i.e., all eigenvalues $\lambda_i \in \bigcup_i B(a_{ii}, \sum_{j \neq i} |a_{ij})$.

Assume for notational simplicity that for each row i of P we have that $\sum_j a_{ij} < 1$. (Otherwise apply the argument to P^{M+1}.) Then this implies for all i that

$$a_{ii} + \sum_{j \neq i} a_{ij} < 1.$$

Since all elements of P are non-negative, this also implies that

$$-1 < a_{ii} - \sum_{j \neq i} a_{ij} \leq a_{ii} + \sum_{j \neq i} a_{ij} < 1.$$

With this and using that a_{ii} is a real number (so that it lies on the real number axis) it follows that all elements in the disk $B(a_{ii}, \sum_{j \neq i} a_{ij})$ have modulus smaller than 1. As this applies to any row i, all disks lie strictly within the complex unit circle. But then, by Gerschgorin's theorem, all eigenvalues of P also lie strictly in the unit circle, hence all eigenvalues have modulus smaller than 1.

s.4.1.9. Take $n_i < c_i$. Then $\Pi_{k=1}^{n_i} \min\{k, c_i\} = \Pi_{k=1}^{n_i} k = n_i!$, and $(c_i \rho_i)^{n_i} = (\lambda_i/\mu_i)^{n_i}$. If $n_i \geq c_i$, then $\Pi_{k=1}^{n_i} \min\{k, c_i\} = c_i! c_i^{n_i - c_i}$, and $(c_i \rho_i)^{n_i} = (\lambda_i/\mu_i)^{n_i} c_i^{n_i}$.

s.4.1.10.

$$P = \begin{pmatrix} \alpha & 1-\alpha \\ \beta_1 & \beta_2 \end{pmatrix}.$$

$$(\lambda_1, \lambda_2) = (\gamma, 0) + (\lambda_1, \lambda_2) P.$$

Solving first for λ_2 leads to $\lambda_2 = (1-\alpha)\lambda_1 + \beta_2 \lambda_2$, so that

$$\lambda_2 = \frac{1-\alpha}{1-\beta_2} \lambda_1.$$

Next, using this and that $\lambda_1 = \alpha \lambda_1 + \beta_1 \lambda_2 + \gamma$ gives with a bit of algebra

$$\begin{aligned} \gamma &= \lambda_1 (1-\alpha) - \beta_1 \lambda_2 \\ &= \lambda_1 \left(1 - \alpha - \beta_1 \frac{1-\alpha}{1-\beta_2} \right) \\ &= \lambda_1 (1-\alpha) \left(1 - \frac{\beta_1}{1-\beta_2} \right) \\ &= \lambda_1 (1-\alpha) \frac{1-\beta_1-\beta_2}{1-\beta_2}. \end{aligned}$$

Hence,

$$\lambda_1 = \frac{\gamma}{1-\alpha} \frac{1-\beta_2}{1-\beta_1-\beta_2}.$$

Thus,

$$\lambda_2 = \frac{1-\alpha}{1-\beta_2} \lambda_1 = \frac{\gamma}{1-\beta_1-\beta_2}.$$

We want of course that $\lambda_1 < \mu_1$ and $\lambda_2 < \mu_2$. With the above expressions this leads to conditions on α, β_1 and β_2. Note that we have three parameters, and two equations; there is not a single condition from which the stability can be guaranteed.

If $\alpha \uparrow 1$, the arrival rate at node 1 explodes. If $\beta_1 = 0$ no jobs are sent from node 2 to node 1.

s.4.1.11. Yes, the network remains a Jackson network. By Burke's law, see 2.7.7, the departure process of each node is Poisson. In one of the earlier questions we derived that splitting (also known as thinning) and merging Poisson streams again lead to Poisson streams. The departures from node j to node k forms a thinned Poisson stream. The external arrivals plus internal arrivals are merged into one Poisson stream, hence the arrivals at a station also form a Poisson stream.

Observe that the exponentiality of the service times and external inter-arrival times and Burke's law are essential for the argument.

4.2 TANDEM QUEUES

Theory and Exercises

Consider two $M/M/1$ stations in tandem. Suppose we can remove the variability in the service processing times at one, but not both, of the servers. Which one is the better one to spend it on, in terms of reducing waiting times? After we obtained some insights into this question, we will provide a model to approximate the waiting time in a tandem of $G/G/1$ queues.

4.2.1. *Assuming that jobs arrive at the first station at rate λ, and are served at rate μ_i at station i, show that the average queueing time for the tandem of two $M/M/1$ queues is given by*

$$\mathsf{E}\left[W_Q\right] = \frac{\rho_1}{1-\rho_1}\frac{1}{\mu_1} + \frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}, \tag{4.2.1}$$

where $\rho_i = \lambda/\mu_i$ and $\mathsf{E}[S_i] = 1/\mu_i$, for $i = 1,2$.

4.2.2. *Suppose we can remove all variability of the2 service process at the second station. Show that in this case the total time in queue is equal to*

$$\mathsf{E}\left[W_Q\right] = \frac{\rho_1}{1-\rho_1}\frac{1}{\mu_1} + \frac{1}{2}\frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}.$$

4.2.3. *Suppose now that we reduce the variability of the service process of the first station. Motivate that*

$$\mathsf{E}\left[W_Q\right] = \frac{1}{2}\frac{\rho_1}{1-\rho_1}\frac{1}{\mu_1} + \frac{1}{2}\frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}$$

is a reasonable approximation of the queueing time. Compare this to the queueing time of the reference situation.

4.2.4. *What do you conclude from the above exercises?*

For a tandem network of $G/G/1$ queues, observe that the SCV of the departure process $C^2_{d,i}$ of the ith station is the SCV of the arrival process $C^2_{a,i+1}$ at station $i+1$. Thus, if we have $C^2_{d,i}$ we can compute the average waiting time at station $i+1$ by means of the $G/G/1$ waiting time approximation.

To obtain an estimate for $C^2_{d,i}$ we reason as follows. Suppose that the load ρ_i at station i is very high. Then the server will seldom be idle, so that the departure process must be reasonably well approximated by the service process. If, however, the load is small, the server will be idle most of the time, and inter-departure times must be approximately distributed as the inter-arrival times. Based on this, we interpolate between these two extremes to get the approximation

$$C^2_{d,i} \approx (1-\rho_i^2)C^2_{a,i} + \rho_i^2 C^2_{s,i}. \tag{4.2.2}$$

4.2.5. *What is C^2_d for the $D/D/1$ queue according to* (4.2.2)*?*

4.2.6. *What is C^2_d for the $M/M/1$ queue according to* (4.2.2)*?*

4.2.7. *Use* (4.2.2) *to show for the $G/D/1$ that $C^2_d < C^2_a$.*

4.2.8. *Consider two $G/G/1$ stations in tandem. Suppose $\lambda = 2$ per hour, $C^2_{a,1} = 2$ at station 1, $C^2_s = 0.5$ at both stations, and $\mathsf{E}[S_1] = 20$ minutes and $\mathsf{E}[S_2] = 25$ minutes. What is the total time jobs spend on average in the system? What is the average number of jobs in the network?*

For a $G/G/c$ queue, we can use the following approximation

$$C_{d,i}^2 = 1 + (1-\rho_i^2)(C_{a,i}^2 - 1) + \frac{\rho_i^2}{\sqrt{c_i}}(C_{s,i}^2 - 1). \tag{4.2.3}$$

4.2.9. *Show that* (4.2.3) *reduces to* (4.2.2) *for the* $G/G/1$ *queue.*

For the interested reader we refer to Zijm, Section 2.4.2, for a discussion of an extension for $G/G/c$ queues in tandem, and to networks. In particular, in networks we need to be concerned with output streams merging into a single input stream at one station, and the splitting of the output stream of a station to several other stations. The algorithm discussed in Zijm, Section 2.4.2, is mainly useful for numerical analysis. We will not discuss it here.

Hints

h.4.2.1. Focus on the waiting times for each station separately, and realize that each is an $M/M/1$ queue. What is the arrival process at the second station? Recall Burke's law, cf., 2.7.7.

h.4.2.2. If we can reduce all service variability at the second server, the second station can be modeled as an $M/D/1$ queue.

h.4.2.3. Realize that now also the distribution of inter-departure times of the first station changes and becomes more regular.

h.4.2.4. What would you do if there would be a third station in this tandem network?

Solutions

s.4.2.1. The first queue is the familiar $M/M/1$ queue (why?). For the sequel it is important to observe that the departure process of the first station, i.e., the distribution of times between jobs leaving the first station, is the same as the arrival process. Consequently, the inter-departure times are also exponentially distributed with parameter λ. (However, the service times are exponentially distributed with parameter μ_1.)

What can we say about the second station? Clearly, the jobs departing at the first station are the arrivals at the second station. Hence, the departure process being exponential with rate λ, the inter-arrival times at the second station are also exponential with rate λ. Consequently, the second queue is also an $M/M/1$ queue.

The total waiting time in the system, i.e., the time spent in both queues, is the sum of the waiting times at the first and second station. As each is an $M/M/1$ queue, the total waiting time has the form:

$$\begin{aligned} \mathsf{E}[W_Q] &= \mathsf{E}[W_{Q,1}] + \mathsf{E}[W_{Q,2}] \\ &= \frac{\rho_1}{1-\rho_1}\frac{1}{\mu_1} + \frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}, \end{aligned} \tag{4.2.4}$$

where $\rho_i = \lambda/\mu_i$ and $\mathsf{E}[S_i] = 1/\mu_i$, for $i = 1,2$.

s.4.2.2. As we reduce the variability of the second server, the service process is no longer exponential. However, the arrival process at the second station is still Poisson. As a consequence,

the queueing discipline changes to the $M/G/1$ queue. The expected waiting time for this case has the form:

$$\mathsf{E}\left[W_{Q,2}\right] = \frac{1+C_{s,2}^2}{2}\frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}, \tag{4.2.5}$$

where $C_{s,2}^2$ is the squared coefficient of variation of the service process of the second server.

By assumption we can entirely remove the variability of the second server. This yields that the coefficient of variation $C_{s,2}^s = 0$. Thus, the service process being deterministic, the second station becomes the $M/D/1$ queue.

The expected waiting time for the $M/D/1$ queue follows immediately from (4.2.5) by setting $C_{s,2}^2 = 0$:

$$\mathsf{E}\left[W_{Q,2}\right] = \frac{1}{2}\frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}. \tag{4.2.6}$$

Clearly, this is half the waiting time of the $M/M/1$ queue.

Since we do not change the first station in any way, this is still an $M/M/1$ queue. Thus, the total time in queue for this scenario becomes:

$$\mathsf{E}\left[W_Q\right] = \frac{\rho_1}{1-\rho_1}\frac{1}{\mu_1} + \frac{1}{2}\frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}.$$

s.4.2.3. Analogous to the previous situation, suppose we can set the coefficient of variation $C_{1,s}^2$ of the first server to zero. Thus, this becomes an $M/D/1$ queue, so that, similar to (4.2.6):

$$\mathsf{E}\left[W_{Q,1}\right] = \frac{1}{2}\frac{\rho_1}{1-\rho_1}\frac{1}{\mu_1}.$$

Contrary to the $M/M/1$ queue, the inter-departures of the $M/D/1$ queue are not exponentially distributed. When the first server is busy, they are deterministic. When the first server is idle, we first need to wait for the next arrival, which is exponentially distributed, and then for this arrival to finish its service, which takes D. Thus, the time to the next departure is $X + D$.

However, for the sake of simplicity, let us simply assume in the sequel of this example that the departure process *is* deterministic.

As we previously remarked, the departure process of the first station forms the arrival process at the second station. Since the departures are assumed to be deterministic, the arrivals at the second station are also deterministic. The service times at the second station, however, are still exponential. Thus, the second station can be modeled as the $D/M/1$ queue. For this queue we need to derive an expression for the waiting time. The simplest approximation follows from an expression for the waiting time of the $G/G/1$ queue.

We know that the expected waiting time for the $G/G/1$ queue has the approximate form:

$$\mathsf{E}\left[W_{Q,2}\right] = \frac{C_{a,2}^2 + C_{s,2}^2}{2}\frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}. \tag{4.2.7}$$

Clearly, for our case, the coefficient of variation $C_{a,s}^2$ of the arrival process becomes, approximately, 0, while $C_{s,2}^2 = 1$, since the service process is still exponential. Hence,

$$\mathsf{E}\left[W_{Q,2}\right] = \frac{1}{2}\frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2}. \tag{4.2.8}$$

Combining (4.2.7) and (4.2.8), the total time in queue becomes:

$$\mathsf{E}[W_Q] = \frac{1}{2}\frac{\rho_1}{1-\rho_1}\frac{1}{\mu_1} + \frac{1}{2}\frac{\rho_2}{1-\rho_2}\frac{1}{\mu_2},$$

which is half the waiting time of the two $M/M/1$ stations in tandem, but also smaller than the situation in which we reduce the variability at the second station.

s.4.2.4. As a general guideline, it seems best to reduce the variability at the first station. The main point to remember is that reducing the variability of the service process at the first station also reduces the variability of its departure process, hence the variability of the arrival processes at the second station. Thus, the situation improves at two locations of the chain of stations, rather than at one.

s.4.2.5. Since the inter-arrival times and the service times are deterministic, $C_a^2 = C_s^2 = 0$. Hence $C_d^2 = 0$.

s.4.2.6. Since the inter-arrival times and the service times are exponential, $C_a^2 = C_s^2 = 1$. Hence $C_d^2 = 1$. This is precisely in line with our earlier insights, in which we obtained that the departure process of an $M/M/1$ queue is Poisson.

s.4.2.7. As $C_s^2 = 0$ for the $G/D/1$ queue, $C_d^2 = (1-\rho^2)C_a^2 < C_a^2$, as $\rho < 1$.

s.4.2.8. First station 1.

```
>>> labda = 2.
>>> S1 = 20./60
>>> rho1 = labda*S1
>>> rho1
0.6666666666666666
>>> ca1 = 2.
>>> cs1 = 0.5
>>> EW1 = (ca1+cs1)/2 * rho1/(1-rho1) * S1
>>> EW1
0.8333333333333331
>>> W1 = EW1 + S1
>>> W1
1.1666666666666665
```

Now station 2. We first need to compute C_{d1}^2.

```
>>> cd1 = (1-rho1**2)*ca1 + rho1**2*cs1
>>> cd1
1.3333333333333335

>>> labda = 2
>>> S2 = 25./60
>>> rho2 = labda*S2
>>> rho2
0.8333333333333334
```

```
>>> ca2 = cd1 # here we use our formula
>>> cs2 = 0.5
>>> EW2 = (ca2+cs2)/2 * rho2/(1-rho2) * S2
>>> EW2
1.909722222222223
>>> W2 = EW2 + S2
>>> W2
2.3263888888888897
```

With Little's law.

```
>>> W=W1+W2
>>> W
3.4930555555555562
>>> L = labda*W
>>> L
6.9861111111111125
```

s.4.2.9. Since $c = 1$ for the $G/G/1$ queue, we get

$$\begin{aligned} C_d^2 &= 1 + (1-\rho^2)(C_a^2 - 1) + \rho^2(C_s^2 - 1) \\ &= 1 + C_a^2 - \rho^2 C_a^2 - 1 + \rho^2 + \rho^2 C_s^2 - \rho^2 \\ &= (1-\rho^2)C_a^2 + \rho^2 C_s^2. \end{aligned}$$

4.3 GORDON-NEWELL NETWORKS

Theory and Exercises

4.3.1. *Provide an interpretation of a single-server queueing server with a finite calling population in terms of a closed network.*

The formula with the visit ratios should be like this:

$$V_k = \sum_{j=0}^{M} V_j P_{jk},$$

i.e., the sum should start at index 0. This is to include the load/unload station.

Also, assume that the load/unload station has just one server.

You should realize that the algorithms discussed in this section are meant to be carried out by computers. Thus the results will be numerical, not in terms of formulas.

Mind the order of V and P in the computation of the visit ratios: do not mix up $VP = V$ with $PV = V$, as in general, $VP \neq PV$. We use $VP = V$.

4.3.2. *Compute the visit ratios for a network with three stations such that all jobs from station 0 move to station 1, from station 1 all move to station 2, and from station 2 half of the jobs move to station 0 and the other half to station 1.*

4.3.3. *Zijm.Ex.3.1.1*

4.3.4. *Zijm.Ex.3.1.2*

4.3.5. *Zijm.Ex.3.1.3*

4.3.6. *Relate Zijm.Eq.3.3 to the form of the steady-state distribution of the number of jobs in an $M/M/c$ queue.*

4.3.7. *Zijm.Ex.3.1.4*

4.3.8. *Zijm.Ex.3.1.5*

Hints

h.4.3.8. First write down all different states, and then use Zijm.Eq.3.2 and 3.3.

Solutions

s.4.3.1. Consider a closed-queueing network with two stations. Station 1 is a single-server station with exponentially distributed service times with mean μ; Station 2 has N parallel exponential servers, each working at rate λ. If Station 1 contains n jobs, then Station 2 contains $N-n$ jobs. The rate at which jobs move from Station 2 to Station 1 is $(N-n)\lambda$, since $N-n$ of the N servers of Station 2 are occupied. Jobs move from Station 1 to Station 2 at rate μ, provided $n \geq 1$.

s.4.3.2. The routing matrix P is

$$P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1/2 & 1/2 & 0 \end{pmatrix}.$$

Solving $V = VP$ leads to $(V_0, V_1, V_2) = (V_2/2, V_0 + V_2/2, V_1)$. Thus, from the last item, $V_2 = V_1$, and from the first $V_0 = V_2/2$. Since $V_0 = 1$, it follows that $V_2 = 2V_0$ and $V_1 = V_2 = 2V_0$.

s.4.3.3. If a part would need refitting or repositioning at the load/unload station, then that part would visit the load/unload station more than once during its stay in the network. The visit ratio of the load/unload station can then no longer be set to 1.

s.4.3.4. Let's number the states from 1 to 4. If station 1 feeds into station 2, and so on, and station 4 into station 1, then

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

The visit ratios are, of course, $V_1 = V_2 = V_3 = V_4$.

s.4.3.5. We number station a as 1, and so on.

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1/5 & 0 & 4/5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/3 & 2/3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

We now solve for V in the equation $V = VP$. Since, by definition, node f is the load/unload node, we set $V_6 = 1$, and then we express the other visit ratios in terms of V_6. Note also that the rows of P sum up to one. Hence, From elementary linear algebra, the second column implies that $V_2 = V_1$, and the third that $V_3 = 4/5 \cdot V_2$, so that $V_3 = 4/5 \cdot V_1$. From the fourth column $V_4 = V_3 \cdot 1/3$, thus, $V_4 = V_1 \cdot 1/3 \cdot 4/5 = V_1 \cdot 4/15$. From the fifth column $V_5 = V_3 \cdot 2/3$, hence $V_5 = V_1 \cdot 4/5 \cdot 2/3 = V_1 \cdot 8/15$. Finally, from the sixth column, $V_6 = V_4 + V_5$, hence $V_6 = V_1(4/15 + 8/15) = V_1 \cdot 4/5$. Therefore, $V_1 = V_6 \cdot 5/4$. Then using $V_6 = 1$ and the above obtained results give $V_1 = V_2 = 5/4$, $V_3 = 1$, $V_4 = 1/3$, and $V_5 = 2/3$.

(Note that this also could be derived from the first column and the second column, i.e., since $V_1 = 1/5\dot{V}_2 + V_6$ and $V_2 = V_1$, rewriting gives, $V_1 = 1/5\dot{V}_2 + V_6 = V_1 = 1/5 \cdot V_1 + V_6$ which gives $4/5 \cdot V_1 = V_6$ and hence $V_1 = 5/4 \cdot V_6$.)

For later courses on Markov chains, it is important to note the following. Write the visit ratio equation $V = VP$ as $V(1-P) = 0$ where 1 is the indicator matrix. Clearly, V is a left eigenvector of the matrix $1-P$ with eigenvalue 0 (recall that $V(1-P) = 0 = 0 \cdot V$). Thus, at least one row or column of the matrix $1-P$ is superfluous to solve for V.

s.4.3.6. Except for the normalization constant, the expressions are the same as equations 1.36 and 1.37 of Zijm's book.

s.4.3.7. Consider network one with routing $1 \to 3 \to 4 \to 2 \to 1$, each transition occurring with probability one, i.e.,

$$P = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

All nodes are is visited equally often. Another network could correspond to the routing $1 \to 2 \to 3 \to 4 \to 1$. Yet another would be this:

$$P = \begin{pmatrix} 1/2 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1/2 & 1/2 \end{pmatrix}.$$

Observe that I choose the rate such that each node receives the same fraction of traffic.

For the interested: There must be many such 'equivalent' networks, but a general method to classify all equivalent networks seems hard. The question comes down to finding all stochastic matrices P that have at least one (left) eigenvector V in common. Observe that the visit ratio equation $VP = V$ implies that V is a left eigenvector with eigenvalue 1.

s.4.3.8. Since $M = 2$, we have three stations: Stations 0, 1, and 2. We also have $N = 2$ jobs. Thus, the states are $(2,0,0)$ (meaning that the load/unload station has 2 jobs, and stations 1 and 2 no jobs), $(0,2,0)$, $(0,0,2)$, $(1,1,0)$, $(1,0,1)$, and $(0,1,1)$. Note that the routing matrix P is not given, so that we cannot compute the visit ratios. Hence I leave them unspecified. Now, realize that for a fixed $\vec{n} = (n_0, n_1, n_2)$, $\Pi_i f_i(n_i) = f_0(n_0) f_1(n_1) f_2(n_2)$, so that if

$$\vec{n} = (2,0,0) \qquad f_0(2) f_1(0) f_2(0) = \left(\frac{V_0}{\mu_0}\right)^2,$$

$$\vec{n} = (0,0,2) \qquad f_0(0) f_1(0) f_2(2) = \left(\frac{V_2}{\mu_2}\right)^2,$$

$$\vec{n} = (0,2,0) \qquad f_0(0)f_1(2)f_2(0) = \frac{1}{2}\left(\frac{V_1}{\mu_1}\right)^2.$$

Note that in this last result we use that Station 1 has two servers. For the other combinations,

$$\vec{n} = (1,1,0) \qquad f_0(1)f_1(1)f_2(0) = \frac{V_0}{\mu_0}\frac{V_1}{\mu_1},$$
$$\vec{n} = (1,0,1) \qquad f_0(1)f_1(0)f_2(1) = \frac{V_0}{\mu_0}\frac{V_2}{\mu_2},$$
$$\vec{n} = (0,1,1) \qquad f_0(0)f_1(1)f_2(1) = \frac{V_1}{\mu_1}\frac{V_2}{\mu_2}.$$

Finally, add up all the above numbers to make $G(M,N)$.

4.4 MVA ALGORITHM

Theory and Exercises

4.4.1. *Consider two stations in tandem, stations 0 and 1. The service times are* $\mathsf{E}[S_0] = 2 = 1/\mu_0$ *and* $\mathsf{E}[S_1] = 3 = 1/\mu_1$ *hours. The routing matrix is*

$$P = \begin{pmatrix} 0 & 1 \\ 1/2 & 1/2 \end{pmatrix}.$$

Apply the MVA algorithm to this case.

4.4.2. *Zijm.Ex.3.1.13. Assume that all stations have just one server.*

4.4.3. *Zijm.Ex.3.1.14*

4.4.4. *Implement the MVA algorithm in your preferred computer language and make Figure 3.2.*

Hints

h.4.4.1. Start with $n = 1$, then consider $n = 2$ and so on.

Solutions

s.4.4.1. From $VP = V$ we conclude that $V_1 = 2V_0$. Take $n = 1$.

$$\begin{aligned}
\mathsf{E}[W_0(1)] &= \mathsf{E}[L_0(0)+1]\,\mathsf{E}[S_0] = \mathsf{E}[S_0] = 2,\\
\mathsf{E}[W_1(1)] &= \mathsf{E}[L_1(0)+1]\,\mathsf{E}[S_1] = \mathsf{E}[S_1] = 3,\\
\mathsf{E}[W(1)] &= \sum_{i=0}^{1} V_i\,\mathsf{E}[W_i(1)] = V_0 2 + V_1 3 = 1\cdot 2 + 2\cdot 3 = 8,\\
\mathrm{TH}_0(1) &= \frac{1}{\mathsf{E}[W(1)]} = \frac{1}{8},\\
\mathsf{E}[L_0(1)] &= \mathrm{TH}_0(1)\,\mathsf{E}[W_0(1)] = \frac{2}{8} = \frac{1}{4},
\end{aligned}$$

$$\mathsf{E}[L_1(1)] = \mathrm{TH}_1(1)\,\mathsf{E}[W_1(1)] = V_1\mathrm{TH}_0(1)\,\mathsf{E}[W_1(1)] = 2\cdot\frac{1}{8}\cdot 3 = \frac{3}{4}.$$

Now take $n = 2$.

$$\begin{aligned}
\mathsf{E}[W_0(2)] &= (\mathsf{E}[L_0(1)+1]\,\mathsf{E}[S_0] = (1/4+1)\mathsf{E}[S_0] = \frac{5}{4}\cdot 2 = \frac{5}{2},\\
\mathsf{E}[W_1(2)] &= (\mathsf{E}[L_1(1)+1]\,\mathsf{E}[S_1] = (3/4+1)\mathsf{E}[S_1] = \frac{7}{4}\cdot 3 = \frac{21}{4},\\
\mathsf{E}[W(2)] &= \sum_{i=0}^{1} V_i\,\mathsf{E}[W_i(2)] = \frac{5}{2} + 2\cdot\frac{21}{4} = 13,\\
\mathrm{TH}_0(2) &= \frac{2}{\mathsf{E}[W(2)]} = \frac{2}{13},\\
\mathrm{TH}_1(2) &= V_1\mathrm{TH}_0(2) = \frac{4}{13},\\
\mathsf{E}[L_0(2)] &= \mathrm{TH}_0(2)\,\mathsf{E}[W_0(2)] = \frac{2}{13}\cdot\frac{5}{2} = \frac{5}{13},\\
\mathsf{E}[L_1(2)] &= \mathrm{TH}_1(2)\,\mathsf{E}[W_1(2)] = \frac{4}{13}\cdot\frac{21}{4} = \frac{21}{13}.
\end{aligned}$$

And so on.

s.4.4.2. Since there are quite a lot of jobs, and station 0 is the bottleneck, i.e., the station with the highest load, most of the time there will be a queue at station 0. In fact, most of the jobs will be in queue in front of station 0. Therefore the rate out of station 0 will be approximately equal to its service rate, i.e., μ_0. Thus, station 1 will receive jobs at a rate $\lambda_1 \approx \mu_0$, i.e., $\lambda_1 \approx 2$. Now I simply model the queueing process at station 1 as an $M/M/1$ queue with $\mu_1 = 2.5$. Hence, $\rho_1 = \lambda_1/\mu_1 = 2/2.5 = 4/5$, hence $\mathsf{E}[L_1] = \rho_1/(1-\rho_1) = (4/5)/(1/5) = 4$.

s.4.4.3. I use the MVA algorithm to compute the expected number of jobs at each of the stations. With an interesting programming concept called caching, or memoization, the implementation becomes very simple. BTW, I will not ask about memoization during the exam. I include the numerical results at the end.

Start with computing the visit ratios.

```
>>> from functools import lru_cache
>>> import numpy as np
>>> from numpy.linalg import eig

>>> mu = np.array([2, 2.5, 3, 3, 3])
>>> c = np.array([1, 1, 1, 1, 1]) # single-server stations
>>> P = np.matrix([
...   [0, 1, 0, 0, 0],
...   [0, 0, 1, 0, 0],
...   [0, 0, 0, 1, 0],
...   [0, 0, 0, 0, 1],
...   [1, 0, 0, 0, 0],
```

```
... ]
... )

>>> v, w = eig(P.T)
>>> x = v.real.argmax()
>>> V = w[:, x].real
>>> V = V / V[0]
>>> print(V)
[[1.]
 [1.]
 [1.]
 [1.]
 [1.]]
```

The visit ratios are according to expectation.

The following implements the MVA algorithm. Note how easy the code becomes with memoization; I only have to specify the recursions and the 'boundary conditions', i.e., what to do when $n = 0$. For the rest I don't have to think about in what exact sequence each of the functions needs to be called. The memoization takes care of all these problems.

```
>>> @lru_cache(maxsize=None)
... def EL(j, n):
...     if n <= 0:
...         return 0
...     else:
...         return TH(j, n) * EW(j, n)
...

>>> @lru_cache(maxsize=None)
... def EW(j, n):
...     return (EL(j, n - 1) + 1) / mu[j]
...

>>> @lru_cache(maxsize=None)
... def TH(j, n):
...     if j == 0:
...           return n / sum(V[j] * EW(j, n) for j in range(len(mu)))
...     else:
...           return V[j] * TH(0, n)
...
```

This is all!

Now I print the expected jobs at the stations.

```
>>> print(" n EL0 EL1 EL2 EL3 EL4 TH0")
 n EL0 EL1 EL2 EL3 EL4 TH0
>>> for n in range(1,30):
...     res = "{:3d}".format(n)
```

```
...     for j in range(len(mu)):
...         res += "{:6.2f}".format(float(EL(j, n)))
...         res += "{:6.2f}".format(float(TH(0,n)))
...     print(res)
...
  1  0.26  0.53  0.21  0.53  0.18  0.53  0.18  0.53  0.18  0.53
  2  0.55  0.87  0.42  0.87  0.34  0.87  0.34  0.87  0.34  0.87
  3  0.87  1.12  0.64  1.12  0.50  1.12  0.50  1.12  0.50  1.12
  4  1.21  1.30  0.85  1.30  0.65  1.30  0.65  1.30  0.65  1.30
  5  1.58  1.43  1.06  1.43  0.79  1.43  0.79  1.43  0.79  1.43
  6  1.99  1.54  1.27  1.54  0.92  1.54  0.92  1.54  0.92  1.54
  7  2.42  1.62  1.47  1.62  1.04  1.62  1.04  1.62  1.04  1.62
  8  2.89  1.69  1.67  1.69  1.15  1.69  1.15  1.69  1.15  1.69
  9  3.39  1.74  1.86  1.74  1.25  1.74  1.25  1.74  1.25  1.74
 10  3.93  1.79  2.05  1.79  1.34  1.79  1.34  1.79  1.34  1.79
 11  4.50  1.83  2.23  1.83  1.42  1.83  1.42  1.83  1.42  1.83
 12  5.10  1.86  2.40  1.86  1.50  1.86  1.50  1.86  1.50  1.86
 13  5.74  1.88  2.56  1.88  1.57  1.88  1.57  1.88  1.57  1.88
 14  6.41  1.90  2.70  1.90  1.63  1.90  1.63  1.90  1.63  1.90
 15  7.11  1.92  2.84  1.92  1.68  1.92  1.68  1.92  1.68  1.92
 16  7.84  1.93  2.97  1.93  1.73  1.93  1.73  1.93  1.73  1.93
 17  8.60  1.95  3.09  1.95  1.77  1.95  1.77  1.95  1.77  1.95
 18  9.39  1.96  3.20  1.96  1.80  1.96  1.80  1.96  1.80  1.96
 19 10.20  1.96  3.30  1.96  1.84  1.96  1.84  1.96  1.84  1.96
 20 11.03  1.97  3.39  1.97  1.86  1.97  1.86  1.97  1.86  1.97
 21 11.88  1.98  3.47  1.98  1.88  1.98  1.88  1.98  1.88  1.98
 22 12.75  1.98  3.54  1.98  1.90  1.98  1.90  1.98  1.90  1.98
 23 13.64  1.98  3.60  1.98  1.92  1.98  1.92  1.98  1.92  1.98
 24 14.54  1.99  3.66  1.99  1.93  1.99  1.93  1.99  1.93  1.99
 25 15.46  1.99  3.70  1.99  1.95  1.99  1.95  1.99  1.95  1.99
 26 16.39  1.99  3.75  1.99  1.96  1.99  1.96  1.99  1.96  1.99
 27 17.33  1.99  3.78  1.99  1.96  1.99  1.96  1.99  1.96  1.99
 28 18.27  1.99  3.82  1.99  1.97  1.99  1.97  1.99  1.97  1.99
 29 19.23  2.00  3.84  2.00  1.98  2.00  1.98  2.00  1.98  2.00
```

This is interesting. Using the insights of the previous question, approximation stations 3, 4 and 5 as $M/M/1$ queues, we have according to this model that $\mathsf{E}[L_3] = \rho_3/(1-\rho_3)$. Taking $\lambda_3 = 2$ and $\mu_3 = 3$, we see that $\rho_3 = 2/3$ so that $\mathsf{E}[L_3] = 2$. This is very close to 1.98.

For the last line, with $n = 29$, the number of jobs at station 0 must therefore be approximately $29 - 4 - 2 - 2 - 2 = 19$. The algorithm supports our intuition!

s.4.4.4. See the answer of the previous question.

4.5 OLD EXAM QUESTIONS

4.5.1 *Multiple-choice Questions*

4.5.1. *[201804] Consider a network with n stations in tandem. At station i, the service times S_i for all machines at that station are the same and constant; station i contains N_i machines. The number of jobs required to keep all machines busy is $N = \sum_{i=1}^{n} N_i$, and the raw processing time $T_0 = \sum_{i=1}^{n} S_i$. Thus, if the number w of allowed jobs in the system is larger than N, the number of jobs waiting somewhere in queue is $w - N$.*

4.5.2. *[201804] Consider an M-station Jackson network with λ_i the total arrival rate of jobs at station i. The average waiting time jobs spend in the system is*

$$\mathsf{E}[W] = \sum_{i=1}^{M} \lambda_i \mathsf{E}[W_i]$$

4.5.3. *[201804] In a Jackson network the time-average probability that station i contains n_i jobs (either in queue or in service) is given $(1-\rho_i)\rho_i^{n_i}$.*

4.5.4. *[201804] For the $G/G/1$ we can approximate the SCV of the inter-departures by the formula $(1-\rho^2)C_a^2 + \rho^2 C_s^2$. For the $M/M/1$ queue this reduces to $C_d^2 = 1$.*

4.5.5. *[201804] Suppose in a tandem network of $G/G/c$ queues we can reduce C_s^2 of just one station by a factor 2. To improve the average waiting time in the entire chain, it is best to reduce $C_{s,1}^2$.*

4.5.6. *[201804] We have a two-station Jackson network. If the routing matrix is $P = \begin{pmatrix} 0 & 1 \\ r & 0 \end{pmatrix}$ the total arrival rate λ_1 at station 1 is finite only if $r = 1$.*

4.5.7. *[201807] A production network consists of 3 single-machine stations in tandem. The processing times constant and such that $t_1 = 2$ hours, $t_2 = 3$ hours and $t_3 = 2$ hours. We claim that the critical WIP $W_0 = 7/3$, the bottleneck capacity $r_b = 1/3$ and the raw processing time $T_0 = 7$.*

4.5.8. *[201904] For an open queueing network we need to compute the visit ratios with the equation $V = VP$, where V is a (horizontal) vector of visit ratios and P is the routing matrix.*

4.5.9. *[201904] We have a two-station single-server open network. Jobs enter the network at the first station with rate γ. A fraction α returns from station 1 to itself; the rest moves to station 2. At station 2, a fraction β_2 returns to station 2 again, a fraction β_1 goes to station 1. Then,*

$$P = \begin{pmatrix} \alpha & 1-\alpha \\ 1-\beta_1-\beta_2 & \beta_2 \end{pmatrix}.$$

4.5.10. *[201904] Jobs arrive at rate λ and are assembled into batches of size B. The average time a job waits until the batch is complete is $\mathsf{E}[W] = \frac{B-1}{2\lambda}$.*

4.5.11. *[201907] We have a queueing network with M exponential servers and Poisson arrival processes Let P_{ij} be the routing matrix. For stability it is necessary that $\sum_{j=1}^{M} P_{ij} < 1$ for at least one i.*

4.5.12. *[201907] We have two $M/M/1$ stations in tandem. The average queueing time for the network is given by*

$$\mathsf{E}\left[W_Q\right] = \frac{\rho_1}{1-\rho_1} + \frac{\rho_2}{1-\rho_2}. \tag{4.5.1}$$

4.5.13. *[201907] Consider a two-station Jackson network with $P = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ and $\gamma = (1,0)$. Let $p(i,j)$ be the stationary probability that the first (second) station contains i (j) jobs. Then*

$$\lambda p(0,0) = \mu_2 p(0,1).$$

4.5.2 *Open Questions*

We have a network with two single-server stations in tandem. Jobs arrive at the first station as a Poisson process with rate λ, the service times at stations 1 and 2 are i.i.d. and exponentially distributed with mean μ_i^{-1} for station i, $i = 1,2$. The entire network can contain at most one job, hence, when there is a job anywhere in the network, any new arrival is lost.

4.5.14. *[201704] Can the first station be characterized as an $M/M/1/1$ queue? why, or why not?*

4.5.15. *[201704] Make a sketch of the states and the transition rates.*

4.5.16. *[201704] What are the balance equations for this queueing network?*

4.5.17. *[201704] Find the stationary distribution of the number of jobs at the first and second station in terms of λ, μ_1 and μ_2.*

4.5.18. *[201704] Compute $\mathsf{E}[L_1]$ and $\mathsf{E}[L_2]$.*

4.5.19. *[201704] Compute $\mathsf{V}[L_1]$.*

4.5.20. *[201704] Suppose $\mu_1 = \mu_2 = \mu$. How much larger than λ should μ minimally be such that the loss probability is less than 5%?*

We have a closed network with two single-server stations and two jobs. Jobs from station 0 go to station 1, and from station 1 to station 0. Services times are exponentially distributed with mean $1/\mu_i$ at station $i, i = 0,1$.

4.5.21. *[201704] What are the visit ratios at each station?*

4.5.22. *[201704] Use the convolution algorithm to determine the value of the normalization constant for this network.*

4.5.23. *[201704] What is the fraction of jobs that see the load/unload station idle when moving to this station?*

We have a network with two single-server stations in tandem. Jobs arrive at the first station as a Poisson process with rate λ, the service times at stations 1 and 2 are i.i.d. and exponentially distributed with mean μ_i^{-1} for station i, $i = 1,2$. The waiting room at the first station is unlimited; the second station can contain at most one job. When the server at the second station is occupied, the server at first station blocks in the sense that it does not start service when the second station is busy.

4.5.24. *[201706] The assumed blocking policy at the first station is equivalent to the* preemptive repeat with without resampling *discipline, which means that the interrupted customer starts again with the original service time. Why is this so?*

4.5.25. *[201706] Can the second station be characterized as an $M/M/1/1$ queue?*

4.5.26. *[201706] Make a sketch of the state space, the transitions and the transition rates.*

4.5.27. *[201706] What is the stability criterion for this queueing network?*

4.5.28. *[201706] Suppose you have some resources (money) available to increase the processing rate of* just *one of the servers or invest in queueing space between stations 1 and 2.. Which of these options should you suggest to analyze first?*

Henceforth, assume that also the first station cannot contain more than one job.

4.5.29. *[201706] Find the stationary joint distribution of the number of jobs at the first and second station in terms of λ, μ_1 and μ_2.*

4.5.30. *[201706] What is the fraction of time the network is empty?*

4.5.31. *[201706] What is the throughput of this queueing network, i.e., the departure rate?*

4.5.32. *[201706] What is throughput of this queueing system in the limit $\mu_2 \to \infty$?*

4.5.3 *Closed Queueing Networks*

We have a closed network with three single-server stations, a load-unload node, and two jobs. Jobs from station 0 go to station 1. For station 1: half of the jobs go to station 0, the other half to station 2. For station 2: one out of three jobs goes to station 1, the rest to station 3. From station 3, all jobs go to station 1. Services times are exponentially distributed with mean 1, 2 and 3 at stations $1,2$ and 3, and 10 at station 0.

4.5.33. *[201706] What is the routing matrix?*

Suppose henceforth that the visit ratios are $(1,2,3,4)$.

4.5.34. *[201706] Use the convolution algorithm to determine $G(2,2)$.*

4.5.35. *[201706] For this network, why is $\mu_i f_i(k) = V_i f_i(k-1)$?*

4.5.36. *[201804] Consider a network of two stations. The first station can only send jobs to the next station when the next station contains less than K jobs. Provide a set of recursions to simulate the queue length process at both stations in discrete time.*

We have a Jackson network with two single-server stations in tandem. Jobs arrive at the first station as a Poisson process with rate λ, the service times at station i is $\text{Exp}(\mu_i)$.

4.5.37. *[201804] Make a sketch of the state space and the transition rates.*

4.5.38. *[201804] What are the balance equations on the boundary $n_1 \geq 0$, $n_2 = 0$.*

4.5.39. *[201804] Show that on the boundary $n_1 \geq 0, n_2 = 0$, the stationary distribution $\pi(n_1, n_2)$ of the number of jobs satisfies the balance equations.*

We have a closed network with two single-server stations. Jobs from station 0 go to station 1, and from station 1 to station 0 with probability 1/3 and from station 1 to itself with probability 2/3. Services times are exponentially distributed with mean 1 at station 0, and 2 at station 1.

4.5.40. *[201804] What is the routing matrix P? Use this to compute the visit ratios at each station.*

4.5.41. *[201804] If there is just one job allowed in the network, determine* $\mathsf{E}[W]$.

4.5.42. *[201804] Use the MVA to determine* $\mathsf{E}[W]$ *in case two jobs are allowed in the network.*

4.5.43. *[201804] Consider a factory that limits the amount of work on the factory floor to w. We may model the queueing process on the factory floor as a closed queueing network. However, in the real factory, the number of jobs on the floor cannot be constant. Why not?*

We have a closed network with three single-server stations. Jobs go from the load/unload station with probability 1/3 to station 2. From stations 1 and 2 jobs can only return to the load/unload station. Services times are exponentially distributed with mean $\mathsf{E}[S_0] = 1$ at the load/unload, $\mathsf{E}[S_1] = 2$ at station 1 and $\mathsf{E}[S_2] = 3$ at station 2.

4.5.44. *[201807] Make a sketch of the transitions; use this to make the routing matrix P.*

4.5.45. *[201807] What are the visit ratios?*

4.5.46. *[201807] If there is just one job allowed in the network, determine* $\mathsf{E}[W]$.

4.5.47. *[201807] If two jobs are allowed in the network, what is the expected number of jobs at the second station?*

The following code implements the mean value analysis algorithm for a closed queueing network.

```
num_stations = 3
V = [1, 2 / 3, 1 / 3]
ES = [1, 2, 3]
EL = [0] * num_stations
EW = [0] * num_stations


for n in range(1, 4):
 for j in range(num_stations):
 EW[j] = EL[j] * ES[j] + ES[j]
 EW_tot = 0
 for j in range(num_stations):
 EW_tot += EW[j]
 TH = n / EW_tot
 for j in range(num_stations):
 EL[j] = EW[j] * V[j] * TH
 print(EL)
```

4.5.48. *[201904] By which line in the code do we control the number of jobs that are present in the network?*

4.5.49. *[201904] One line in the code above is wrong. Repair the incorrect line (you can write the correct expression in mathematical notation; it is not necessary to provide the python code.)*

4.5.50. *[201904] Assume that the algorithm is correct. Suppose that we would merge lines 8–16 into the following:*

```
1 for n in range(1, 4):
2  EW_tot = 0
3  for j in range(num_stations):
4  EW[j] = EL[j] * ES[j] + ES[j]
5  EW_tot += EW[j]
6  TH = n / EW_tot
7  EL[j] = EW[j] * V[j] * TH
```

Explain why is this wrong.

In the tutorial we developed a number of important simulation concepts.

4.5.51. *[201904] Why is an event stack useful in discrete-event simulation?*

4.5.52. *[201904] Why do we use classes and objects in the simulations?*

Solutions

s.4.5.1. Answer = B.

In general the number of jobs in queue is much higher than $w - N$. Consider the example $S_1 = 10$ and $N_1 = 10$, and $S_2 = 1, N_2 = 20$, and $n = 2$. Clearly, 19 machines at station 2 are always empty.

s.4.5.2. Answer = B. A simple reason is that the units left and right don't match: left time, right number per time times time.

s.4.5.3. Answer = A.

s.4.5.4. Answer = A.

s.4.5.5. Answer = A.

s.4.5.6. Answer = B, see Zijm, Page 38.

When $r = 1$, the matrix P has an eigenvalue 1. Hence, the equation $\lambda = \gamma + \lambda P$ has no inverse in this case.

s.4.5.7. Answer = A.

s.4.5.8. Answer = B. The claim applies to closed queueing networks.

s.4.5.9. Answer = B, 4.1.10

s.4.5.10. Answer = A, 3.2.5

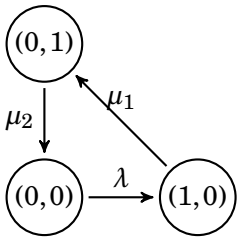
s.4.5.11. Answer = A, 4.1.7.

s.4.5.12. Answer = B, see 4.2.1. It is evidently wrong: the units don't check.

s.4.5.13. Answer = A.

s.4.5.14. No, if the server at the first station is free but the second is occupied, the first server still has to reject any arriving job. This is not the case for the $M/M/1/1$ queue.

s.4.5.15.



s.4.5.16.

$$\mu_1 p(1,0) = \lambda p(0,0)$$
$$\mu_2 p(0,1) = \mu_1 p(1,0) = \lambda p(0,0).$$

s.4.5.17. Define $\rho_i = \lambda/\mu_i$.

$$p(1,0) = \rho_1 p(0,0)$$
$$p(0,1) = \rho_2 p(0,0).$$

With the normalization requirement $p(0,0) + p(1,0) + p(0,1) = 1$ we get $p(0,0)(1+\rho_1+\rho_2) = 1$, hence

$$p(0,0) = \frac{1}{1+\rho_1+\rho_2}.$$

s.4.5.18.

$$\mathsf{E}[L_1] = 0(p(0,0)+p(0,1)) + 1p(1,0) = \frac{\rho_1}{1+\rho_1+\rho_2}$$

and

$$\mathsf{E}[L_2] = \frac{\rho_2}{1+\rho_1+\rho_2}$$

s.4.5.19.

$$\mathsf{V}[L_1] = \mathsf{E}\left[L_1^2\right] - (\mathsf{E}[L_1])^2 = \frac{\rho_1}{1+\rho_1+\rho_2} - \left(\frac{\rho_1}{1+\rho_1+\rho_2}\right)^2.$$

s.4.5.20. The acceptance probability is $p(0,0)$. Let $\rho = \lambda/\mu$. Then

$$0.95 = \frac{19}{20} \leq \frac{1}{1+2\rho} \iff 2\rho \leq 1/19 \iff \rho \leq 1/38. \iff \lambda < \frac{\mu}{38}.$$

s.4.5.21. Clearly, $V_0 = V_1 = 1$.

s.4.5.22. From the boundary conditions we know $G(m,0)$ and $G(0,n)$. Note that $M = 1$, since there just two stations. Since there are $N = 2$ jobs, we need $G(1,2)$.

$$\begin{aligned} G(1,2) &= f_1(0)G(0,2) + f_1(1)G(0,1) + f_1(2)G(0,0) \\ &= 1 \cdot \mu_0^{-2} + \frac{1}{\mu_1}\frac{\mu_0}{+}\frac{\mu_1^2}{1} \\ &= \frac{1}{16} + \frac{1}{12} + \frac{9}{.} \end{aligned}$$

s.4.5.23. When the load/unload station is idle, there are no jobs, hence we need $\mathsf{P}(n_0 = 0)$. Clearly, when $n_0 = 0$, $n_1 = 2$. Hence,

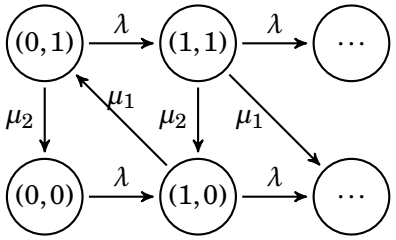
$$\pi(0,2) = \frac{1}{G(1,2)} f_0(0) f_1(2).$$

Since $f_0(0) = 1$ and $f_1(2) = \mu_1^{-2}$, the answer follows.

s.4.5.24. Because the service times are exponentially distributed, hence memoryless.

s.4.5.25. No, due to blocking, the departure process of the first station depends on the state of the next server. Hence, the how jobs arrive at the second station depends on the state of the second server. For the $M/M/1/1$ the arrival process does not depend on the state of the server; only the process of accepting jobs depends on the state of the server.

s.4.5.26.



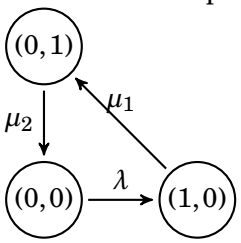
s.4.5.27. For step to the right, i.e., arrival, there must be a service at the first and second station. Hence $\lambda(\mathsf{E}[S_1] + \mathsf{E}[S_2]) < 1$.

The wording of the actual exam was wrong; I included the solution in the question... For this reason I removed the question. However, I still gave a point for a reasonable answer.

s.4.5.28. Due to blocking the first server and second server are hardly occupied. If we have unlimited buffer space between the two stations, the stability condition is $\lambda \min\{ES_1, \mathsf{E}[S_2]\} < 1$. Thus, if possible, expanding the queueing space is an easy solution.

However, any reasonable answer would do here.

s.4.5.29. The state space plus transitions becomes like this now:



Define $\rho_i = \lambda/\mu_i$. Write $p = p(0,0), q = p(1,0), r = p(0,1), s = p(1,1)$. Then,

$$\begin{aligned} \lambda p &= \mu_2 r \\ \lambda r &= \mu_2 s, \\ \mu_1 q &= \lambda p + \mu_2 s. \end{aligned}$$

We have four unknowns and three equations. With normalization we have four equations, so the above should suffice. Expression everything in terms of p:

$$r = \rho_2 p$$

$$s = \rho_2 r = \rho_2^2 p,$$
$$q = \rho_1 p + \mu_2 s/\mu_1 = \rho_1 p + \rho_1\rho_2 p = p\rho_1(1+\rho_2).$$

The normalization condition gives

$$p(1+\rho_1(1+\rho_2)+\rho_2+\rho_2^2) = 1.$$

s.4.5.30. This is p.

s.4.5.31. Jobs arrive at rate λ. Only the jobs that arrive when the first server is free are accepted. Hence $\lambda(p+r)$.

s.4.5.32. In the limit $\mu_2 \to \infty$, there is no job at the second station. Thus, in this case, $r = s = 0$. Moreover

$$1+\rho_1(1+\rho_2)+\rho_2+\rho_2^2 \to 1+\rho_1.$$

Hence $p = (1+\rho_1)^{-1}$. And this is what we get for a single server station with blocking.

It is actually interesting to also consider the limit $\mu_1 \to \infty$, rather than taking the limit $\mu_2 \to \infty$. Why are the answers so different?

s.4.5.33.

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/3 & 0 & 2/3 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

s.4.5.34. From the boundary conditions we know $G(m,0)$ and $G(0,n)$. Note that $M = 1$, since there just two stations. Since there are $N = 2$ jobs, we need $G(1,2)$.

$$\begin{aligned} G(1,2) &= f_1(0)G(0,2) + f_1(1)G(0,1) + f_1(2)G(0,0) \\ &= 1\cdot\mu_0^{-2} + \frac{1}{\mu_1}\frac{1}{\mu_0} + \frac{1}{\mu_1^2}1 \\ &= \frac{1}{16} + \frac{1}{12} + \frac{1}{9}. \end{aligned}$$

s.4.5.35. One way (that I accepted) is to use the formula for f, but that is actually not completely right, because the formula for f is based on the statement in the question, not the other way around.

The left-hand side is the rate at which jobs leave station i when k jobs are present, the right-hand side is the rate at which jobs arrive to station i when it contains $k-1$ jobs (so that once the job has arrived, the station contains k jobs.)

s.4.5.36. There can be different ways in which you model the system. One way is like this.

$$\begin{aligned} d_{k,1} &= \min\{c_{k,1}, Q_{k-1,1}\}\mathbb{1}_{Q_{k-1,2}<K} \\ Q_{k,1} &= Q_{k-1,1} + a_{k,1} - d_{k,1} \\ d_{k,2} &= \min\{c_{k,2}, Q_{k-1,2}\} \\ Q_{k,2} &= Q_{k-1,2} + d_{k,1} - d_{k,2}. \end{aligned}$$

Another interesting way is to take this for $d_{k,1}$.

$$d_{k,1} = \min\{c_{k,1}, Q_{k-1,1}, K - Q_{k-1,2}\}\, \mathbb{1}_{Q_{k-1,2}<K}.$$

Like this, the second queue can be replenished to level K, but cannot exceed level K.

If you forgot to relate $a_{k,2}$ and $d_{k,1}$ in some way: $-1/2$.

I accepted the following mistake: $d_{k,1} = \min\{c_{k,1}, Q_{k-1,1}, K - Q_{k-1,2}\}$. What happens if $K - Q_{k-1,2} < 0$? This can happen if $Q_{0,2} > K$.

s.4.5.37. See Figure 2.2 in Zijm's book.

Some students came up with very near to trivial networks. In those cases I gave 1/2 for this and the next two problems in total. (Consistently solving trivial problems cannot result in points...)

s.4.5.38. At $(0,0)$, the rates out and in are $\pi(0,0)\lambda = \pi(0,1)\mu_2$. At the rest of the boundary, $(\lambda + \mu_1)\pi(n_1,0) = \lambda\pi(n_1 - 1,0) + \mu_2\pi(n_1,1)$.

I accepted if you forgot the state $(0,0)$.

s.4.5.39. Fill in the stationary distribution $\pi(n_1,n_2) = (1-\rho_1)(1-\rho_2)\rho^{n_1}\rho^{n_2}$, in the above balance equation, and check that both sides cancel.

s.4.5.40. $P = \begin{pmatrix} 0 & 1 \\ 1/3 & 2/3 \end{pmatrix}$. Set $V_0 = 1$. Then solve for V_1 in the equation $V = VP$, where $V = (V_0, V_1)$, to see that $V_1 = 3$. Indeed, station 1 is visited three times as often as station 0.

If the row sums of P are not 1, this means that jobs leave the network. This is impossible in closed queueing networks. Hence, such P's are wrong. $-1/2$.

You should also know that the load-unload node, indicated as node 0, is only visited once during a routing. This assumption is in the book of Zijm, and implicit in the entire analysis. Hence $V_{00} > 0$: $-1/2$.

If you assume that $V_0 + V_1 = 1$: $-1/2$.

s.4.5.41.

$$\begin{aligned}
\mathsf{E}[W_0] &= \mathsf{E}[S_0] = 1 \\
\mathsf{E}[W_1] &= \mathsf{E}[S_1] = 2 \\
\mathsf{E}[W] &= \mathsf{E}[W_0] + V_1\,\mathsf{E}[W_1] = 1\cdot 1 + 3\cdot 2 = 7 \\
\mathrm{TH}_0 &= \frac{1}{\mathsf{E}[W]} = 1/7 \\
\mathrm{TH}_1 &= V_1\mathrm{TH}_0 = 3/7 \\
\mathsf{E}[L_1] &= \mathrm{TH}_1\,\mathsf{E}[W_1] = 3/7\cdot 2 = 6/7, \\
\mathsf{E}[L_0] &= 1/7\cdot 1 = 1/7
\end{aligned}$$

If you forgot to include the visit ratios: $-1/2$.

s.4.5.42. With the previous problem we have the case with one job. Now, with the arrival theorem,

$$\mathsf{E}[W_0(2)] = \mathsf{E}[L_0(1)]\,\mathsf{E}[S_0] + \mathsf{E}[S_0] = \frac{1}{7}1 + 1 = 8/7$$

$$\begin{aligned}
\mathsf{E}[W_1(2)] &= \mathsf{E}[L_1(1)]\,\mathsf{E}[S_1] + \mathsf{E}[S_1] = \frac{6}{7}2 + 2 = 26/7 \\
\mathsf{E}[W(2)] &= \mathsf{E}[W_0] + V_1\,\mathsf{E}[W_1] = 8/7 + 26/7 = 34/7 \\
\mathrm{TH}_0(2) &= \frac{1}{\mathsf{E}[W(2)]} = 7/34 \\
\mathrm{TH}_1(2) &= V_1\mathrm{TH}_0(2) = 21/34
\end{aligned}$$

In hindsight, it would have more interesting to ask for the expected number of jobs at station 1, i.e., $\mathsf{E}[L_1(2)]$.

s.4.5.43. There are some different ways to look at this. A correct answer depends on the motivation. An interesting answer would be like this. (Observe, to get a point for this problem, I expect a less complete answer than this from you.) In a real factory, the demand pattern has seasonal patterns. At times with low demand, the arrival rate of jobs is lower than the throughput that results from allowing w jobs on the floor (observe from the MVA that throughput is an increasing function of w). But if the arrival rate is smaller than the service rate of the bottleneck, i.e., $\rho < 1$, then the bottleneck station must be idle $1-\rho$ of the time. But when the bottleneck is idle, in particular, the total number of jobs in the closed queueing system must be less than w. The reason for this is that in closed queueing networks, queues build up in front of bottlenecks, and not so much in front of the non-bottleneck machines. But then the number of jobs on the factory floor cannot be constant.

Once again, other answers can certainly be ok too.

s.4.5.44.

$$P = \begin{pmatrix} 0 & 2/3 & 1/3 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

s.4.5.45. Set $V_0 = 1$. Then solve for V_1 and V_2 in the equation $V = VP$, where $V = (V_0, V_1, V_2)$. This gives

$$\begin{aligned}
V_0 &= 1 \\
V_1 &= \frac{2}{3} \\
V_2 &= \frac{1}{3}
\end{aligned}$$

If you would not set $V_0 = 1$, but solve $V_0 + V_1 + V_2 = 1$, then this is VERY wrong: -1/2 point.

s.4.5.46.

$$\begin{aligned}
\mathsf{E}[W_0] &= \mathsf{E}[S_0] = 1 \\
\mathsf{E}[W_1] &= \mathsf{E}[S_1] = 2 \\
\mathsf{E}[W_1] &= \mathsf{E}[S_3] = 3 \\
\mathsf{E}[W] &= \mathsf{E}[W_0] + V_1\,\mathsf{E}[W_1] + V_2\,\mathsf{E}[W_2] = 1 \cdot 1 + \frac{2}{3} \cdot 2 + \frac{1}{3} \cdot 3 = \frac{10}{3} \\
\mathrm{TH}_0 &= \frac{1}{\mathsf{E}[W]} = \frac{3}{10}
\end{aligned}$$

$$\begin{aligned}
\mathrm{TH}_1 &= V_1\mathrm{TH}_0 = \frac{2}{3}\frac{3}{10} = \frac{1}{5}\\
\mathrm{TH}_2 &= V_2\mathrm{TH}_0 = \frac{1}{3}\frac{3}{10} = \frac{1}{10}\\
\mathsf{E}[L_0] &= \mathrm{TH}_0\,\mathsf{E}[W_0] = \frac{3}{10}1 = \frac{3}{10}\\
\mathsf{E}[L_1] &= \mathrm{TH}_1\,\mathsf{E}[W_1] = \frac{1}{5}2 = \frac{2}{5}\\
\mathsf{E}[L_2] &= \mathrm{TH}_2\,\mathsf{E}[W_2] = \frac{1}{10}3 = \frac{3}{10}.
\end{aligned}$$

s.4.5.47. With the previous problem we have the case with one job. Now, with the arrival theorem,

$$\begin{aligned}
\mathsf{E}[W_0(2)] &= (\mathsf{E}[L_0(1)]+1)\,\mathsf{E}[S_0] = \frac{13}{10}1 = \frac{13}{10}\\
\mathsf{E}[W_1(2)] &= (\mathsf{E}[L_1(1)]+1)\,\mathsf{E}[S_1] = \frac{7}{5}2 = \frac{14}{5},\\
\mathsf{E}[W_2(2)] &= (\mathsf{E}[L_2(1)]+1)\,\mathsf{E}[S_2] = \frac{13}{10}3 = \frac{39}{10}\\
\mathsf{E}[W(2)] &= \mathsf{E}[W_0(2)] + V_1\,\mathsf{E}[W_1(2)] + V_2\,\mathsf{E}[W_2(2)] = \frac{13}{10} + \frac{2}{3}\frac{14}{5} + \frac{1}{3}\frac{39}{10} = \frac{67}{15}\\
\mathrm{TH}_0(2) &= \frac{2}{\mathsf{E}[W(2)]} = \frac{30}{67}\\
\mathrm{TH}_2(2) &= V_2\mathrm{TH}_0(2) = \frac{1}{3}\frac{30}{67} = \frac{10}{67}\\
\mathsf{E}[L_2(2)] &= \mathrm{TH}_2(2)\,\mathsf{E}[W_2(2)] = \frac{10}{67}\frac{39}{10} = \frac{39}{67}.
\end{aligned}$$

s.4.5.48. Line 8

s.4.5.49. Line 13 should read like this: `EW_tot += V[j] * EW[j]`.

s.4.5.50. Since `EW_tot` is updated for every j, the throughput depends on j. This is of course wrong. We should first compute $\mathsf{E}[W_j]$ for all stations, then compute the total time in the system, and then the throughput.

s.4.5.51. It is used to organize the events in the correct chronological order.

s.4.5.52. With classes we can organize functions (behavior) and state in a neat way. Then we make objects of classes to store the state for specific individuals, like jobs. We assemble the data of a simulation in these objects, so that we can do statistical analysis at the end.

For a detailed discussion, see the web, but that is for the really interested student.

BIBLIOGRAPHY

F. Baccelli and W.A. Massey. A sample path analysis of the $M/M/1$ queue. *Journal of Applied Probability*, 26(2):418–422, 1988.

G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*. John Wiley & Sons, 2006.

M. Capiński and T. Zastawniak. *Probability through Problems*. Springer Verlag, 2nd edition, 2003.

D.R. Cox, editor. *Renewal Theory*. John Wiley & Sons Inc, New York, 1962.

M. El-Taha and S. Stidham Jr. *Sample-Path Analysis of Queueing Systems*. Kluwer Academic Publishers, 1998.

R.W. Hall. *Queueing Methods for Services and Manufacturing*. Prentice Hall, 1991.

H.C. Tijms. *Stochastic Models, An Algorithmic Approach*. J. Wiley & Sons, 1994.

H.C. Tijms. *A First Course in Stochastic Models*. John Wiley & Sons, Chichester, 2003.

A.A. Yushkevich and E.B. Dynkin. *Markov Processes: Theorems and Problems*. Plenum Press, 1969.

NOTATION

a_k = Number of arrivals in the kth period

$A(t)$ = Number of arrivals in $[0,t]$

A_k = Arrival time of kth job

$\tilde{A}_k$ = Start of service of kth job

c_n = Service/production capacity in the nth period

d_n = Number of departures in the nth period

c = Number of servers

C_a^2 = Squared coefficient of variation of the inter-arrival times

C_s^2 = Squared coefficient of variation of the service times

$D(t)$ = Number of departures in $[0,t]$

$D_Q(t)$ = Number of customers/jobs that departed from the queue in $[0,t]$

D_k = Departure time of kth job

F = Distribution of the service time of a job

$L(t)$ = Number of customers/jobs in the system at time t

$Q(t)$ = Number of customers/jobs in queue at time t

$L_S(t)$ = Number of customers/jobs in service at time t

$\mathsf{E}[L]$ = Long run (time) average of the number of jobs in the system

$\mathsf{E}[Q]$ = Long run (time) average of the number of jobs in queue

$\mathsf{E}[L_S]$ = Long run (time) average of the number of jobs in service

$N(t)$ = Number of arrivals in $[0,t]$

$N(s,t)$ = Number of arrivals in $(s,t]$

$p(n)$ = Long-run time average that the system contains n jobs

Q_k = Queue length as seen by the kth job, or at the *end* of the kth period

S_k = Service time required by the kth job

$S(t)$ = Total service time available in $[0,t]$

S = Generic service time of a job

t = Time

W_k = Time in the system of kth job

$W_{Q,k}$ = Time in the queue of kth job

$\mathsf{E}[W]$ = Sample average of the sojourn time

$\mathsf{E}\left[W_Q\right]$ = Sample average of the time in queue

X_k = Inter-arrival time between job $k-1$ and job k

X = Generic inter-arrival time between two consecutive jobs

δ = Departure rate

λ = Arrival rate

μ = Service rate

$\pi(n)$ = Stationary probability that an arrival sees n jobs in the system

ρ = Load on the system

FORMULA SHEET

$$\rho = \lambda \frac{\mathsf{E}[S]}{c}$$

$$\mathsf{E}\left[W_Q\right] = \frac{C_a^2 + C_s^2}{2} \frac{\rho^{\sqrt{2(c+1)}-1}}{c(1-\rho)} \mathsf{E}[S]$$

Batching: $C_{sB}^2 = \dfrac{B\mathsf{V}[S_0] + \mathsf{V}[T]}{(B\,\mathsf{E}[S_0] + \mathsf{E}[T])^2}$

Nonpreemptive: $\mathsf{V}[S] = \mathsf{V}[S_0] + \dfrac{\mathsf{V}[T]}{B} + \dfrac{B-1}{B^2}(\mathsf{E}[T])^2$

Preemptive: $A = \dfrac{m_f}{m_r + m_f}, C_s^2 = C_0^2 + 2A(1-A)\dfrac{m_r}{\mathsf{E}[S_0]}$

$$C_{di}^2 = 1 + (1-\rho_i^2)(C_{ai}^2 - 1) + \frac{\rho_i^2}{\sqrt{c_i}}(C_{si}^2 - 1)$$

$$f_i(n_i) = \begin{cases} G(i)^{-1}(c_i\rho_i)^{n_i}(n_i!)^{-1}, & \text{if } n_i < c_i, \\ G(i)^{-1} c_i^{c_i} \rho_i^{n_i} (c_i!)^{-1}, & \text{if } n_i \geq c_i \end{cases}$$

$$\text{with } G(i) = \sum_{n=0}^{c_i-1} \frac{(c_i\rho_i)^n}{n!} + \frac{(c_i\rho_i)^{c_i}}{c_i!} \frac{1}{1-\rho_i}$$

$$\mathsf{E}[L_i] = \frac{(c_i\rho_i)^{c_i}}{c_i! G(i)} \frac{\rho_i}{(1-\rho_i)^2} + c_i\rho_i$$

$$f_i(n_i) = \frac{1}{\Pi_{k=1}^{n_i} \min\{k, c_i\}} \left(\frac{V_i}{\mu_i}\right)^{n_i}, i = 0 \ldots M$$

$$V_i = (VP)_i = \sum_{j=0}^{M} V_j P_{ji}$$

INDEX