# Analysis of Queueing Systems with Sample Paths and Simulation

Nicky D. van Foreest

March 2, 2017

# Contents

# 1 Introduction

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 of more customers get his/her shopping for free. Of course, there are some constraints: at least one of the cashier facilities has to 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.) 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 longer or equal than 4, 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 making of hamburgers. Typically, hamburgers are made-to-stock, in other words, they 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 some 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 on 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 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 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 with 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 house, 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*.

In these notes we will be primarily concerned with making models of queueing systems such that we can compute or estimate the above performance measures. Part of our work is to derive analytic models. 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. However, these models have severe shortcomings when it comes to analyzing real queueing systems, in particular when particular control rules have to be assessed. 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 cases simulation is a very useful and natural approach.

In the first part of these notes we concentrate on the analysis of *sample paths of a queueing process*. We 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, most 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 paths, and based on such sample paths we derive behavioral and statistical properties of the system. Thus, the performance measures defined for sample paths are precisely those used for simulation. Our aim is not to provide rigorous proofs for all results derived below; for the proofs and further background discussion we refer to **?**. As a consequence we tacitly assume in the remainder that results derived from the (long-run) analysis of a particular sample path are equal to their 'probabilistic counterpart'.

In the second part 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. As such, theory, simulation and algorithms form a nicely round out part of work. For this part we refer to book of Prof. Zijm; the present set of notes augment the discussion there.

I urge you to make *all* exercises in this set of notes. Many 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) carreer. Moreover, the exercises are meant to *illustrate* the material and to force you to *think* about it. Thus, the main text does not contain many examples; the exercises form the examples.

You'll notice that many 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. Next to this, I did not include the exercises with the intention that you would find them easy. The problems should be doalbe, but hard.

The solution manual is meant to prevent you from getting stuck and to help you increase your knowledge of probability, linear algebra, programming (analysis with computer support), and queueing in particular. Thus, read the solutions very carefully.

As a guideline to making the exercises I recommend the following approach. First read the notes. Then attempt to make a exercises for 10 minutes or so by yourself. If by that time you have not obtained a good idea on how to approach the problem, check the solution manual. Once you have understood the solution, try to repeat the arguments *with the solution manual closed*.

The meaning of the symbols in the margin of pages are as follows:

- The symbol in the margin means that you have to memorize the *emphasized concepts*.
- The symbol in the margin means that this question has a *hint*.
- The symbol that this question or its solution requires still some *work on my part*; you can skip it.

Finally I would like to acknowledge dr. J.W. Nieuwenhuis for our many discussions on the formal aspects of queueing theory and prof. dr. W.H.M. Zijm for allowing me to use the first few chapters of his book.

# 2 Single-Station Queueing Systems

In this chapter, we start with a discussion of the exponential distribution and the related Poisson process, as these concepts are perhaps the most important building blocks of queueing theory. With these concepts we can specify the arrival and service processes of customers, so that we can construct queueing processes and define performance measures to provide insight into the (transient and average) behavior of queueing processes. As it turns out, these constructions can be easily implemented as computer programs, thereby allowing to use simulation to analyze queueing systems. We then continue with developing models for various single-station queueing systems in steady-state, which is, in a sense to be discusses later, the long-run behavior of a stochastic system.[1] In the analysis we use sample-path arguments to count how often certain events occur as functions of time. Then we define probabilities in terms of limits of fractions of these counting processes. Another useful aspect of sample-path analysis is that the definitions for the performance measures are entirely constructive, hence by leaving out the limits, they provide expressions that can be right away used in statistical analysis of (simulations of) queueing systems. Level-crossing arguments will be of particular importance as we use these time and again to develop recursions by which we can compute steady-state probabilities of the queue length or waiting time process.

## 2.1 Exponential Distribution

As we will see in the sections to come, the modeling and analysis of any queueing system involves the specification of the (probability) distribution of the time between consecutive arrival epochs of jobs, or the specification of the distribution of the number of jobs that arrive in a certain interval. For the first case, the most common distribution is the exponential distribution, while for the second it is the Poisson distribution. For these reasons we start our discussion of the analysis of queueing system with these two exceedingly important distributions. In the ensuing sections we will use these distributions time and again.

As mentioned, one of the most useful models for the inter-arrival times of jobs assumes that the sequence $\{X_i\}$ of inter-arrival times is a set of *independent and identically distributed (i.i.d.)* random variables. Let us write $X$ for the generic random time between two successive arrivals. For many queueing systems, measurements of the inter-arrival times between consecutive arrivals show that it is reasonable to model an inter-arrival $X$ as an *exponentially distributed* random variable, i.e.,

$$\mathbb{P}\{X \le t\} = 1 - e^{-\lambda t} := G(t)$$

The constant $\lambda$ is often called the *rate*. The reason behind this will be clarified once we relate the exponential distribution to the Poisson process in Section 2.2. In the sequel we often write $X \sim \exp \lambda$ to mean that $X$ is exponentially distributed with rate $\lambda$.

Let us show with simulation how the exponential distribution originates. Consider $N$ people that regularly visit a shop. We assume that we can characterize the interarrival times $\{X_k^i, k =$

[1]This statement is, admittedly, vague, to say the least.

$1, 2, \ldots\}$ of customer $i$ by some distribution function, for instance the uniform distribution. Then,

$$A_k^i = A_{k-1}^i + X_k^i = \sum_{j=1}^n X_j^i,$$

is the arrival moment of the $k$th visit of customer $i$. Now the show owner 'sees' the superposition of the arrivals of the customers. One way to compute the arrival moments of all customers is to put all the numbers $\{A_k^i, k = 1, \ldots, n, i = 1, \ldots, N\}$ into one set, and sort these numbers in increasing order. This results in the (sorted) set of arrivals $\{A_k, k = 1, 2, \ldots\}$ at the shop, and then

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

must be the inter-arrival time between the $k-1$th and $k$th visit to the shop. Thus, starting from interarrival times of individual customers we constructed interarrival times as seen by the shop.

To plot the *empirical distribution function*, or the histogram, of the inter-arrival times at the shop, consider

$$\mathbb{P}_n(X \leq x) = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{X_i \leq x},$$

where[2] $\mathbb{1}_{X_i \leq x} = 1$ if $X_i \leq x$ and $\mathbb{1}_{X_i \leq x} = 0$ if $X_i > x$. Thus, for a simulation of duration $n$, this formula counts all interarrival times that are smaller than $x$.

We now compare $\mathbb{P}_n$ to the density of the exponential distribution, i.e., to $\lambda e^{-\lambda t}$ several simulation scenarios. As a first example, take $N = 1$ and let the computer generate $n = 100$ uniformly distributed numbers on the set $[4, 6]$. Thus, the time between two visits of an individual customer is somewhere between 4 and 6 hours. In a second simulation we take $N = 3$, and in the third, $N = 10$. The emperical distributions are shown, from left to right, in the three panels in Figure 2.1. The continuous curve is the graph of $\lambda e^{-\lambda x}$ where $\lambda = N/5$. Recall from Eq. (2.20) that when $N$ persons visits the shop, each with an average interarrival time of 5 hours, it must be that the arrival rate is $N/5$. As a second example we take the inter-arrival times to be normally distributed times with mean 5 and $\sigma = 1$. The results are shown in the second row of Figure 2.1.

As the graphs show, even when the customer population consists of 10 members each visiting the shop with an inter-arrival time that is quite 'far' from exponential, the distribution of the inter-arrival times as observed by the shop is very well approximated by an exponential distribution. Thus, for a real shop, with many thousands of customers, or a hospital, call center, in fact for nearly every system that deals with random demand, it seems reasonable to use the exponential distribution to model inter-arrival times. In conclusion, the main conditions to use an exponential distribution are: 1) arrivals have to be drawn from a large population, and 2) each of the arriving customers decides, independent of the others, to visit the system.

Another reason to use the exponential distribution is that an exponentially distributed random variable is *memoryless*, that is, $X$ is memoryless if it satisfies the property that

$$\mathbb{P}\{X > t + h | X > t\} = \mathbb{P}\{X > h\}.$$

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 $h$. Thus, no matter how long we

[2]More generally, $\mathbb{1}_A$ is the indicator function of the event $A$ meaning that $\mathbb{1}_A = 1$ if $A$ is true and $\mathbb{1}_A = 0$ otherwise.

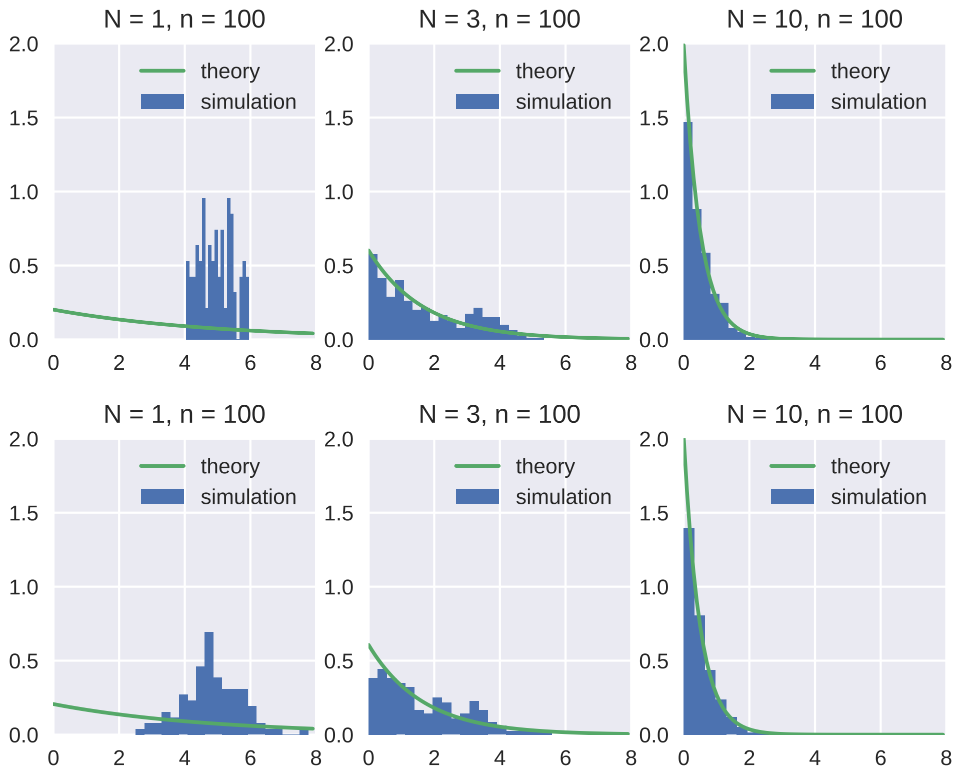


Figure 2.1: The interarrival process as seen by the shop owner. Observe that the green line intersects the $y$-axis at level $N/5$, which is equal to the arrival rate when $N$ persons visit the shop. The parameter $L$ is the simulation length, i.e., the number of arrivals per customer.

have been waiting for the next arrival to occur, the time that it occurs in the next $h$ seconds remains the same. This property seems to be vindicated also 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.

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 want to check this.

Finally, the reader should realize that it is simple, by means of computers, to generate exponentially distributed inter-arrival times. Thus, it is easy to use such exponentially distributed random variables to simulate queueing systems.

## 2.2 Poisson Distribution

In this section we provide a derivation of, and motivation for, the Poisson process, and clarify its relation with the exponential distribution at the end.

Consider a machine that fails occasionally. Let us write $N(s,t)$ for the number of failures occurring during a time interval of $(s,t]$. We assume, without loss of generality, that repairs are instantaneous. Clearly, as we do not know in advance how often the machine will fail, we model $N(s,t)$ as a random variable for all times $s$ and $t$.

Our first assumption is that the failure behavior of the machine does not significantly change over time. Then it is reasonable to assume that the expected number of failure is proportional to the length of the interval $T$. Thus, it is reasonable to assume that there exists some constant $\lambda$ such that

$$\mathbb{E}[N(s,t)] = \lambda(t-s) \tag{2.1}$$

The constant $\lambda$ is often called the *arrival rate*, or failure rate in this case.

The second assumption is that $\{N(s,t), s \leq t\}$ has *stationary* and *independent increments*. Stationarity means that the distribution of the number of arrivals are the same for all intervals of equal length. Formally, $N(s_1,t_1)$ has the same distribution as $N(s_2,t_2)$ if $t_2 - s_2 = t_1 - s_1$. Independence means, roughly speaking, that knowing that $N(s_1,t_1) = n$, does not help to make any predictions about $N(s_2,t_2)$ if the intervals $(s_1,t_1)$ and $(s_2,t_2)$ have no overlap.

To find the distribution of $N(0,t)$, let us split the interval $[0,t]$ into $n$ sub-intervals, all of equal length, and ask: 'What is the probability that the machine will fail in some given sub-interval.' By our second assumption, the failure behavior is constant over time. Therefore, the probability $p$ to fail in each interval should be equal. Moreover, if $n$ is large, $p$ must be small, for otherwise (2.2) could not be true. As a consequence, if the time intervals are very small, we can safely neglect the probability that two or more failures occur in one such tiny interval.

As a consequence, then, we can model the occurrence of a failure in some period $i$ as a Bernoulli distributed random variable $B_i$ such that $\mathbb{P}\{B_i = 1\}$ and $\mathbb{P}\{B_i = 0\} = 1 - \mathbb{P}\{B_i = 1\}$, and we assume that $\{B_i\}$ are independent. The total number of failures $N_n(t)$ that occur in $n$ intervals is then binomially distributed

$$\mathbb{P}\{N_n(t) = k\} = \binom{n}{k} p^k (1-p)^{n-k}. \tag{2.2}$$

In the exercises we ask you to use this to motivate that, in some appropriate sense, $N_n(t)$ converges to $N(t)$ for $n \to \infty$ such that

$$\mathbb{P}\{N(t) = k\} = e^{-\lambda t} \frac{(\lambda t)^k}{k!}. \tag{2.3}$$

We say that $N(t)$ is *Poisson distributed* with rate $\lambda$, and write $N(t) \sim P(\lambda t)$.

Moreover, if there are no failures in some interval $[0,t]$, then it must be that $N(t) = 0$ and $A_1$, i.e., the occurrence of the first failure, must be larger than $t$. Therefore,

$$\mathbb{P}\{A_1 > t\} = \mathbb{P}\{X_1 > t\} = \mathbb{P}\{N(t) = 0\} = e^{-\lambda t} \frac{(\lambda t)^0}{0!} = e^{-\lambda t}.$$

The relations we discussed above are of paramount importance in the analysis of queueing process. We summarize this by a theorem.

**Theorem 2.2.1.** A counting process $\{N(t)\}$ is a Poisson process with rate $\lambda$ if and only if the inter-arrival times $\{X_i\}$, i.e., the times between consecutive arrivals, are i.i.d. and $\mathbb{P}\{X_1 \leq t\} = 1 - e^{-\lambda t}$. In other words, $X_i \sim \exp(\lambda) \Leftrightarrow N(t) \sim P(\lambda t)$

## 2.3 Kendall's Notation to Characterize Queueing Processes

As will become apparent in Sections 2.4 and 2.5, the construction of any 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 interarrival times and service times are mutually independent.

To characterize the type of queueing process it is common to use the *abbreviation* $A/B/c/K$ where $A$ is the distribution of the interarrival times, $B$ the distribution of the services, $c$ the number of servers, and $K$ the size of the queue. 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-serve (FCFS).

Let us illustrate the shorthand $A/B/c/K$ with some examples:

- $M/M/1$: the distribution of the interarrival times is $M$emory-less, hence exponential, the service times are also $M$emoryless, and there is 1 server. As $K$ is unspecified, it is assumed to be infinite.
- $M/M/c$: A *multi-server* queue with $c$ servers in which all servers have the same capacity. Jobs arrive according to a Poisson process and have exponentially distributed processing times.
- $M(n)/M(n)/1$: the interarrival times are exponential, just as the service times, but the rates of the arrival and service processes may depend on the queue length $n$.
- $M/M/c/K$: interarrival times and process times are exponential, and the *system capacity* is $K$ jobs. Thus, the queue can contain at most $K-c$ jobs.[3]
- $M^X/M/1$: Customers arrive with exponentially distributed interarrival 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*.
- $M/G/1$: the interarrival times are exponentially distributed, the service times can have any $G$eneral distribution (with finite mean), and there is 1 server.
- $M/G/\infty$: exponential interarrival 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 wait in queue; upon arrival there is always a free server available.
- $M/D/1-LIFO$. Now job service times are $D$eterministic, and the service sequence is last-in-first-out (LIFO).
- $G/G/1$: generally distributed interarrival and service times, 1 server.

In the sequel we will use Kendall's notation frequently to distinguish the different queueing models. Ensure that you familiarize yourself with this notation.

[3] Sometimes, the $K$ stands for the capacity in the queue, not the entire system. The notation differs among authors. Due to the lack of consistency, I might also not be consistent; be warned.

## 2.4 Construction of Discrete-Time Queueing Processes

In this section we discuss a case as this provides real-life motivation to analyze queueing systems. After decribing the case, we we develop a set of recursions by which we can construct queueing systems in discrete time. As it turns out, this case is too hard to analyze by mathematical means, so that simulation is the best way forward. Interestingly, the structure of the simulation is very simple. As such, simulation is an exceedingly convincing tool to communicate the results of an analysis of a queueing system to managers (and the like).

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 is 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 that all 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 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 after each other, 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 less in quiet weeks. Assuming that the system is stable, i.e., the service capacity exceeds the 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.

To evaluate the effect of these suggestions on reducing the queueing dynamics we develop a simple simulator and a number of plots. The simulation of a queueing system involves the specification of its behavior over time. The easiest method to construct queueing processes, hence to develop simulations, is to 'chop up' time in periods and develop recursions for the behavior of the queue from period to period. Note that the length of such a period depends on the case 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.

Using fixed sized periods has it advantages as it does not require 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 need to be specified, which is useful since in many practical settings, e.g., production environments, it is easier to provide data in these terms then in terms of inter-arrival and service times. It is, however, necessary to make some careful choices about timing.

Let us *define*

$$\begin{aligned} a_k &= \text{number of jobs that arrive at period } k, \\ c_k &= \text{number of jobs that can be served during period } k, \\ d_k &= \text{number of jobs that depart the queue } \textit{during} \text{ period } k, \\ Q_k &= \text{number of jobs in queue at the } \textit{end} \text{ of period } k. \end{aligned} \tag{2.4}$$

In the sequel we also call $a_k$ the *batch* of job arrivals in period $k$. 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 during period $k$, in the latter case, they *cannot* be served during period $k$.

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

$$d_k = \min\{Q_{k-1}, c_k\}, \tag{2.5a}$$

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

$$Q_k = Q_{k-1} - d_k + a_k. \tag{2.5b}$$

As an example, suppose that $c_k = 7$ for all $k$, and $a_1 = 5$, $a_2 = 4$ and $a_3 = 9$; also $Q_0 = 8$. Then, $d_1 = 7$, $Q_1 = 8 - 7 + 5 = 6$, $d_2 = 6$, $Q_2 = 6 - 6 + 4 = 4$, $d_3 = 4$, $Q_3 = 4 - 4 + 9 = 9$, and so on.

Of course we are not going to carry out these computations by hand. Typically we use company data about the sequence of arrivals $\{a_k\}_{k=1,2,\ldots}$ and the capacity $\{c_k\}_{k=1,\ldots}$ and feed this data into a computer to compute the recursions (2.6). 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 computurs, to obtain numbers $a_1, \ldots, a_n$ for $n \gg 1000$, and so on.

We now show how to adapt (2.5) and (2.6) to the case introduced above.

As a first step we model the arrival process of patients as a Poisson process, c.f., Section 2.2. 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 slighly 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:

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

We next 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's 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.,

$$C = \begin{pmatrix} 0 & 2 & 2 & 2 & 2 & 0 & \dots \\ 2 & 0 & 2 & 2 & 2 & 2 & \dots \\ 3 & 3 & 0 & 3 & 3 & 3 & \dots \\ 4 & 4 & 4 & 0 & 4 & 4 & \dots \\ 4 & 4 & 4 & 4 & 0 & 4 & \dots \end{pmatrix},$$

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 (2.6) 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 Figure 2.2. It is apparent that suggestions 1 and 2 above do not significantly affect the behavior of the queue length process.

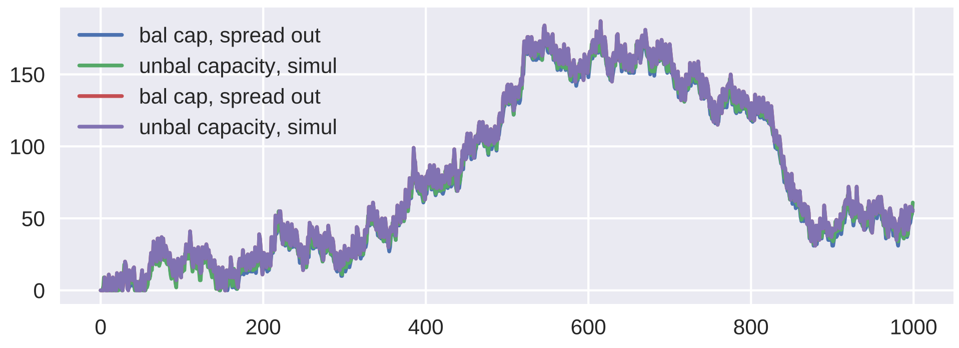


Figure 2.2: Effect of capacity and holiday plans

Now we consider suggestion 3, which comes down to doing more intakes when it is busy, and do less when it is quiet. A simple rule to implement this is by considering last week's queue $Q_{n-1}$: if $Q_{n-1} < 12$, i.e., the service capacity of one week, then do $e$ intakes less. Here, $e = 1$ or 2, or perhaps a larger number; it corresponds to the amount of control we want to exercise. When $Q_{n-1} > 24$, i.e., larger than two weeks of intakes, do $e$ intakes more. Let's consider three different control levels, $e = 1$, $e = 2$, and $e = 5$; thus in the last case all psychiatrists do one extra intake. 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.

Figure 2.3 shows a striking difference indeed. The queue does not explode any more, and already taking $e = 1$ has a large influence.

From the 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.

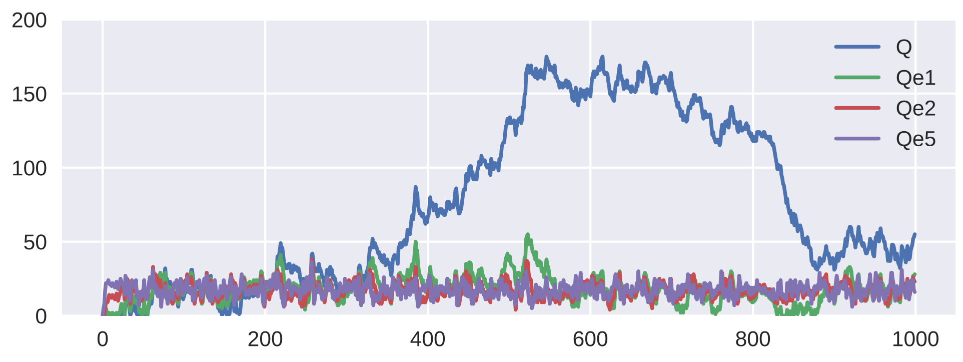


Figure 2.3: Controling the number of intakes

Thus, we made models for the arrival and service processes and then developed a simple recursion of the type (2.6) to *construct* a queueing process. Finally, for the analysis, we made plots of these processes.

In more abstract terms the study of queueing system is focussed 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 performance measures of relevance, such as the average waiting time, c.f. Section 2.8. 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 add service capacity with the aim to reduce the service times, hence reduce the average waiting time. With simulation it is easy to study the effect of, hence evaluate, such decisions.

Observe that, even with these simple recursions, we can obtain considerable insight into this, otherwise, very complicated controlled queueing process. (If the reader doubts the value of simulation, s/he should try to develop other mathematical methods to analyze multi-server queueing system with vacations, of which this is an example. Warning, do not even attempt: you'll fail as it is most probably too hard.) The simplicity of the recursions is deceitful, but with these simple recursions we can analyse many practical queueing situations. Together with students the author applied it numerous times, for instance,

- Should a certain hospital invest in a new MRI scanner to reduce waiting times?
- When to switch on and off a tin bath at an eletronics component factory?
- What is the effect of reducing the number of jobs in a paint factory?
- Post parcel routing in a post sorting center.
- Controlling inventories at various companies.
- Throughput time analysis at courts.

And so on, and so on. The recursions are indeed astonisingly useful. We therefore urge the reader to practice with this type of queueing modeling. The exercises below provide ample material for this purpose.

In passing we remark that yet more powerful simulations can be carried out with, so-called, event-based simulations. We refer to [**?**, Section 4.5] for a general description of this technique. The author applied this to quite complicated queueing networks:

- Analysis of the packaging process of a large beer factory.
- Performance analysis of large telecommunication networks
- Optimization of truck routing and queueing for a sugar factory.

Due to lack of time, we decided not to include this.

The reader should understand from the above case that, once we have the recursions, we can analyze the system and make plots to evaluate suggestions for improvement. Thus, getting the recursions is crucial to construct, i.e., model, queueing processes. For this reason, most of the exercises below focus on obtaining recursions for many different queueing systems.

## 2.5 Construction of the G/G/1 Queueing Process in Continuous Time

In the previous section we considered time in discrete 'chunks', minutes, hours, days, and so on. For given numbers of arrivals and capacity per period we use a set of recursions (2.6) to compute the departures and queue length per period. Another way to construct a queueing system is to consider inter-arrival times between consecutive customers and the service times each of these customers require. With this we obtain a description of the queueing system in continuous time. The goal of this section is to develop a set of recursions for the $G/G/1$ single-server queue.

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

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

$$A_k = \min\{t : A(t) \geq k\}, \tag{2.6}$$

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

$$X_k = A_k - A_{k-1}. \tag{2.7}$$

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 number of arrivals $\{A(t)\}$. 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

$$A(t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \leq t}, \tag{2.8a}$$

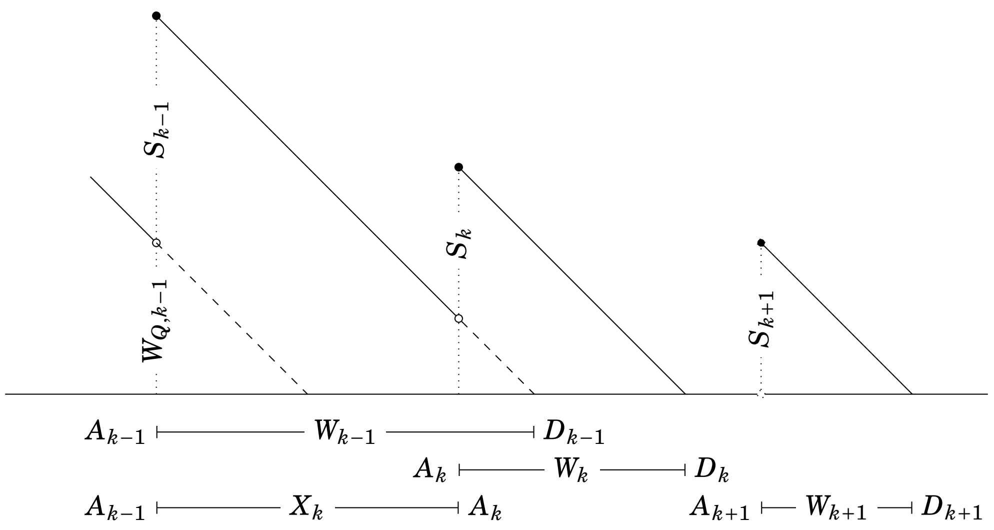


Figure 2.4: Construction of the $G/G/1$ queue in continuous time. The sojourn time $W_k$ of the $k$th job is sum of the work in queue $W_{Q,k}$ at its arrival epoch $A_k$ and its service time $S_k$; its departure time is then $D_k = A_k + W_k$. The waiting time of job $k$ is clearly equal to $W_{k-1} - X_k$. 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 $-1$.

where $\mathbb{1}$ is the indicator function. Thus, in the above we count all arrivals that occur up to time $t$. Another, equivalent, way to define $A(t)$ is

$$A(t) = \max\{k : A_k \le t\}. \tag{2.8b}$$

Clearly, we see that from the inter-arrival times $\{X_k\}$ it is possible to construct $\{A_k\}$ and $\{A(t)\}$, and the other way around, from $\{A(t)\}$ we can find $\{A_k\}$ and $\{X_k\}$. Figure 2.4 shows these relations graphically. To memorize it may be helpfull to write it like this:

$$\begin{aligned} A_k &: \mathbb{N} \to \mathbb{R}, \quad \text{job id (integer) to arrival time (real number)}, \\ A(t) &: \mathbb{R} \to \mathbb{N}, \quad \text{time (real number) to number of jobs (integer)}. \end{aligned}$$

To compute the departure times $\{D_k\}$ we proceed in stages. The first stage is to construct the *waiting time in queue* $\{W_{Q,k}\}$ as seen by the arrivals. In Figure 2.4 observe that the waiting time of the $k$th 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 $k$. In other words,

$$W_{Q,k} = [W_{Q,k-1} + S_{k-1} - X_k]^+, \tag{2.9}$$

where $[x]^+ = \max\{x, 0\}$. If we set $W_{Q,0} = 0$, we can compute $W_{Q,1}$ from this formula, and then $W_{Q,2}$ and so on.

The time job $k$ *leaves the queue and moves on to the server* is

$$D_{Q,k} = A_k + W_{Q,k},$$

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, $D_{Q,k}$ is 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

$$D_k = D_{Q,k} + S_k.$$

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

$$W_k = D_k - A_k = D_{Q,k} + S_k - A_k = W_{Q,k} + S_k, \tag{2.10}$$

where each of these equations has its own interpretation.

A bit of similar reasoning gives another recursion for $W_k$:

$$\begin{aligned} W_{Q,k} &= [W_{k-1} - X_k]^+, \\ W_k &= W_{Q,k-1} + S_k = [W_{k-1} - X_k]^+ + S_k. \end{aligned} \tag{2.11}$$

from which follows a recursion for $D_k$

$$D_k = A_k + W_k. \tag{2.12}$$

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

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

Once we have the arrival and departure processes it is easy to compute the *number of jobs in the system* at time $t$ as

$$L(t) = A(t) - D(t) + L(0), \tag{2.13}$$

where $L(0)$ is the number of jobs in the system at time $t = 0$; typically we assume that $L(0) = 0$. Thus, if we were to plot $A(t)$ and $D(t)$ as functions of $t$, then the difference $L(t)$ between the graphs of $A(t)$ and $D(t)$ tracks the number in the system, see Figure 2.5.

The number in the system as seen by the $k$th arrival is defined as

$$L_k = L(A_k), \tag{2.14}$$

since the $k$th job arrives at time $A_k$ and $\{L(s)\}$ is right-continuous.

Observe that in a queueing system, jobs can be in queue or in service. For this reason we distinguish between the number in the system $L(t)$, the number in queue $L_Q(t)$, and the number of jobs in service $L_s(t)$. If we know $D_Q(t)$, i.e. the number of jobs that departed from the queue up to time $t$, then

$$L_Q(t) = A(t) - D_Q(t)$$

must be the number of jobs in queue. The above expressions for $L(t)$ and $L_Q(t)$ then show that the number in service must be

$$L_s(t) = D_Q(t) - D(t) = L(t) - L_Q(t).$$

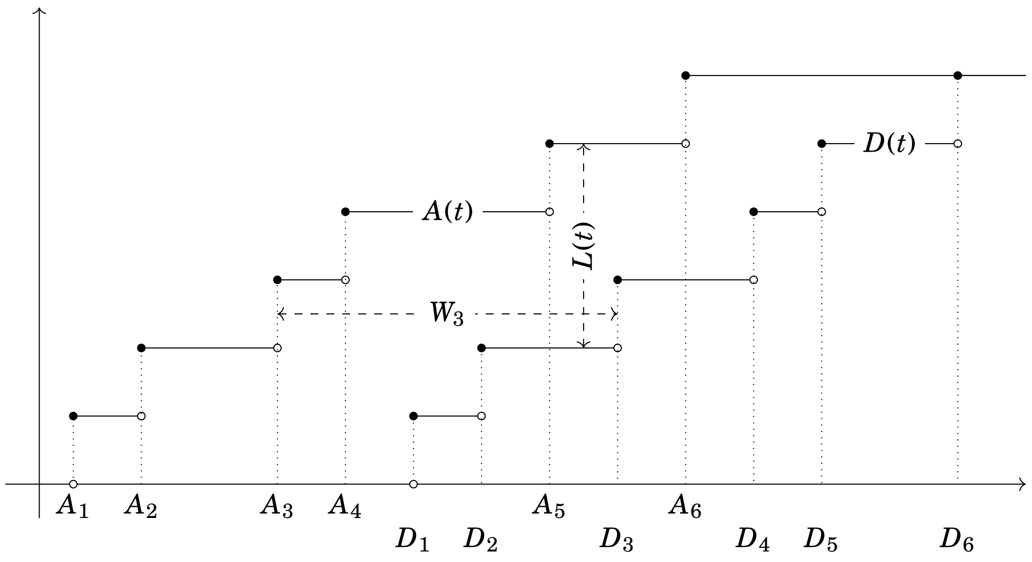


Figure 2.5: Relation between the arrival process $\{A(t)\}$, the departure process $\{D(t)\}$, the number in the system $\{L(t)\}$ and the waiting times $\{W_k\}$.

Finally, 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. Thus, the lines with slope $-1$ in Figure 2.4 show (a sample path of) the virtual waiting time.

Observe that, just as in Section 2.4, we have obtained a set of recursions by which we can run a simulation of a queueing process of whatever length we need, provided we have a sequence of have inter-arrival times $\{X_k\}$ and service times $\{S_k\}$. A bit of experimentation with computer programs such as *R* or python will reveal that this is easy.

## 2.6 Queueing Processes as Regulated Random Walks

In the construction of queueing processes as set out in section 2.4 we are given two sequences of i.i.d. random variables: the number of arrivals $\{a_k\}$ per period and the service capacities $\{c_k\}$. Assuming that jobs can be served in the period they arrive, the departure and queue length processes are generated by the recursions

$$\begin{aligned} Q_k &= [Q_{k-1} + a_k - c_k]^+, \\ d_k &= Q_{k-1} + a_k - Q_k, \end{aligned} \tag{2.15}$$

where $[x]^+ := \max\{x, 0\}$. Observe now that the relation for $Q_k$ shares a resemblance to a random walk $\{Z_k, k = 0, 1, \dots\}$ with $Z_k$ given by

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

Clearly, $\{Z_k\}$ is 'free', i.e., it can take positive and negative values, but $\{Q_k\}$ is restricted to the non-negative integers. In this section we show how to build the queueing process $\{Q_k\}$ from the random walk $\{Z_k\}$ by a device called a *reflection map*. Besides the elegance of this construction, the construction leads to spectacularly fast simulations. Moreover, we can use the probabilistic tools that have been developed for the random walk to analyze queueing systems. On example is the distribution of the time until an especially large queue 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.

In an exercise below we derive that $Q_k$ satisfies the relation

$$Q_k = Z_k - \min_{0\leq i\leq k} Z_i \wedge 0, \tag{2.16}$$

where $Z_k$ is defined by the above random walk and $a \wedge b = \min\{a,b\}$ [4]. This recursion leads to really interesting graphs, see Figures 2.6 and 2.7. We consider two examples. Let $a_k \sim B(0.3)$, i.e., $a_k$ is Bernoulli-distributed with success parameter $p = 0.3$, i.e., $\mathbb{P}\{a_k = 1\} = 0.3 = 1 - \mathbb{P}\{a_k = 0\}$, and $c_k \sim B(0.4)$. As a second example, we take $a_k \sim B(0.49)$ and construct the random walk as

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

Thus, if $a_k = 1$, the random walk increases by one step, while if $a_k = 0$, the random walk decreases by one step.

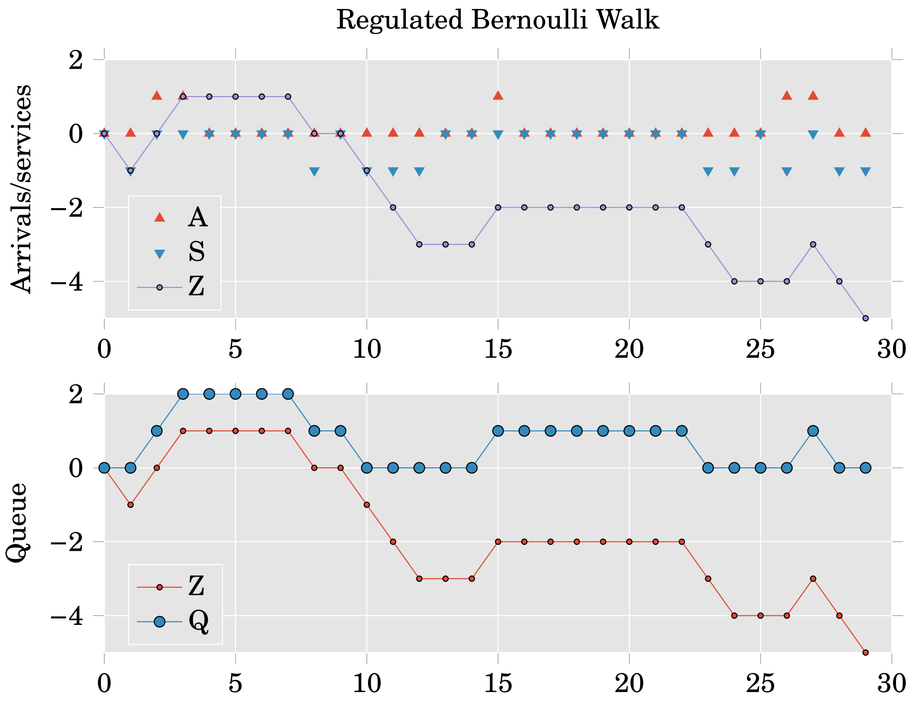


Figure 2.6: The upper panel shows a graph of the random walk $Z$. An upward pointing triangle corresponds to an arrival, a downward triangle to a potential service. The lower panel shows the queueing process $\{Q_k\}$ as a random walk with reflection.

Now that we have seen that random walks can be converted into queueing systems, we study in an exercise below the transient distribution, i.e., the distribution as a function of time, of the random walk $\{Z_k\}$. The aim of this exercise is to show that there is no simple function by

[4]For further detail, refer to **?**.

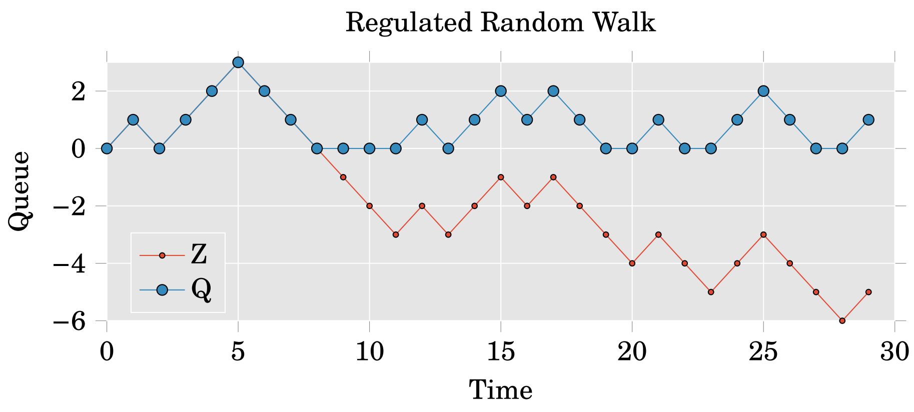


Figure 2.7: Another example of a reflected random walk.

which we can compute the transient distribution of this relatively simple random walk. Since a queueing process is typically a more complicated object (as it is a constrained random walk), our hopes to finding anything simple for the transient analysis of the $M/M/1$ queue should not be too high. And the $M/M/1$ is but the simplest queueing system; other queueing systems will be more complicated yet. We therefore give up the analysis of such transient queueing systems and we henceforth contend ourselves with the analysis of queueing systems in the limit as $t \to \infty$. This of course warrants two questions: what type of limit is actually meant here, and is such limiting situation reached in any reasonable time? We address these questions subsequently.

The *long-run limiting behavior* of a queueing system 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 $Q$ such that $Q_k \to Q$ in some sense? The answer to this question is in the affirmative, provided some simple stability conditions are satisfied, see Section 2.7. 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 $\{Q_k\}$ as random variables in their own right. Note that up to now we just considered each $Q_k$ as a *number*, i.e., a measurement or simulation of the queue length time of the $k$th period. Defining $Q_k$ 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 $\{Q_k\}$ are certainly not i.i.d., but, as should be apparent from Eq. (2.12), 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 $\{Q_k\}$ converges to some limiting distribution function $G$, say. Finally, it is necessary to show that it is possible to construct a random variable $Q$ that has $G$ as its distribution function. In this sense, then, we can say that $Q_k \to Q$. The random variable $Q$ is known as the *steady-state limit* of the sequence of random variables $\{Q_k\}$, and the distribution $G$ of $Q$ is known as the *limiting* or *stationary distribution* of $\{Q_k\}$.

In these notes we sidestep all these fundamental issues, as the details require measure theory and more advanced probability theory than we can deal with in this course. However, it can all be made precise.

Next, as an example, we address the rate of convergence of 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 Eq. (2.10). 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

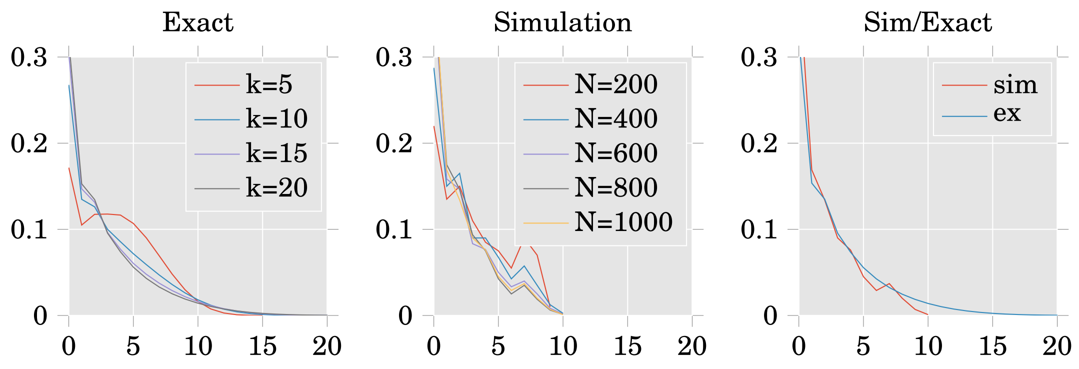


Figure 2.8: The density of $W_{Q,k}$ for $k = 5, 10, 15, 20$ computed by an exact method as compared the density obtained by simulation of different run lengths $N = 200, 400, \ldots 1000$. The right panel compares the exact density of $W_{Q,20}$ to the density obtained by simulation for $N = 1000$.

Eq. (2.10) to compute the *exact* distribution of $W_{Q,k}$ for $k = 1, 2, \ldots, 20$, c.f., the left panel in Figure 2.8. We see that when $k = 5$, the 'hump' of $\mathbb{P}\left\{W_{Q,5} = x\right\}$ 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

$$\mathbb{P}\left\{W_Q \leq x\right\} = n^{-1} \sum_{k=1}^{n} \mathbb{1}_{W_{Q,k} \leq x},$$

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

## 2.7 Rate Stability and Utilization

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, on the long run, jobs arrive faster than they can leave, the system must 'explode'. Thus, the first performance measures we need to estimate when analyzing a queueing system are the arrival and departure rate, and then we need to check that the arrival rate is smaller than the departure rate. In particular, the load, defined as the ratio of the arrival rate and service rate is of importance. In this section we define and relate these concepts. As a reminder, we keep the discussion in these notes mostly at an intuitive level, and refer to **?** for proofs and further background.

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.,

$$\lambda = \lim_{t\to\infty} \frac{A(t)}{t}. \tag{2.17}$$

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 i.i.d. and distributed as a generic random variable $X$ with finite mean $\mathbb{E}[X]$, we can construct $\{A_k\}$ and $\{A(t)\}$ as described in Section 2.5; the strong law of large numbers guarantees that the above limit exists.

Observe that at time $t = A_n$, precisely $n$ arrivals occurred. Thus, by applying the definition of $A(t)$ at the epochs $A_n$, we see that $A(A_n) = n$. Thus,

$$\frac{1}{n}\sum_{k=1}^{n} X_k = \frac{A_n}{n} = \frac{A_n}{A(A_n)}.$$

But since $A_n \to \infty$ if $n \to \infty$, it follows from Eq. (2.19) that the average inter-arrival time between two consecutive jobs is

$$\mathbb{E}[X] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} X_k = \lim_{n\to\infty} \frac{A_n}{A(A_n)} = \lim_{t\to\infty} \frac{t}{A(t)} = \frac{1}{\lambda}, \tag{2.18}$$

where we take $t = A_n$ in the limit for $t \to \infty$. In words the above states that the arrival rate $\lambda$ is the inverse of the expected inter-arrival time.

The development of the departure times $\{D_k\}$ is entirely analogous to that of the arrival times; we leave it to the reader to provide the details. As a result we can define the *departure rate* as

$$\lim_{t\to\infty} \frac{D(t)}{t} = \gamma \tag{2.19}$$

Assume now that there is a single server. Let $S_k$ be the required service time of the $k$th 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

$$U(t) = \sup\{n : U_n \le t\}.$$

and define the *service* or *processing rate* as

$$\mu = \lim_{t\to\infty} \frac{U(t)}{t}.$$

In the same way as we derived that $\mathbb{E}[X] = 1/\lambda$, we obtain for the expected (or average) amount of service required by an individual job

$$\mathbb{E}[S] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} S_k = \lim_{n\to\infty} \frac{U_n}{n} = \lim_{n\to\infty} \frac{U_n}{U(U_n)} = \lim_{t\to\infty} \frac{t}{U(t)} = \frac{1}{\mu}.$$

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 the number of arrivals, i.e., $D(t) \le A(t)$ for all $t$. Therefore,

$$\gamma := \lim_{t} \frac{D(t)}{t} \le \lim_{t} \frac{A(t)}{t} = \lambda. \tag{2.20}$$

We call a system *(rate) stable* if

$$\lambda = \gamma,$$

in other words, the system is stable if, on the long run, jobs leave the system just as fast as they arrive. Observe that if $\lambda > \gamma$, then the queue 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, $D(t) \leq U(t)$ for all $t$. Combining this with the fact that $\gamma \leq \lambda$, we get

$$\gamma \leq \min\{\lambda, \mu\}.$$

When $\mu \geq \lambda$ the above inequality reduces to $\gamma = \lambda$ for rate-stable systems. (It is interesting to prove this.) As it turns out, when $\mu = \lambda$ and $\mathbb{V}[S_n] > 0$ or $\mathbb{V}[X_n] > 0$ then $\lim_t L(t)/t$ does not necessarily exist. For this reason we henceforth require that $\mu > \lambda$.

The concept of *load* or *utilization*, denoted by the symbol $\rho$, is fundamental. One way to define it is as the limiting fraction of time the server is busy, i.e.,

$$\rho = \lim_{t\to\infty} \frac{1}{t} \int_0^t \mathbb{1}_{L(s)>0} \, \mathrm{d}s.$$

Interestingly, we can express this in terms of the arrival rate $\lambda$ and service rate $\mu$. Observe that

$$\sum_{k=1}^{A(t)} S_k \geq \int_0^t \mathbb{1}_{L(s)>0} \, \mathrm{d}s \geq \sum_{k=1}^{D(t)} S_k,$$

since $t$ can lie half way a service interval and $A(t) \geq D(t)$. As $A(t) \to \infty$ as $t \to \infty$,

$$\lim_{t\to\infty} \frac{1}{t} \sum_{k=1}^{A(t)} S_k = \lim_{t\to\infty} \frac{A(t)}{t} \frac{1}{A(t)} \sum_{k=1}^{A(t)} S_k = \lim_{t\to\infty} \frac{A(t)}{t} \cdot \lim_{t\to\infty} \frac{1}{A(t)} \sum_{k=1}^{A(t)} S_k = \lambda \mathbb{E}[S].$$

Applying similar limits to the other inequality gives

$$\lambda \mathbb{E}[S] \geq \rho \geq \gamma \mathbb{E}[S].$$

Hence, if $\gamma = \lambda$, $\rho = \lambda \mathbb{E}[S]$.

From the identies $\lambda^{-1} = \mathbb{E}[X]$ and $\mu^{-1} = \mathbb{E}[S]$, we get a further set of relations:

$$\rho = \lambda \mathbb{E}[S] = \frac{\lambda}{\mu} = \frac{\mathbb{E}[S]}{\mathbb{E}[X]}.$$

Thus, the load has also the interpretation as the rate at which jobs arrive multiplied by 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 $\rho = \mathbb{E}[S]/\mathbb{E}[X] < 1$ 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., $\mathbb{E}[S] < \mathbb{E}[X]$.

## 2.8 (Limits of) Emperical Performance Measures

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; the most important being the utilization $\rho$. We provide an overview of the relations between these performance measures in Figure 2.9.

With the construction of queueing processes in Section 2.5 we can compute the waiting time as observed by the first $n$, say, jobs. Thus, the average waiting time of the first $n$ arrivals is given by $n^{-1}\sum_{k=1}^{n} W_k$. We therefore define the *expected waiting time* as

$$\mathbb{E}[W] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} W_k, \tag{2.21}$$

and the expected time in queue as

$$\mathbb{E}\left[W_Q\right] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} W_{Q,k}. \tag{2.22}$$

Note that these performance measures are limits of *emperical* 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 $\mathbb{E}[W]$ is the average waiting time as 'seen by arrivals'. The *distribution of the waiting times at arrival times* can be found by counting:

$$\mathbb{P}\{W \leq x\} = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} \mathbb{1}_{W_k \leq x}. \tag{2.23}$$

Finally, the (sample) *average number of jobs* in the system as seen by arrivals is given by

$$\mathbb{E}[L] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} L_k, \tag{2.24}$$

where $L_k = L(A_k)$, i.e., the number in system at the arrival epoch of the $k$th job. The *distribution of* $\{L(t)\}$ *as seen by customers upon arrival*, is

$$\mathbb{P}\{L \leq m\} = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} \mathbb{1}_{L_k \leq m}. \tag{2.25}$$

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. Thus, if we simulate the queueing system up to time $t$, the *time-average number of jobs in the system* is given by

$$\mathscr{L}(t) = \frac{1}{t}\int_0^t L(s)\,\mathrm{d}s = \frac{1}{t}\int_0^t (A(s) - D(s))\,\mathrm{d}s, \tag{2.26}$$

where we use that $L(t) = A(t) - D(t)$ is the total number of jobs in the system at time $t$, c.f. Figure 2.5. Observe from the second equation that $\int_0^t L(s)\,\mathrm{d}s$ is the area enclosed between the graphs of $\{A(t)\}$ and $\{D(t)\}$. Assuming the limit exists for $t \to \infty$, we define

$$\mathbb{E}[L] = \lim_{t\to\infty} \frac{1}{t}\int_0^t L(s)\,\mathrm{d}s \tag{2.27}$$

Observe that, notwithstanding that the symbols are the same, this expectation need not be the same as (2.26), c.f. on of the exercises below. Next, define the following probability as the *time-average fraction of time the system contains at most m jobs*:

$$\mathbb{P}\{L \leq m\} = \lim_{t\to\infty} \frac{1}{t}\int_0^t \mathbb{1}_{L(s)\leq m}\,\mathrm{d}s. \tag{2.28}$$

Again, this probability need not be the same as what customers see upon arrival.

It is evident how to formulate the counterparts for the waiting time process $\{W(t)\}$.

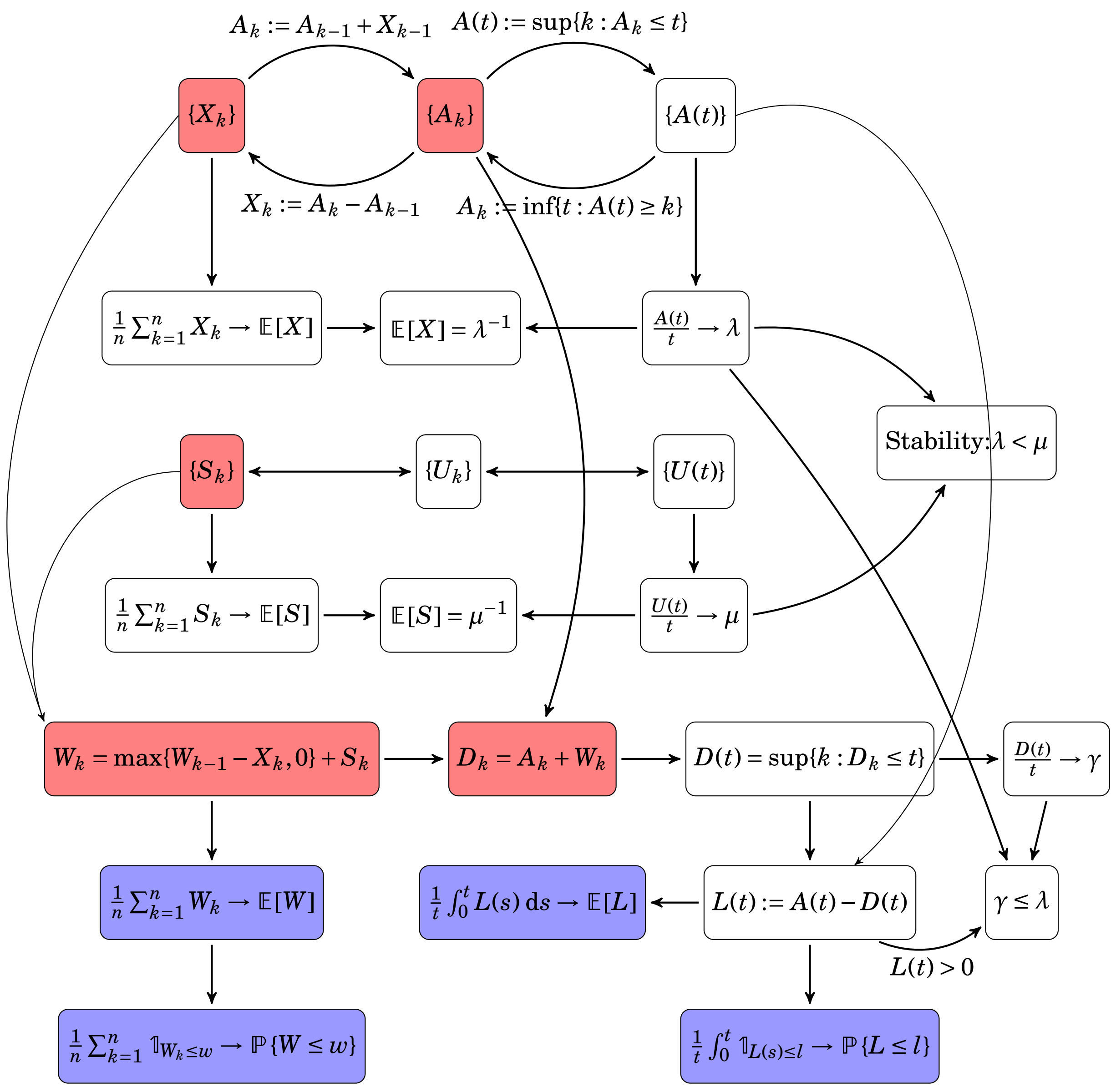

$A_k := A_{k-1} + X_{k-1}$
$A(t) := \sup\{k : A_k \leq t\}$
$\{X_k\}$
$\{A_k\}$
$\{A(t)\}$
$X_k := A_k - A_{k-1}$
$A_k := \inf\{t : A(t) \geq k\}$
$\frac{1}{n}\sum_{k=1}^n X_k \to \mathbb{E}[X]$
$\mathbb{E}[X] = \lambda^{-1}$
$\frac{A(t)}{t} \to \lambda$
Stability: $\lambda < \mu$
$\{S_k\}$
$\{U_k\}$
$\{U(t)\}$
$\frac{1}{n}\sum_{k=1}^n S_k \to \mathbb{E}[S]$
$\mathbb{E}[S] = \mu^{-1}$
$\frac{U(t)}{t} \to \mu$
$W_k = \max\{W_{k-1} - X_k, 0\} + S_k$
$D_k = A_k + W_k$
$D(t) = \sup\{k : D_k \leq t\}$
$\frac{D(t)}{t} \to \gamma$
$\frac{1}{n}\sum_{k=1}^n W_k \to \mathbb{E}[W]$
$\frac{1}{t}\int_0^t L(s)\,\mathrm{d}s \to \mathbb{E}[L]$
$L(t) := A(t) - D(t)$
$\gamma \leq \lambda$
$L(t) > 0$
$\frac{1}{n}\sum_{k=1}^n \mathbb{1}_{W_k \leq w} \to \mathbb{P}\{W \leq w\}$
$\frac{1}{t}\int_0^t \mathbb{1}_{L(s) \leq l} \to \mathbb{P}\{L \leq l\}$

Figure 2.9: Here we sketch the relations between the construction of the $G/G/1$ queue from the primary data, i.e., the interarrival times $\{X_k; k \geq 0\}$ and the service times $\{S_k; k \geq 0\}$, and different performance measure.

$$\begin{tikzcd} {} \arrow[r, dash] & A(t) \arrow[r] & \boxed{L(t)=A(t)-D(t)} \arrow[r, dash] & D(t) \arrow[r] & {} \end{tikzcd}$$

Figure 2.10: What goes in the box (i.e., $A(t)$) and has not yet left (i.e., $D(t)$) must be in the box, hence $L(t) = A(t) - D(t)$.

## 2.9 Level-Crossing and Balance Equations

Consider a system at which customers arrive and depart in single entities, such as customers in a shop or jobs at some machine. If the system starts empty, then we know that the number $L(t)$ is the system at time $t$ is equal to $A(t) - D(t)$, see Figure 2.10. Let us denote an arrival as an 'up-crossing' and a departure as a 'down-crossing'. Then, clearly $L(t)$ is the number of up-crossings up to time $t$ minus the number of down-crossings up to time $t$. If $L(t)$ remains finite, or more generally $\lim_t L(t)/t = 0$, then it must be that

$$\lambda = \lim_t \frac{A(t)}{t} = \lim_t \frac{D(t)+L(t)}{t} = \lim_t \frac{D(t)}{t} + \lim_t \frac{L(t)}{t} = \gamma.$$

Hence, when $L(t)/t \to 0$, the *up crossing rate* $\lim_t A(t)/t = \lambda$ is equal to the *down-crossing rate* $\lim_t D(t)/t = \gamma$. We will generalize these notions of up- and downcrossing in this section to derive the *stationary*, also know as *long-run time average* or *steady-state*, distribution $p(n)$ that the system contains $n$ jobs.

Let us say that the system is in *state $n$ when it contains $n$ jobs*, c.f. Figure 2.12, and the system *crosses* level $n$ when the state changes from $n$ to $n+1$, either 'from below' when an arrival occurs, or 'from above' when a departure occurs. The main result we derive in this section is the rate at which level $n$ is crossed from below and above must match, i.e.,

$$\lambda(n)p(n) = \mu(n+1)p(n),$$

where $\lambda(n)$ is the arrival rate in state $n$, $p(n)$ the fraction of time the system is in state $n$ and $\mu(n+1)$ the departure rate from $n+1$. To establish this 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. Once we have the proper definitions, the above result will follow straightaway.

Define[5]

$$A(n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \leq t} \mathbb{1}_{L(A_k-)=n}, \tag{2.29a}$$

$$Y(n,t) = \int_0^t \mathbb{1}_{L(s)=n} \, \mathrm{d}s, \tag{2.29b}$$

$$p(n,t) = \frac{1}{t}\int_0^t \mathbb{1}_{L(s)=n} \, \mathrm{d}s = \frac{Y(n,t)}{t}, \tag{2.29c}$$

respectively, so that $A(n,t)$ is the number of arrivals up to time $t$ that saw $n$ customers in the system at their arrival, $Y(n,t)$ is the total time the number in the system $L(s) = n$ during $[0,t]$ so that $p(n,t)$ is the fraction of time that $L(s) = n$ in $[0,t]$. (Recall in the definition of $A(n,t)$ that $L(t)$ is *right-continuous*.)

[5]It is common to write $f(x+) = \lim_{h\downarrow 0} f(x+h)$ for the right limit of a function $f$ at $x$ and $f(x-) = \lim_{h\downarrow 0} f(x-h)$ for the left limit. When $f(x-) = f(x+)$, $f$ is continuous at $x$. Since in queueing systems we are concerned with processes with jumps, we need to be quite particular about left and right limits at jump epochs.

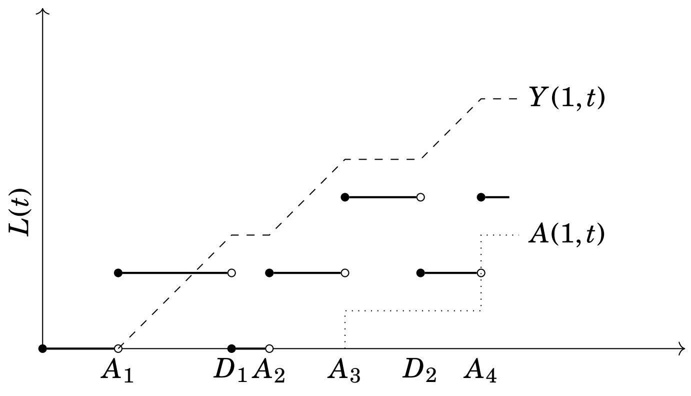


Figure 2.11: Plots of $Y(1,t)$ and $A(1,t)$. (For visual clarity, we subtracted ½ from $A(1,t)$, for otherwise its graph would partly overlap with the graph of $L$.)

Continuing with the above notions define

$$\lambda(n) = \lim_{t\to\infty} \frac{A(n,t)}{Y(n,t)}$$

as the *arrival rate in state* $n$. To clarify this notion, observe that $A(n,t)$ counts the number of arrivals that see $n$ jobs in the system, while $Y(n,t)$ tracks the amount of time the system contains $n$ jobs. Suppose that at $T$ a job arrives that sees $n$ in the system. Then $A(n,T) = A(n,T-)+1$, and this job arrives in an interval that it tracked by $Y(n,t)$, precisely because this job sees $n$ 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 $n$ divided by the time the system contains $n$ jobs.

Similarly, denoting by

$$D(n,t) = \sum_k \mathbb{1}_{D_k\leq t}\, \mathbb{1}_{L(D_k)=n}$$

the number of departures up to time $t$ that *leave* $n$ *customers behind*, we 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 2.12 shows how $A(n,t)$ and $\lambda(n)$ relate to $D(n+1,t)$ and $\mu(n)$.

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 $|A(n,t)-D(n,t)| \leq 1$ (Think about this.). From this observation it follows immediately that

$$\lim_{t\to\infty} \frac{A(n,t)}{t} = \lim_{t\to\infty} \frac{D(n,t)}{t}. \tag{2.30}$$

With this equation we can obtain two nice and fundamental identities. The first we develop now; the second follows in Section 2.12.

The rate of jobs that 'see the system with $n$ jobs' can be defined as $A(n,t)/t$. Taking limits we get

$$\frac{A(n,t)}{t} = \frac{A(n,t)}{Y(n,t)} \frac{Y(n,t)}{t} \to \lambda(n)p(n), \tag{2.31a}$$

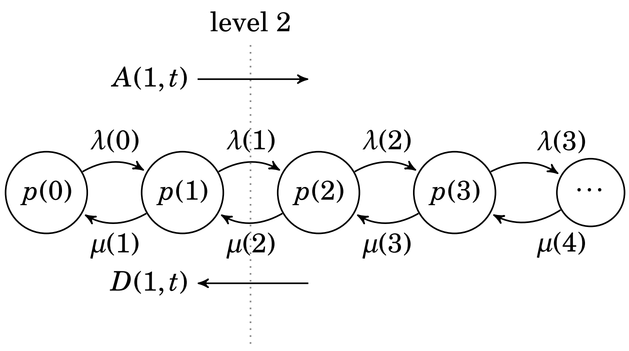


Figure 2.12: $A(1,t)$ counts the number of jobs up to time $t$ that saw 1 job in the system; thus, right after the arrival of such a job the system contains 2 jobs. Similarly, $D(1,t)$ counts the number of departures that leave 1 job behind. Each time $A(1,t)$ increases by one or $D(1,t)$ decreases by one, level 2 (the dotted line separating states 1 and 2) is crossed. The number of times this level is crossed from below must be the same (plus or minus 1) the number of times it is crossed from above.

where we use the above definitions for $\lambda(n)$ and $p(n)$. Similarly, the departure rate of jobs that leave $n$ jobs behind is

$$\frac{D(n,t)}{t} = \frac{D(n,t)}{Y(n+1,t)} \frac{Y(n+1,t)}{t} \to \mu(n+1)p(n+1). \tag{2.31b}$$

Combining this with (2.32) we arrive at *the level-crossing equations*

$$\lambda(n)p(n) = \mu(n+1)p(n+1). \tag{2.32}$$

This result turns out to be exceedingly useful. Once we can specify $\lambda(n)$ 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

$$p(n+1) = \frac{\lambda(n)}{\mu(n+1)} p(n). \tag{2.33}$$

Thus, if we have $p(n)$ we can compute $p(n+1)$, and so on. In other words, if $p(0)$ is known, than $p(1)$ follows, from which $p(2)$ follows, and so on. A straightaway iteration then leads to

$$p(n+1) = \frac{\lambda(n)\lambda(n-1)\cdots\lambda(0)}{\mu(n+1)\mu(n)\cdots\mu(1)} p(0). \tag{2.34}$$

Finally, to determine $p(0)$ we can use the fact that numbers $p(n)$ have to be normalized. Let the *normalization constant* be given by

$$G = 1 + \frac{\lambda(0)}{\mu(1)} + \frac{\lambda(0)\lambda(1)}{\mu(1)\mu(2)} + \cdots. \tag{2.35}$$

Then, $p(0) = G^{-1}$ and $p(n)$ follows from Eq. (2.36).

Let us now express a few important performance measures in terms of $p(n)$: the average number of items $\mathbb{E}[L]$ in the system and the fraction of time $\mathbb{P}\{L \le n\}$ the system contains at least $n$ jobs. 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

$$\mathscr{L}(t) = \frac{1}{t}\int_0^t \left(\sum_n n\, \mathbb{1}_{L(s)=n}\right) \mathrm{d}s = \sum_n \frac{n}{t}\int_0^t \mathbb{1}_{L(s)=n}\, \mathrm{d}s, \tag{2.36}$$

where we interchange the integral and the summation[6]. It then follows from Eq. (2.31c) that

$$\mathscr{L}(t) = \sum_n n\, p(n,t).$$

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), it follows that

$$\mathscr{L}(t) \to \sum_{n=0}^{\infty} n p(n) = \mathbb{E}[L], \text{ as } t \to \infty.$$

In words, $\mathscr{L}(t)$ converges to the long-run time average $\mathbb{E}[L]$ of the number in the system. In a loose sense we can say that $\mathbb{E}[L]$ is the average number in the system as perceived by the *server*. (Recall that this is not necessarily the same as what *arriving* jobs 'see'). Similarly, the probability that the system contains at least $n$ jobs is

$$\mathbb{P}\{L \geq n\} := \sum_{i=n}^{\infty} p(i).$$

From the above we conclude that from the probabilities $p(n)$, more specifically, from a specification of $\lambda(n)$ and $\mu(n)$, we can compute numerous performance measures. Thus, determining $p(n)$ from Eqs. (2.35) and (2.37) for concrete queueing examples is the task of the next sections.

Before we embark on this task, we note that the level-crossing cannot always be used as we do here. The reason is that it is not always natural, or easy, to decompose the state space 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, c.f. Figure 2.13. 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,

$$O(n,t) = A(n,t) + D(n-1,t),$$

is counts how often state $n$ is left either by an arrival in state $n$ or a departure to state $n-1$.

Of course, $|I(n,t) - O(n,t)| \leq 1$. Thus, from the fact that

$$\frac{I(n,t)}{t} = \frac{A(n-1,t)}{t} + \frac{D(n,t)}{t} \to \lambda(n-1)p(n-1) + \mu(n+1)p(n+1)$$

and

$$\frac{O(n,t)}{t} = \frac{A(n,t)}{t} + \frac{D(n-1,t)}{t} \to \lambda(n)p(n) + \mu(n)p(n)$$

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 $n \geq 0$ 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 that hold along any sensible sample path, i.e., counting differences, we obtain very useful statistical/probabilistic results after taking limits.

[6]This is allowed as the integrand is non-negative. More generally, the interested reader should check Fubini's theorem.

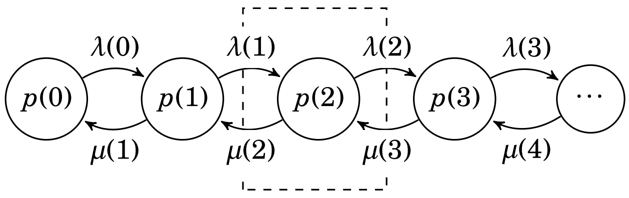


Figure 2.13: For the balance equations count how often a box around a state is crossed from inside and outside. On the long run the entering and leaving rates should be equal. For the example here, the rate out is $p(2)\lambda(2)+p(2)\mu(2)$ while the rate in is $p(1)\lambda(1)+p(3)\mu(3)$.

**Interpretation** The definitions in (2.31) 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 Eq. (2.31a) is 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 want 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 is $wnY(n,t)$ the total cost up to time $t$ to have $n$ jobs in queue, hence the total cost up up to time $t$

$$C(t) = w\sum_{n=0}^{\infty} nY(n,t),$$

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.

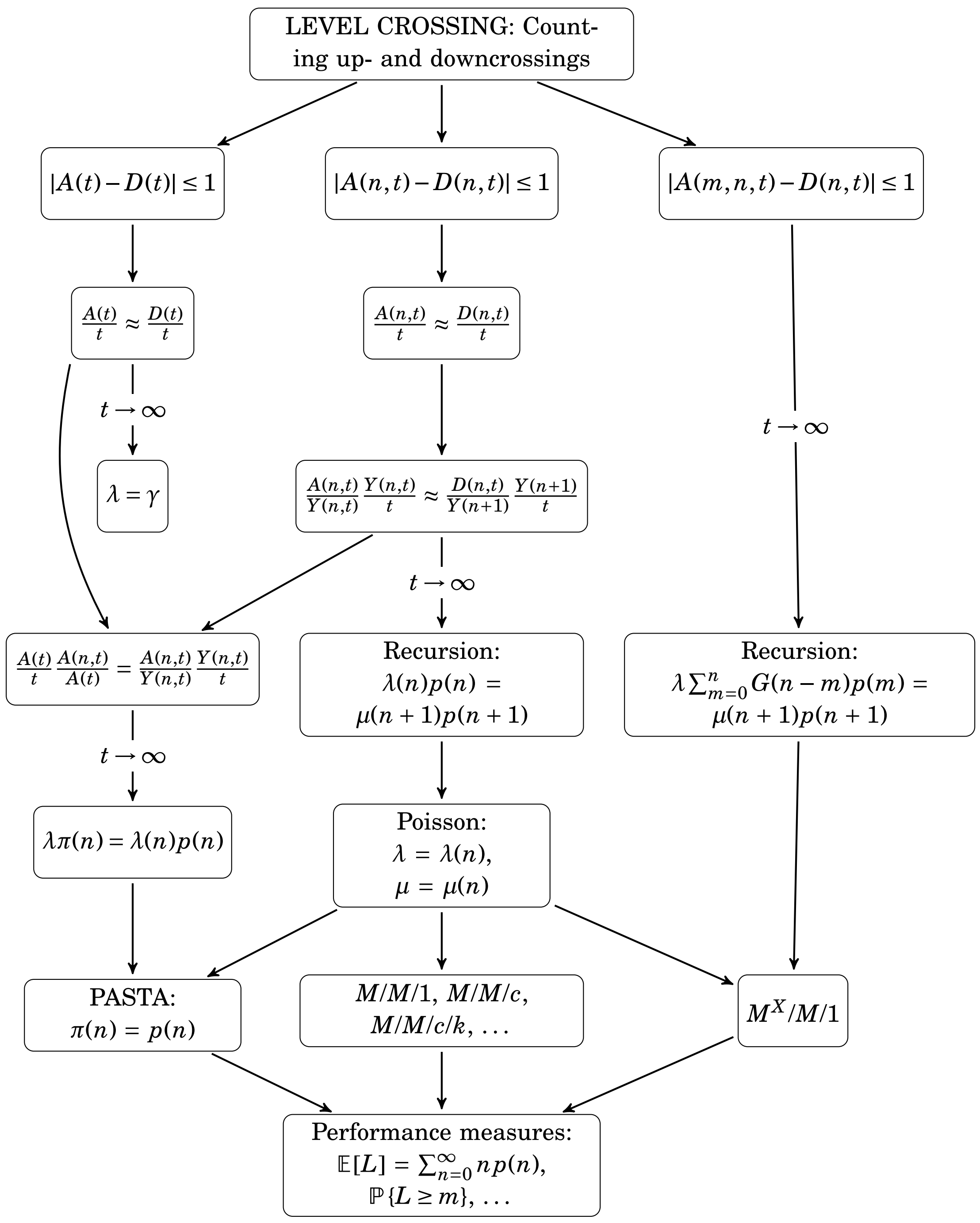

LEVEL CROSSING: Counting up- and downcrossings
$|A(t)-D(t)|\leq 1$
$|A(n,t)-D(n,t)|\leq 1$
$|A(m,n,t)-D(n,t)|\leq 1$
$\frac{A(t)}{t}\approx\frac{D(t)}{t}$
$\frac{A(n,t)}{t}\approx\frac{D(n,t)}{t}$
$t\to\infty$
$\lambda=\gamma$
$\frac{A(n,t)}{Y(n,t)}\frac{Y(n,t)}{t}\approx\frac{D(n,t)}{Y(n+1)}\frac{Y(n+1)}{t}$
$t\to\infty$
$t\to\infty$
$\frac{A(t)}{t}\frac{A(n,t)}{A(t)}=\frac{A(n,t)}{Y(n,t)}\frac{Y(n,t)}{t}$
Recursion: $\lambda(n)p(n)=\mu(n+1)p(n+1)$
Recursion: $\lambda\sum_{m=0}^{n}G(n-m)p(m)=\mu(n+1)p(n+1)$
$t\to\infty$
$\lambda\pi(n)=\lambda(n)p(n)$
Poisson: $\lambda=\lambda(n)$, $\mu=\mu(n)$
PASTA: $\pi(n)=p(n)$
$M/M/1$, $M/M/c$, $M/M/c/k$, …
$M^X/M/1$
Performance measures: $\mathbb{E}[L]=\sum_{n=0}^{\infty}np(n)$, $\mathbb{P}\{L\geq m\}$, …

Figure 2.14: 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.

## 2.10 M/M/1 queue

In the $M/M/1$ queue, one server serves jobs arriving with exponentially distributed interarrival times and each requiring an exponentially distributed processing time. With Eq. (2.35), i.e., $\lambda(n)p(n) = \mu(n+1)p(n+1)$ we can derive a number of important results for this queueing process.

Recall from Section 2.6 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 number in the $L(t) > 0$) 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, or the queue for that matter, $\lambda(n) = \lambda$ and $\mu(n) = \mu$ for all $n$. Thus, (2.35) 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, from the normalization condition

$$1 = \sum_{n=0}^{\infty} p(n) = p(0)\sum_{n=0}^{\infty}\rho^n = \frac{p(0)}{1-\rho},$$

so

$$p(0) = 1-\rho, \qquad\qquad p(n) = (1-\rho)\rho^n. \tag{2.37}$$

How can we use these equations? 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. Next, in the exercises below we derive that

$$\mathbb{E}[L] = \frac{\rho}{1-\rho}. \tag{2.38}$$

and

$$\mathbb{P}\{L \geq n\} = \rho^n. \tag{2.39}$$

Let us interpret these expressions. First of all, we only need estimates of $\lambda$ and $\mu$ to characterize the utilization of the server, the average queue length and the queue length distribution. In fact, only the ratio $\rho = \lambda/\mu$ is required. As such, assuming that interarrival times and services are exponentially distributed, by measuring the fraction of time the server is busy, we obtain an estimate for $1 - p(0)$, hence for $\rho$. With this, we can estimate $\mathbb{E}[L]$ and $\mathbb{P}\{L \geq n\}$. Next, the fact that $\mathbb{E}[L] \sim (1-\rho)^{-1}$ for $\rho \to 1$ implies that the average waiting time *increases asymptotically fast* to infinity when $\rho \to 1$. In practical terms, this means that striving for a high load results in (very) long average waiting times. As a consequence, this formula tells us that, in the design and operation of queueing systems, we should avoid the situation in which

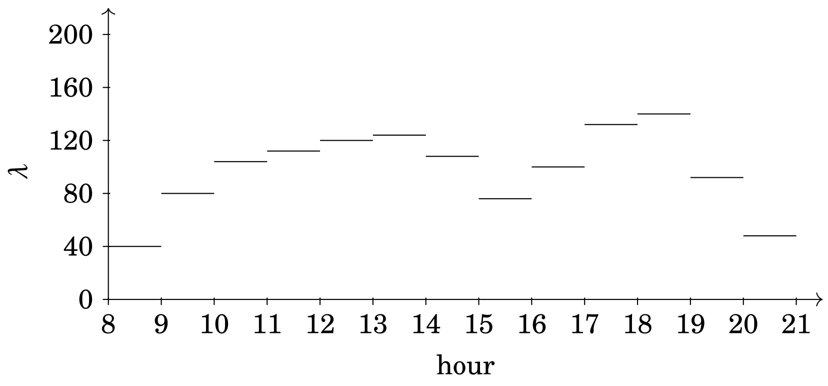


Figure 2.15: A demand profile of the arrival rate $\lambda$ modeled as constant over each hour.

$\rho \approx 1$. Thus, we need to make a trade-off between the server utilization $1 - p(0) = \rho$ and the waiting time. In high load regimes, the server is used efficiently, but the queue lengths are excessive. In any sensible system, we should keep $\rho$ quite a bit below 1, and accept lower utilization then might seem necessary.

The expression for $\mathbb{P}\{L \geq n\}$ shows that the probability that the queue length exceeds some threshold decreases exponentially fast (for any reasonable $\rho$). Moreover, 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 $\mathbb{P}\{L \geq 10\} = \rho^{10}$ is an estimator of the fraction of customers lost. Again, for the $M/M/1$ queue, the knowledge of $\rho$ suffices to compute this estimate.

In the context of inventory theory these equations are particularly useful, see Question **??**.

**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 such that the fraction of the time the queue length exceeds 10 is less than 1%.

The next step is to find the *relevant data*. For a supermarket this 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 can make a *demand profile*: the customer arrival rate per hour, c.f. Figure 2.15. 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, we model the arrival process as Poisson with an arrival rate that is constant during an hour and is specified by the demand profile, and we take the service time distribution as exponential with a mean of 1.5 minutes. We also *model* the behavior of the multi-server as a single fast server. Thus, we neglect any differences between a station with, for instance, 5 cashiers and a single server that works 5 times as fast as a normal cashier. As yet another simplification, we change the objective somewhat such that the number of jobs in the queueing system should not exceed 10. (Thus, the objective is no longer formulated in terms of queue lengths.)

To determine the required number of servers per hour let us consider the hour from 12 to 13. In the demand profile we see that the arrival rate in this hour is $\lambda_{12} = 120$ customers per hour. The objective $\mathbb{P}\{L > 10\} \leq 1\%$ translates into $\rho^{11} \leq 1\%$, hence, $\rho \leq 0.67$. Using that $\mathbb{E}[S] = 1.5$ minutes, we need the number of servers $c$ to be such that

$$\frac{120}{60} 1.5 \frac{1}{c} = \rho < 0.66,$$

hence $c \approx 5$. Interestingly, for this model we only need to be concerned with $\lambda \mathbb{E}[S]$ to meet the objective. Therefore $c$ is very simple to compute by the result

$$c \geq \lambda \frac{\mathbb{E}[S]}{0.66} = \lambda 2.3 \approx 2.5\lambda.$$

where $\lambda$ is the arrival rate (per minute, *not* per hour). Thus, the conversion of the demand profile to a *load* profile is trivial: divide the hourly arrival rate by 60 and multiply by 2.5. The load profile tells us the minimal number of cashiers needed per hour.

Now we need to *cover the load profile with service shifts*. This is typically not easy since shifts have to satisfy all kinds of rules, e.g.: 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 not 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, e.g.: shifts with hours after 18h are more expensive per hour; when the supermarket covers traveling costs, short shifts have higher marginal travel-ling costs; and so on.

The usual way to solve the covering problem 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 + indicate a working hour and − a break of an hour. Then generate shift types for each of these plans with starting times 8am, 9am, and on 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 shift 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 given by

$$\min \sum_i c_i x_i.$$

such that

$$\sum_i x_i \mathbb{1}_{t \in s_i} \geq \frac{\lambda_t}{20}$$

for all hours $t$ the shop is open and $\lambda_t$ is the demand for hour $t$.

Conceptually it is easy how to make this approach more accurate and realistic. The queueing model is too simple, so as start we could replace it by the multi-server model (with exponential service times) we will discuss in Section 2.11. If the data tells us that the exponential distribution for the service times is very inaccurate, we might use the results of Section 2.20, however, this only provides us with the mean waiting times for the $M/G/c$ queue, not the queue length distribution, which is a drawback. (The analytic results for the queue lenght distribution of the $G/G/c$ queue are not so easy and also not very accurate. Whether it makes sense to prefer these expressions over the closed-form results of the $M/M/c$ queue is not so easy to say. It depends on the context.) Finally, the objective should depend on the number of cashiers that are open. Assuming that customers spread evenly over the queues, the objective is such that no queue is longer than 3 say. The objective is therefore such that the queue length should be smaller than 3 times the number of cashiers $c$. Finally, in many shops, the cashier capacity is made dependent on the queue length; when the queues are long, stockers open extra cashier positions until the first server becomes idle. Thus, as there is spare capacity, the factor $2.5\lambda$ can be made smaller, perhaps set to $2\lambda$. However, this factor only changes the load profile, not the method to solve it.

Clearly, it does not require much effort to make the model very complicated, and it is easy to make it complicated to the extent that the model becomes useless. Mind: a good model is simple (but not too simple). So, notwithstanding the weaknesses of the above model, it is pretty powerful already.

## 2.11 M(n)/M(n)/1 Queue

As it turns out, many more single-server queueing situations can be modeled and analyzed by making a judicious choice of $\lambda(n)$ and $\mu(n)$ in (2.35), to wit, $\lambda(n)p(n) = \mu(n+1)p(n+1)$. For these queueing systems we just present the results. In the exercises we ask you to derive the formulas.

For the $M/M/1/K$, i.e., a system that cannot contain more than $K$ jobs, take

$$\lambda(n) = \begin{cases} \lambda, & \text{if } n \leq K, \\ 0, & \text{if } n > K, \end{cases}$$
$$\mu(n) = \mu.$$

Then,

$$p(n) = \frac{\rho^n}{G} \tag{2.40a}$$
$$G = \frac{1-\rho^{K+1}}{1-\rho}, \tag{2.40b}$$
$$P_{\text{loss}} = \mathbb{P}\{L = K\} = \frac{\rho^K}{G}. \tag{2.40c}$$

For the $M/M/c$ queue we can take

$$\lambda(n) = \lambda,$$
$$\mu(n) = \begin{cases} n\mu, & \text{if } n < c, \\ c\mu, & \text{if } n \geq c. \end{cases}$$

This model is also known as the *Erlang* $C$-formula. Define the load as

$$\rho = \frac{\lambda}{c\mu}.$$

Then,

$$p(n) = \frac{1}{G}\frac{(c\rho)^n}{n!}, n = 0, \dots, c-1 \tag{2.41a}$$

$$p(n) = \frac{1}{G}\frac{c^c\rho^n}{c!}, n = c, c+1, \dots \tag{2.41b}$$

$$G = \sum_{n=0}^{c-1}\frac{(c\rho)^n}{n!} + \frac{(c\rho)^c}{(1-\rho)c!}, \tag{2.41c}$$

$$\mathbb{E}\left[L_Q\right] = \sum_{n=c+1}^{\infty}(n-c)p(n) = \frac{(c\rho)^c}{c!G}\frac{\rho}{(1-\rho)^2}, \tag{2.41d}$$

$$\mathbb{E}[L_S] = \sum_{n=01}^{c-1} np(n) = \frac{\lambda}{\mu}. \tag{2.41e}$$

By taking the limit $c \to \infty$ we obtain the $M/M/\infty$ queue, in which case

$$\rho = \frac{\lambda}{\mu} \tag{2.42a}$$

$$p(n) = \frac{1}{G}\frac{\rho^n}{n!} \tag{2.42b}$$

$$G = e^\rho \tag{2.42c}$$

$$\mathbb{E}[L] = \mathbb{E}[L_S] = \rho. \tag{2.42d}$$

A model, often used to determine the number of beds at hospitals, is the $M/M/c/c$ model, also known as the Erlang $B$-formula; here jobs cannot be in queue, only in service, and the system has $c$ servers.

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 x

$$\lambda(n) = \begin{cases} \lambda, & \text{if } n = 0, \\ \lambda/2, & \text{if } n = 1, \\ \lambda/4, & \text{if } n = 2, \\ 0, & \text{if } n > 2, \end{cases}$$

and $\mu(n) = \mu$.

Observe that in the example with balking we made a subtle implicit assumption; in Section 2.12 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 $\mathbb{P}\{L \leq n\}$ 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 in Section 2.12.

## 2.12 Poisson Arrivals See Time Averages

Suppose the following limit exists:

$$\pi(n) = \lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} \mathbb{1}_{L(A_k-)=n},$$

where $\pi(n)$ is the long-run fraction of jobs that observe $n$ customers in the system at the moment of arrival. It is natural to ask whether $\pi(n)$ and $p(n)$ are related, that is, whether what customers see upon arrival is related to the time-average behavior of the system. We will derive a condition in this section under which $\pi(n) = p(n)$.

We can make some progress with this question by rewriting $\pi(n)$ in the following way. Since $A(t) \to \infty$ as $t \to \infty$, it is reasonable that[7]

$$\begin{aligned}\pi(n) &= \lim_{t\to\infty} \frac{1}{A(t)} \sum_{k=1}^{A(t)} \mathbb{1}_{L(A_k-)=n} \\ &= \lim_{t\to\infty} \frac{1}{A(t)} \sum_{k=1}^{\infty} \mathbb{1}_{A_k\leq t, L(A_k-)=n} \\ &= \lim_{t\to\infty} \frac{A(n,t)}{A(t)},\end{aligned} \tag{2.43}$$

where we use (2.31a) in the last row. But,

$$\frac{A(n,t)}{t} = \frac{A(n,t)}{A(t)}\frac{A(t)}{t} = \frac{A(t)}{t}\frac{A(n,t)}{A(t)} \to \lambda\pi(n), \quad \text{as } t \to \infty, \tag{2.44}$$

where we use Eq. (2.19) (i.e., $A(t)/t \to \lambda$). that Next, by Eq. (2.33),

$$\frac{A(n,t)}{t} = \frac{A(n,t)}{Y(n,t)}\frac{Y(n,t)}{t} \to \lambda(n)p(n), \quad \text{as } t \to \infty.$$

Thus

$$\lambda\pi(n) = \lambda(n)p(n), \tag{2.45}$$

from which follows our final result:

$$\lambda(n) = \lambda \iff \pi(n) = p(n),$$

So, why is this useful? Well, in words, it means that if the arrival rate does not depend on the state of the system, i.e., $\lambda = \lambda(n)$, the sample average $\pi(n)$ is equal to the time-average $p(n)$, i.e., $\pi(n) = p(n)$. But, when $\pi(n) = p(n)$, the customer perception at arrival moments, i.e., $\pi(n)$, is the same as the server perception, i.e., $p(n)$.

Of course, this property is not satisfied for any general queueing system, see the exercise below in which $X_k = 1$ hour for all $k$ and $S_k = 59$ minutes. However, it is true when the arrival process is Poisson. This fact is typically called *PASTA*: Poisson Arrivals See Time Averages.

Thus, when customers arrive in accordance to a Poisson process (so that the inter-arrival times are exponentially distributed), it must be that $\pi(n) = p(n)$, hence, for the $M/M/1$ queue, we see that

$$\pi(n) = p(n) = (1-\rho)\rho^n.$$

[7] See below for the proof.

With the above reasoning, we can also establish a relation between $\pi(n)$ and the statistics of the system as obtained by the departures, i.e., $\delta(n)$, to be defined presently. For this we turn again to Eq. (2.32), i.e., $|A(n,t) - D(n,t)| \leq 1$. To obtain Eq. (2.34) we divided both sides of this equation by the time the system spends in a certain state. We can also use another form:

$$\frac{A(t)}{t}\frac{A(n,t)}{A(t)} = \frac{A(n,t)}{t} \approx \frac{D(n,t)}{t} = \frac{D(t)}{t}\frac{D(n,t)}{D(t)}.$$

Taking limits at the left and right, we see again that the left hand becomes $\lambda\pi(n)$. For the right hand side, we use Eq. (2.21) and define, analogous to (2.45),

$$\delta(n) = \lim_{t\to\infty} \frac{D(n,t)}{D(t)}. \tag{2.46}$$

Thus, $\delta(n)$ is the long-run fraction of jobs that leave $n$ jobs *behind*. Clearly, then, if the limits exist, the right hand side tends to $\gamma\delta(n)$ as $t \to \infty$. Hence, for (queueing) systems in which customers arrive and leave as single units, we have

$$\lambda\pi(n) = \gamma\delta(n). \tag{2.47}$$

Moreover, if the system is rate-stable, i.e., the output rate $\gamma$ is equal to the input rate $\lambda$, we obtain

$$\lambda = \gamma \iff \pi(n) = \delta(n). \tag{2.48}$$

This means that the system as seen by arrivals, i.e., $\pi(n)$, is the same as what jobs leave behind, i.e., $\delta(n)$.

There is a subtle problem in (2.45) and the derivation of (2.46): $\pi(n)$ 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. The resolution lies in the *renewal reward theorem*. As this theorem is intuitive and very useful in its own right we state this theorem first, and then apply it to show (2.46). In Figure 2.16 we provide graphical motivation why this theorem is true; for the (simple) proof we refer to **?**.

**Theorem 2.12.1** (Renewal Reward Theorem, $Y = \lambda X$)**.** Suppose the counting process $\{N(t), t \geq 0\}$ is such that $N(t)/t \to \lambda$ as $t \to \infty$, where $0 < \lambda < \infty$. Let $\{Y(t), t \geq 0\}$ be a non-decreasing right-continuous (deterministic) process. Define $X_k = Y(T_k) - Y(T_{k-1})$, $k \geq 1$, where $T_k$ are the epochs at which $N$ increases, i.e., $N(t) = \{k : T_k \leq t\}$. Then $Y(t)/t$ has a limit iff $n^{-1}\sum_k^n X_k$ has a limit in which case $Y = \lambda X$, i.e.,

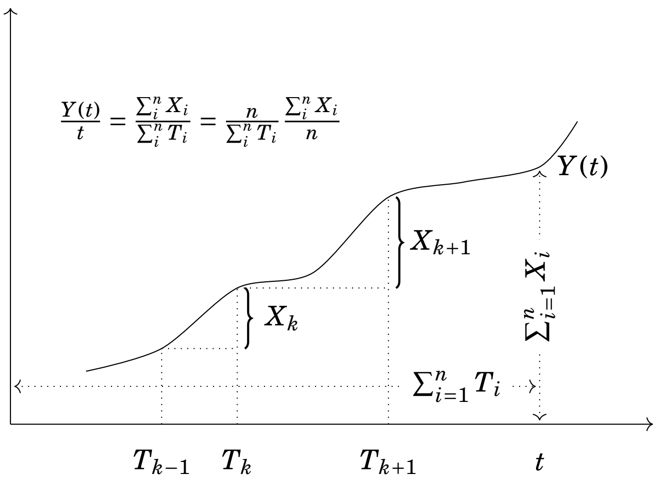
$$\lim_t \frac{Y(t)}{t} = Y \iff \lim_n \frac{1}{n}\sum_k^n X_k = X \implies Y = \lambda X.$$

Now, to show that (2.45) is valid, define

$$Y(t) := A(n,t) = \sum_{k=1}^{A(t)} \mathbb{1}_{L(A_k-)=n}$$
$$X_k := Y(A_k) - Y(A_{k-1}) = A(n, A_k) - A(n, A_{k-1}) = \mathbb{1}_{L(A_k-)=n}$$

Then,

$$X = \lim_{m\to\infty} \frac{1}{m}\sum_{k=1}^m X_k = \lim_{m\to\infty}\frac{1}{m}\sum_{k=1}^m \mathbb{1}_{L(A_k-)=n} = \pi(n).$$



$\frac{Y(t)}{t} = \frac{\sum_i^n X_i}{\sum_i^n T_i} = \frac{n}{\sum_i^n T_i} \frac{\sum_i^n X_i}{n}$
$Y(t)$
$X_{k+1}$
$X_k$
$\sum_{i=1}^n X_i$
$\sum_{i=1}^n T_i$
$T_{k-1}$
$T_k$
$T_{k+1}$
$t$

Figure 2.16: A graphical 'proof' of $Y = \lambda X$. Here $Y(t)/t \to Y$, $n/\sum_i^n T_i \to \lambda$ and $n^{-1}\sum_i^n X_i \to X$. (Observe that in the figure $X_k$ does not represent an interarrival time; instead it corresponds to the increment of the (graph of) $Y$ between two consecutive epochs $T_{k-1}$ and $T_k$ at which $Y$ is observed.)

Since $Y = \lim_t Y(t)/t = \lim_t A(n,t)/t$ it follows from the renewal reward theorem that

$$Y = \lambda X \implies \lim_t \frac{A(n,t)}{t} = \lambda X = \lambda \pi(n).$$

Thus, Eq. (2.46) follows from the renewal reward theorem.

In many (stochastic) systems the relation $Y = \lambda X$ is useful. As a further example, suppose customers pay on average an amount $X$ and customers arrive at rate $\lambda$, then the system earns money at rate $\lambda X = Y$.

## 2.13 Little's Law

There is an important relation between the average time $\mathbb{E}[W]$ a job spends in the system and the long-run time-average number $\mathbb{E}[L]$ of jobs that is contained in the system. This is Little's law:

$$\mathbb{E}[L] = \lambda \mathbb{E}[W]. \tag{2.49}$$

Here we provide a sketch of the proof, c.f., **?** for the details. 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 with defining a few intuitively useful concepts. Clearly, from (2.28),

$$\mathscr{L}(t) = \frac{1}{t}\int_0^t L(s)\,\mathrm{d}s = \frac{1}{t}\int_0^t (A(s) - D(s))\,\mathrm{d}s$$

is the time-average of the number of jobs in the system during $[0,t]$. Observe once again from the second equation that $\int_0^t L(s)\,\mathrm{d}s$ is the area enclosed between the graphs of $A(s)$ and $D(s)$.

The waiting time of the $k$th job is the time between the moment the job arrives and departs, that is

$$W_k = \int_0^\infty \mathbb{1}_{A_k \leq s < D_k} \, \mathrm{d}s.$$

We can actually relate $W_k$ to $\mathscr{L}(t)$, see Figure 2.5. Consider a departure time $T$ at which the system is empty. Observe that $A(T) = D(T)$, as at time $T$ all jobs that arrived up to $T$ also have left. Thus, for all jobs $k \leq A(T)$ we have that $D_k \leq A(T)$, so that we can replace the integration bounds in the above expression for $W_k$ by

$$W_k = \int_0^T \mathbb{1}_{A_k \leq s < D_k} \, \mathrm{d}s.$$

Moreover, if $s \leq T$,

$$L(s) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k \leq s < D_k} = \sum_{k=1}^{A(T)} \mathbb{1}_{A_k \leq s < D_k}.$$

Therefore,

$$\begin{aligned}\int_0^T L(s) \, \mathrm{d}s &= \int_0^T \sum_{k=1}^{A(T)} 1\{A_k \leq s < D_k\} \, \mathrm{d}s \\ &= \sum_{k=1}^{A(T)} \int_0^T 1\{A_k \leq s < D_k\} \, \mathrm{d}s = \sum_{k=1}^{A(T)} W_k.\end{aligned}$$

Thus, 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 $T$. From this,

$$\mathscr{L}(T) = \frac{1}{T} \int_0^T L(s) \, \mathrm{d}s = \frac{A(T)}{T} \frac{1}{A(T)} \sum_{k=1}^{A(T)} W_k.$$

Finally, assuming there are an infinite number of times $0 \leq T_i < T_{i+1} < \cdots$, $T_i \to \infty$, at which $A(T_i) = D(T_i)$ and the following limits exist

$$\mathscr{L}(T_i) \to \mathbb{E}[L], \qquad \frac{A(T_i)}{T_i} \to \lambda, \qquad \frac{1}{A(T_i)} \sum_{k=1}^{A(T_i)} W_k \to \mathbb{E}[W],$$

we obtain *Little's law*

$$\mathbb{E}[L] = \lambda \mathbb{E}[W].$$

As a useful first application, consider a server of the $G/G/1$ queue as a system by itself. Assume the system is rate-stable, for otherwise the above limits do not exists. The arrival rate at the server is $\lambda$ and the time a job remains in at the server is $\mathbb{E}[S]$. Thus, the averate number of jobs at the server is $\mathbb{E}[L_S] = \lambda \mathbb{E}[S]$. As $\lambda \mathbb{E}[S] = \rho$, we get $\mathbb{E}[L_S] = \rho$.

**Remark 2.13.1.** Sometimes (often?) students memorize Little's law in the wrong way. Thus, as an easy check, use the dimensions of the concepts: $\mathbb{E}[L]$ is an average *number*, $\lambda$ is a *rate*, i.e., *numbers per unit time*, and $\mathbb{E}[W]$ is waiting *time*. *Checking the dimensions* in the formula prevents painfull mistakes.

Note also that $\mathbb{E}[L] \neq L(t)$; to clarify, the expected number of items in the system is not necessarily equal to the number of items in the system at some arbitrary time $t$. Thus, Little's law need not hold at all moments in time; it is a statement about *averages*.

**Summary** Perhaps a summary of the most useful results and concepts of the previous sections are in order here:

- Arrival rate, departure rate, rate stability: $\lambda = \gamma$
- PASTA: $\lambda\pi(n) = \lambda(n)p(n)$,
- Recursions $\lambda(n)p(n) = \mu(n+1)p(n+1)$,
- Renewal reward: $Y = \lambda X$
- Little's law: $\mathbb{E}[L] = \lambda\mathbb{E}[W]$.

*Memorize these results.* We will use them often in the sequel, but in other courses these results will also prove very useful.

## 2.14 Some Useful Identities

With the PASTA property and Little's law we can derive a number of useful and simple results for the $M/G/1$ queue. Recall, to use the PASTA, we need to assume that jobs arrive as a Poisson process.

The fraction of time the server is empty is $1-\rho = p(0)$. By PASTA, $\pi(0) = p(0)$, hence the fraction of customers that enter an empty system is also $1-\rho$.

Suppose that at $A_k$, i.e., the arrival epoch of the $k$th job, the server is busy. The remaining service time $S_{r,k}$ as seen by job $k$ is the time between $A_k$ and the departure epoch of the job in service. If the server is free at $A_k$, we set $S_{r,k} = 0$. Define the average *remaining service time* as seen at arrival epochs by the limit

$$\mathbb{E}[S_r] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n} S_{r,k},$$

provided this limit exists. Note that $\mathbb{E}[S_r]$ includes the fraction of jobs that find the server idle. If we need the remaining service time for the jobs that see the server busy, use that a fraction $\rho$ sees the server occupied upon arrival, while a fraction $1-\rho$ sees a free server. Therefore

$$\mathbb{E}[S_r] = \rho\,\mathbb{E}[S_r \,|\, S_r > 0] + (1-\rho)\cdot\mathbb{E}[S_r \,|\, S_r = 0] = \rho\,\mathbb{E}[S_r \,|\, S_r > 0], \tag{2.50}$$

since, evidently, $\mathbb{E}[S_r \,|\, S_r = 0] = 0$. Observe that we used the PASTA property here.

Next, consider the waiting time in queue. 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, in turn, 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 $\mathbb{E}[L_Q]$. Thus, the expected time in queue is

$$\mathbb{E}[W_Q] = \mathbb{E}[S_r] + \mathbb{E}[L_Q]\,\mathbb{E}[S]. \tag{2.51}$$

Now, from Little's law we know that $\mathbb{E}[L_Q] = \lambda\mathbb{E}[W_Q]$. Using this,

$$\mathbb{E}[W_Q] = \mathbb{E}[S_r] + \lambda\mathbb{E}[W_Q]\,\mathbb{E}[S] = \mathbb{E}[S_r] + \rho\,\mathbb{E}[W_Q],$$

since $\rho = \lambda \mathbb{E}[S]$. But this gives for the $M/G/1$ queue that

$$\mathbb{E}\left[W_Q\right] = \frac{\mathbb{E}[S_r]}{1-\rho} = \frac{\rho}{1-\rho}\mathbb{E}[S_r \,|\, S_r > 0], \tag{2.52}$$

where we use (2.52) in the last equation.

The average waiting time $\mathbb{E}[W]$ in the entire system, i.e., in queue plus in service, becomes

$$\mathbb{E}[W] = \mathbb{E}\left[W_Q\right] + \mathbb{E}[S] = \frac{\mathbb{E}[S_r]}{1-\rho} + \mathbb{E}[S].$$

The situation can be significantly simplified for the $M/M/1$ queue as then the service times are also exponential, hence memoryless, implying that $\mathbb{E}[S_r \,|\, S_r > 0] = \mathbb{E}[S]$. Thus, for the $M/M/1$ queue,

$$\mathbb{E}[W] = \mathbb{E}\left[W_Q\right] + \mathbb{E}[S] = \frac{\mathbb{E}[S]}{1-\rho}.$$

Another way to derive the above result is to conclude from PASTA that the expected number of jobs in the system at an arrival is $\mathbb{E}[L]$. Since all these jobs require an expected service time $\mathbb{E}[S]$, including the job in service by the memory-less property, the time in queue is $\mathbb{E}[L]\,\mathbb{E}[S]$. The time in the system is then the waiting time plus the service time, hence,

$$\mathbb{E}[W] = \mathbb{E}[L]\,\mathbb{E}[S] + \mathbb{E}[S] = \lambda \mathbb{E}[W]\,\mathbb{E}[S] + \mathbb{E}[S] = \rho \mathbb{E}[W] + \mathbb{E}[S], \tag{2.53}$$

where we use Little's law $\mathbb{E}[L] = \lambda \mathbb{E}[W]$. Hence,

$$\mathbb{E}[W] = \frac{\mathbb{E}[S]}{1-\rho}.$$

Also,

$$\mathbb{E}\left[W_Q\right] = \mathbb{E}[L]\,\mathbb{E}[S] = \lambda \mathbb{E}[W]\,\mathbb{E}[S] = \rho \mathbb{E}[W] = \frac{\rho}{1-\rho}\mathbb{E}[S] = \frac{\rho^2}{1-\rho}\frac{1}{\lambda}, \tag{2.54}$$

which is consistent with our earlier result.

For the average queue length we use Little's law. Then

$$\mathbb{E}[L] = \lambda \mathbb{E}[W] = \frac{\lambda \mathbb{E}[S]}{1-\rho} = \frac{\rho}{1-\rho},$$
$$\mathbb{E}\left[L_Q\right] = \lambda \mathbb{E}\left[W_Q\right] = \frac{\rho^2}{1-\rho}.$$

Finally, the expected number of jobs in service $\mathbb{E}[L_s]$, which is equal to the expected number of busy servers, must be

$$\mathbb{E}[L_s] = \mathbb{E}[L] - \mathbb{E}\left[L_Q\right] = \frac{\rho}{1-\rho} - \frac{\rho^2}{1-\rho} = \rho,$$

again in accordance with our earlier result.

## 2.15 $\mathbf{M^X/M/1}$ Queue: Expected Waiting Time

It is not always the case that jobs arrive in single units, they can also arrive as batches. For instance, cars and busses arrive at a fast food restaurant so that the batch size consists of the number of people in the vehicle. In this section we derive a formula to compute the expected queue length for this queueing process.

Assume that jobs arrive as a Poisson process with rate $\lambda$ 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 number of items that job $k$ brings into the system. We assume that $\{B_k\}$ is an i.i.d. sequence of discrete random variables distributed as a generic random variable $B$ such that $\mathbb{P}\{B=k\}=f(k)$, where $f(k)$ is a given set of probabilities. We write

$$G(k)=\mathbb{P}\{B>k\}=\sum_{m=k+1}^{\infty} f(m),$$

for the *survivor function* of $B$. We also assume that the service time of each item is exponentially distributed with average $1/\mu$. Thus, the average time to serve a job is $\mathbb{E}[B]\,\mathbb{E}[S]=\mathbb{E}[B]/\mu$. This queueing situation denoted by the shorthand $M^X/M/1$.

The first criterion we must check for the $M^X/M/1$ queue is the stability: the service rate must be larger than the arrival rate of work. To determine the latter, the total number of items $N(t)$ arrived up to time $t$ must be equal to the number of arrivals $A(t)$ up to time $t$ times the batch size of each arrival, i.e.,

$$N(t)=\sum_{k=0}^{\infty} B_k \mathbb{1}_{A_k\leq t}=\sum_{k=0}^{A(t)} B_k.$$

The (stochastic) process $\{N(t)\}$ is known as the *compound Poisson process*. Clearly, the rate at which items arrive is approximately

$$\frac{N(t)}{t}=\frac{A(t)}{t}\frac{1}{A(t)}\sum_{i=k}^{A(t)} B_k.$$

As in the limit $A(t)/t\to\lambda$ and $(A(t))^{-1}\sum_{i=k}^{A(t)} B_k\to\mathbb{E}[B]$, we see that

$$\frac{N(t)}{t}\to\lambda\,\mathbb{E}[B].$$

Thus, for stability, it is necessary that the service rate $\mu=1/\mathbb{E}[S]$ for items is larger than the rate at which items arrive. Hence we require that the load is such that

$$\rho=\lambda\,\mathbb{E}[B]\,\mathbb{E}[S]=\frac{\lambda\,\mathbb{E}[B]}{\mu}<1.$$

For the average waiting time, we can use the derivation of $\mathbb{E}\left[W_Q\right]=\mathbb{E}[S_r]/(1-\rho)$ of Section 2.14 for inspiration. Suppose a batch finds $\mathbb{E}[L]$ items in the system upon arrival. By the memoryless property of the service distribution, the expected remaining service time of the item in service (if any) is $\mathbb{E}[S]$. Therefore the expected time the entire batch, i.e., the job, spends in queue is

$$\mathbb{E}\left[W_{Q,b}\right]=\mathbb{E}[L]\,\mathbb{E}[S];$$

compare Eq. 2.56. Note that this is not the same as $\mathbb{E}\left[W_Q\right]$, which is the expected time an *item* spends in queue. If $B_r$ is the number of items of the batch currently in service ($B_r = 0$ if the server is idle), and $L_{Q,b}$ the number of batches in queue, then

$$\mathbb{E}[L] = \mathbb{E}\left[L_{Q,b}\right] \mathbb{E}[B] + \mathbb{E}[B_r].$$

With Little's law, $\mathbb{E}\left[L_{Q,b}\right] = \lambda \mathbb{E}\left[W_{Q,b}\right]$, hence

$$\mathbb{E}\left[W_{Q,b}\right] = \lambda \mathbb{E}[S] \mathbb{E}[B] \mathbb{E}\left[W_{Q,b}\right] + \mathbb{E}[S] \mathbb{E}[B_r] = \rho \mathbb{E}\left[W_{Q,b}\right] + \mathbb{E}[S] \mathbb{E}[B_r],$$

hence,

$$\mathbb{E}\left[W_{Q,b}\right] = \frac{\mathbb{E}[S]}{1-\rho} \mathbb{E}[B_r].$$

And then, from the first equation,

$$\mathbb{E}[L] = \frac{\mathbb{E}\left[W_{Q,b}\right]}{\mathbb{E}[S]} = \frac{\mathbb{E}[B_r]}{1-\rho}.$$

Below we prove that

$$\mathbb{E}[B_r] = \rho \frac{\mathbb{E}\left[B^2\right]}{2\mathbb{E}[B]} + \frac{\rho}{2}. \tag{2.55}$$

In an exercise below we rewrite this to

$$\mathbb{E}[L] = \frac{\mathbb{E}[B_r]}{1-\rho} = \frac{1+C_s^2}{2} \frac{\rho}{1-\rho} \mathbb{E}[B] + \frac{1}{2} \frac{\rho}{1-\rho}, \tag{2.56}$$

where

$$C_s^2 = \frac{\mathbb{V}[B]}{(\mathbb{E}[B])^2},$$

is the square coefficient of variation of the batch sizes. Observe that this is nearly the same as $M/G/1$ waiting time formula 2.61, a result we derive in Section 2.16.

Thus, to compute the average number of items in the system, we only need to know the first and second moment (or the variance) of the batch size $B$. Thus, no matter how 'complicated' the distribution of $B$, when its second moment exists, the average queue length and waiting time can be computed with the above result.

We now turn to proving (2.57) with sample-path arguments and counting; it is an elegant line of reasoning. First concentrate on the time the server is busy, i.e., just remove all idle times, and plot the remaining job size during the service of the job, c.f, Figure 2.17. Second, define $R$ as the remaining number of items to be served of the job in service at some arbitrary point in time. Let us show that

$$\mathbb{P}\{R = i\} = \frac{\mathbb{P}\{B \geq i\}}{\mathbb{E}[B]} = \frac{G(i-1)}{\mathbb{E}[B]}.$$

In Figure 2.17 concentrate on level $i$. Only jobs whose initial batch size is larger or equal to $i$ will produce, during its service, a remaining number of items equal to $i$. Thus, by counting, we see in Figure 2.17 that $\sum_{k=1}^n \mathbb{1}_{B_k \geq i}$ is the number of times there are precisely $i$ remaining items. We also see that $\sum_{k=1}^n B_k$ is the total time the server is busy. Thus, the fraction of time there are $i$ remaining items is

$$\frac{\sum_{k=1}^n \mathbb{1}_{B_k \geq i}}{\sum_{k=1}^n B_k}.$$

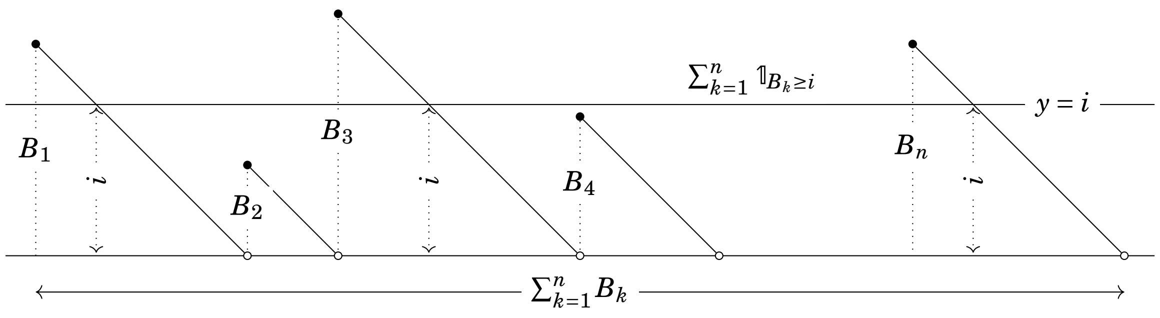


Figure 2.17: The remaining job size as a function of time. A batch crosses the line $y = i$ iff it contains an $i$th item. Thus, the number of $i$th items in $n$ batches is $\sum_{k=1}^n \mathbb{1}_{B_k \geq i}$. The total number of items arrived is $\sum_k^n B_k$. The fraction of time an $i$th item is in service is therefore $\sum_k^n \mathbb{1}_{B_k \geq i} / \sum_k^n B_k$.

By PASTA this is also the fraction of jobs that see $i$ items in service. Finally, assuming that the limits exist,

$$\mathbb{P}\{R = i\} = \lim_{n\to\infty} \frac{\sum_{k=1}^n \mathbb{1}_{B_k \geq i}}{\sum_{k=1}^n B_k} = \lim_{n\to\infty} \frac{n^{-1}\sum_{k=1}^n \mathbb{1}_{B_k \geq i}}{n^{-1}\sum_{k=1}^n B_k} = \frac{G(i-1)}{\mathbb{E}[B]}.$$

With this expression for $\mathbb{P}\{R = i\}$, the expected remaining time becomes

$$\begin{aligned}\mathbb{E}[R] &= \sum_{i=1}^{\infty} i\,\mathbb{P}\{R = i\} = \sum_{i=1}^{\infty} i \frac{G(i-1)}{\mathbb{E}[B]} \\ &= \sum_{i=0}^{\infty} (i+1)\frac{G(i)}{\mathbb{E}[B]} = \sum_{i=0}^{\infty} i \frac{G(i)}{\mathbb{E}[B]} + \sum_{i=0}^{\infty} \frac{G(i)}{\mathbb{E}[B]} \\ &= \frac{\mathbb{E}\left[B^2\right]}{2\mathbb{E}[B]} - \frac{\mathbb{E}[B]}{2\mathbb{E}[B]} + \frac{\mathbb{E}[B]}{\mathbb{E}[B]} = \frac{\mathbb{E}\left[B^2\right]}{2\mathbb{E}[B]} + \frac{1}{2},\end{aligned}$$

where we use the results of the exercises below.

Finally, recalling that in the above we conditioned on the server being busy, we get that $\mathbb{E}[B_r] = \rho\,\mathbb{E}[R]$. Combining this with the above expression we obtain (2.57).

**Remark 2.15.1.** The above derivation for $\mathbb{P}\{R = i\}$ is important for its own sake. $\mathbb{P}\{R = i\}$ is known as the *remaining lifetime distribution* or the *residual life* and used in, for instance, maintenance planning. The derivation is worth remembering, in particular the sample path derivation.

## 2.16 M/G/1 Queue: Expected Waiting Time

Let's focus on one queue in a supermarket, served by one cashier, and assume that customers do not jockey, i.e., change queue. What can we say about the average waiting time in queue if service times are not exponential, like in the $M/M/1$ queue, but have a more general distribution? One of the celebrated results of queueing theory is the Pollackzek-Khintchine formula by which we can compute the average waiting time formula for the $M/G/1$ queue. In this section we derive this result by means of sample path arguments.

Recall that Eq. (2.54) states that

$$\mathbb{E}\left[W_Q\right] = \frac{\mathbb{E}[S_r]}{1-\rho}$$

This equation is valid for the $M/G/1$ queue, as in the derivation we used the PASTA property but did not make any assumption about the service time distribution, except that it has a finite mean. To use it, we need to compute the average remaining service time for generally distributed service times. This is what we will do here.

Consider the $k$the job of some sample path the $M/G/1$ queueing process. This job requires $S_k$ units of service, let its service time start at time $A_k$ so that it departs the server at time $D_k = A_k + S_k$, c.f. Figure 2.18. Define

$$R_k(s) = (D_k - s)\mathbb{1}_{A_k \leq s < D_k}$$

as the remaining service time at time $s$ of job $k$. So see this, observe that when the time $s \in [A_k, D_k)$, the remaining service time until job $k$ departs is $D_k - s$, while if $s \not\in [A_k, D_k)$, job $k$ is not in service so it cannot have any remaining service as well.

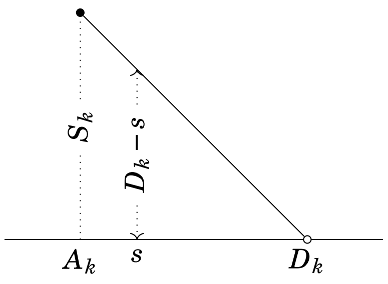


Figure 2.18: Remaining service time.

For the average remaining service time, we can add all remaining service times up to some time $t$, and then divide by $t$, specifically,

$$R(t) = \frac{1}{t}\int_0^t \sum_{k=1}^{\infty} R_k(s)\,\mathrm{d}s,$$

so that we can define

$$\mathbb{E}[S_r] = \lim_{t\to\infty} R_t,$$

provided this limit exists along the sample path.

Now observe that as $t$ can be somewhere in a service interval, it is easiest to consider $R(t)$ at departure times $\{D_n\}$, as at a time $D_n$ precisely $n$ jobs departed. Thus,

$$R(D_n) = \frac{1}{D_n}\int_0^{D_n} \sum_{k=1}^{\infty} R_k(s)\,\mathrm{d}s = \frac{1}{D_n}\sum_{k=1}^{n}\int_0^{D_n} R_k(s)\,\mathrm{d}s,$$

the interchange being allowed by the positivity of $R_k(s)$. But, if $k < n$ so that $D_k < D_n$,

$$\begin{aligned}\int_0^{D_n} R_k(s)\,\mathrm{d}s &= \int_0^{D_n} (D_k - s)\mathbb{1}_{A_k \le s < D_k}\,\mathrm{d}s = \int_{A_k}^{D_k} (D_k - s)\,\mathrm{d}s \\ &= \int_{A_k}^{A_k+S_k} (A_k + S_k - s)\,\mathrm{d}s = \int_0^{S_k} (S_k - s)\,\mathrm{d}s \\ &= \int_0^{S_k} s\,\mathrm{d}s = \frac{S_k^2}{2}.\end{aligned}$$

With this,

$$R(D_n) = \frac{1}{D_n}\sum_{k=1}^{n}\frac{S_k^2}{2} = \frac{n}{D_n}\frac{1}{n}\sum_{k=1}^{n}\frac{S_k^2}{2}.$$

Finally, assuming that along this sample path, $D_n \to \infty$ as $t \to \infty$, $n/D_n \to \gamma$, $\gamma = \lambda$ by rate stability, and the limit $n^{-1}\sum_{k=1}^{n} S_k^2$ exists as $n \to \infty$, we get

$$\mathbb{E}[S_r] = \lim_{D_n\to\infty} R(D_n) = \frac{\lambda}{2}\mathbb{E}\left[S^2\right]. \tag{2.57}$$

In the exercises we ask you to prove that

$$\mathbb{E}[S_r] = \frac{1+C_s^2}{2}\rho\,\mathbb{E}[S], \tag{2.58}$$

where

$$C_s^2 = \frac{\mathbb{V}[S]}{(\mathbb{E}[S])^2},$$

With the above we have obtained the *Pollackzek-Khintchine* (PK) formula for the average waiting time in queue:

$$\mathbb{E}\left[W_Q\right] = \frac{1}{1-\rho}\mathbb{E}[S_r] = \frac{1+C_s^2}{2}\frac{\rho}{1-\rho}\mathbb{E}[S]. \tag{2.59}$$

This proves to be a very useful result; a few of the problems below will clarify how to use it.

## 2.17 M$^X$/M/1 Queue Length Distribution

In this section we extend the analysis of the $M^X/M/1$ batch queue that we started in Section 2.15. We present a numerical, recursive, scheme to compute the stationary distribution of the number of items in queue.

To find the distribution $p(n)$, which is equal to $\pi(n)$ by PASTA, we turn to level-crossing arguments. However, the arguments that lead to the level-crossing equation (2.32) need to generalized. To see this, we consider an example. If $L(t) = 3$, the system contains 3 items (but not necessarily 3 jobs.) 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 $B_k = 20$, then $L(A_k) = 3 + 20 = 23$, that is, right after the arrival of job $k$ the system contains 23 items. In this case, at the arrival of job $k$, all levels between states 3 and 23 are crossed. The left panel in Figure 2.19 demonstrates this in more general terms. Level $n$ can be up-crossed from below from many states, in fact from any level $m < n$. However, it can only be down-crossed from state $n + 1$.

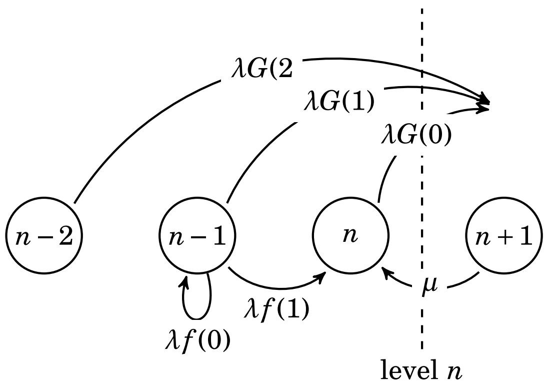


Figure 2.19: Level crossing of level $n$. Observe that when the system is in state $n-2$, the arrival of any batch larger than 2 ensures that level $n$ is crossed from below. The rate at which such events happen is $\lambda \mathbb{P}\{X > 2\} = \lambda G(2)$. Similarly, in state $n-1$ the arrival of any batch larger than one item ensures that level $n$ is crossed, and this occurs with rate $\lambda G(1)$, and so on.

As always with level-crossing arguments, we turn to counting how often level $n$ is upcrossed and downcrossed as a function of time. The downcrossing rate is easy; it is exactly the same as for the $M/M/1$ queue. Thus, from Eq. (2.33b), the downcrossing rate is

$$\frac{D(n,t)}{t} = \frac{D(n,t)}{Y(n+1,t)} \frac{Y(n+1,t)}{t} \to \mu(n+1)p(n+1) = \mu\pi(n+1), \tag{2.60}$$

where we use that $\mu(n+1) = \mu$ by the memoryless property of the service times and PASTA to see that $\pi(n+1) = p(n+1)$. Observe that this part was easy because there is just one way (i.e., there is just one arrow from right to left in Figure 2.19) to downcross level $n$, namely from $n+1$ to $n$. Counting upcrossings requires quite some more work.

Let us write $\mathbb{1}_{L(A_k-)\leq n} \mathbb{1}_{L(A_k)>n}$ to track whether the $k$th job sees $n$ or less items in the system upon arrival, i.e. condition $\mathbb{1}_{L(A_k-)\leq n}$, and is such that after its arrival the system contains more than $n$ items, i.e., condition $\mathbb{1}_{L(A_k)>n}$. Thus, if $\mathbb{1}_{L(A_k-)\leq n} \mathbb{1}_{L(A_k)>n} = 1$, job $k$ crossed level $n$ from below. Hence

$$\sum_{k=1}^{\infty} \mathbb{1}_{A_k\leq t} \mathbb{1}_{L(A_k-)\leq n} \mathbb{1}_{L(A_k)>n},$$

counts all arrivals up to time $t$ that enforce a crossing of level $n$ from below.

From Figure 2.19 we see that an upcrossing can be decomposed into:

$$\begin{aligned}\mathbb{1}_{L(A_k-)\leq n} \mathbb{1}_{L(A_k)>n} &= \mathbb{1}_{L(A_k-)=n} \mathbb{1}_{B_k>0} + \mathbb{1}_{L(A_k-)=n-1} \mathbb{1}_{B_k>1} + \cdots + \mathbb{1}_{L(A_k-)=0} \mathbb{1}_{B_k>n} \\ &= \sum_{m=0}^{n} \mathbb{1}_{L(A_k-)=m} \mathbb{1}_{B_k>n-m}.\end{aligned}$$

In other words, paths that upcross level $n$ require that any job that sees $m$ ($m \leq n$) in the system upon arrival must bring a batch larger than $n-m$ items.

Define

$$A(m,n,t) = \sum_{k=1}^{\infty} \mathbb{1}_{A_k\leq t} \mathbb{1}_{L(A_k-)=m} \mathbb{1}_{B_k>n-m}.$$

as the number of jobs that, up to time $t$, see $m$ in the system upon arrival and have batch size larger than $n-m$. Then the number of times level $n$ is upcrossed up to time $t$ can be written as $\sum_{m=0}^{n} A(m,n,t)$, hence the upcrossing rate up to time $t$ is

$$\frac{1}{t}\sum_{m=0}^{n} A(m,n,t).$$

Simplifying this comes down to finding the right quantities by which to divide. Some experimentation will reveal that the following gives the most relevant insights:

$$\frac{A(m,n,t)}{t} = \frac{A(t)}{t}\frac{A(m,t)}{A(t)}\frac{A(m,n,t)}{A(m,t)}. \tag{2.61}$$

Here $A(t)/t$ and $A(m,t)/A(t)$ have the same meaning as in Section 2.12, hence the first of these fractions converges to $\lambda$ and the second to $\pi(m)$ as $t\to\infty$. Now, provided the limit exists, we define

$$\lim_{t\to\infty}\frac{A(m,n,t)}{A(m,t)} = \lim_{t\to\infty}\frac{\sum_{k=1}^{\infty}\mathbb{1}_{A_k\le t}\mathbb{1}_{L(A_k-)=m,B_k>n-m}}{\sum_{k=1}^{\infty}\mathbb{1}_{A_k\le t}\mathbb{1}_{L(A_k-)=m}} = \mathbb{P}\{B>n-m \mid L(A-)=m\},$$

where $\mathbb{P}\{L(A-)=m\}$ has the interpretation of the steady-state probability that a job sees $m$ jobs in the system upon arrival. By *assumption*, $B$ and $L(A-)$ are independent[8], and therefore,

$$\lim_{t\to\infty}\frac{A(m,n,t)}{A(n,t)} = \mathbb{P}\{B>n-m\} = G(n-m).$$

By combining the above and making the usual assumptions about the existence of all limits involved we find

$$\lim_{t\to\infty}\frac{A(m,n,t)}{t} = \lambda\pi(m)G(n-m).$$

Thus, Eq. (2.63) has the interpretation that the rate at which level $n$ is crossed from below from state $m$ is equal 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 $n-m$.

The last step is to relate the up- and downcrossing rates. Recall that for the $M/M/1$ the number of upcrossings and downcrossings of level $n$ could not differ more than one, i.e., $|A(n,t)-D(n,t)|\le 1$. Similarly, for the $M^X/M/1$ queue we have

$$\left|\sum_{m=0}^{n} A(m,n,t) - D(n,t)\right| \le 1. \tag{2.62}$$

Thus, taking the limit $t\to\infty$ in this equation, we conclude that the upcrossing rate and the downcrossing rate of level $n$ must be the same. Hence,

$$\lambda\sum_{m=0}^{n} G(n-m)\pi(m) = \mu\pi(n+1). \tag{2.63}$$

[8]Thus,

$$\mathbb{P}\{B>n-m \mid L(A-)=m\} = \frac{\mathbb{P}\{B>n-m, L(A-)=m\}}{\mathbb{P}\{L(A-)=m\}} = \frac{\mathbb{P}\{B>n-m\}\,\mathbb{P}\{L(A-)=m\}}{\mathbb{P}\{L(A-)=m\}} = \mathbb{P}\{B>n-m\}.$$

For the excess probabilities $\mathbb{P}\{L > n\}$ we need to compute the actual probabilities $\pi(n)$ from the recursion (2.65). To apply this, temporarily set $\pi(0) = 1$. Then, for $n = 1$, we have $\mu\pi(1) = \lambda\pi(0) = \lambda$, . Then, since $G(0) = 1$, and writing $\alpha = \lambda/\mu$, the *unnormalized* state probabilities are

$$\begin{aligned}
\pi(1) &= \alpha \\
\pi(2) &= \alpha G(0)\pi(1) + \alpha G(1)\pi(0) = \alpha(\alpha + G(1)) = \alpha^2 + \alpha G(1), \\
\pi(3) &= \alpha[G(0)\pi(2) + G(1)\pi(1) + G(2)\pi(0)] = \alpha[\alpha^2 + 2\alpha G(1)) + G(2)],
\end{aligned}$$

and so on. (Evidently, the computer is better suited to continue with this.)

It is left to find the normalization constant. As this recursion does not lead to a closed form expression for $p(n)$, such as Eq. (2.39), we need to use a criterion to stop this iterative procedure. Finding general conditions on when to stop is not directly easy, but a pragmatic approach is to simply stop at some (large) number $N$ and take $G = \sum_{i=0}^{N} \pi(i)$ as the normalization constant. Then $\pi(0) = 1/G$, $\pi(1) = \alpha/G$, and so on.

**Remark 2.17.1.** An interesting extension is to consider a queueing process with batch services, i.e., the $M/M^Y/1$ queue. Construction a recursion for the steady-state probabilities $p(n)$ for this case is not hard, in fact, mostly analogous to Eq. (2.65). However, solving the recursion is much less simple, hence we will not discuss this further.

## 2.18 M/G/1 Queue Length Distribution

In Section 2.16 we derived the Pollackzek-Khintchine Eq. (2.61) by which we can compute the average waiting time in queue for the $M/G/1$ queue. Then, with Little's law, we therefore also have the average queue length. However, if we need the loss probability $\mathbb{P}\{L > n\}$, we need expressions for the stationary distribution of the number of jobs in the system $p(n) = \mathbb{P}\{L = n\}$. Here we present a set of recursions by which we can compute $p(n)$ for a given arrival rate $\lambda$ and service distribution $F$ of the service times.

To derive the stationary distribution $p(n)$ we work in steps, and use similar level-crossing arguments as in Section 2.17. As we will see, the argumentation is quite subtle, as it uses an interplay of the PASTA property and the relation of Eq. (2.50) between $\pi(n)$, $p(n)$ and $\delta(n)$. The important idea is to focus on the *departure* moments, as these moments can be considered as restarts of the service times, and on the number of arrivals $Y_k$ during the service time of the $k$th job. Assume that $f(j) = \mathbb{P}\{Y_k = j\}$ is given and write $G(j) = \mathbb{P}\{Y_k > j\}$.

First we concentrate on a downcrossing of level $n$, see Figure 2.20. Suppose that the '$k-1$'th job leaves $n+1$ jobs behind after its service completion. Then, if during the service time of the $k$th job, no other jobs arrive, it must be that job $k$ leaves $n$ jobs behind. Thus, job $k$ realizes a downcrossing of level $n$ only when $L(D_{k-1}) = n+1$ and $Y_k = 0$, i.e.,

$$L(D_{k-1}) = n+1, Y_k = 0 \implies L(D_k) = n$$

Let us next count all downcrossings up to time $t$. Clearly,

$$D(n+1, 0, t) = \sum_{k=1}^{\infty} \mathbb{1}_{D_k \leq t}\, \mathbb{1}_{L(D_{k-1})=n+1}\, \mathbb{1}_{Y_k=0}$$

is the number of jobs that depart before time $t$, i.e., $\mathbb{1}_{D_k \leq t}$, and whose predecessors leave $n+1$ jobs behind, i.e., $\mathbb{1}_{L(D_{k-1})=n+1}$, and such that no jobs arrive during the service time, i.e., $\mathbb{1}_{Y_k=0}$. .

Then, by using the definitions and limits developed in Sections 2.7 and 2.12,

$$\begin{aligned}\lim_{t\to\infty}\frac{D(n+1,0,t)}{t} &= \lim_{t\to\infty}\frac{D(t)}{t}\frac{D(n+1,t)}{D(t)}\frac{D(n+1,0,t)}{D(n+1,t)}\\ &= \gamma\delta(n+1)\lim_{t\to\infty}\frac{D(n+1,0,t)}{D(n+1,t)}\\ &= \gamma\delta(n+1)\mathbb{P}\{Y=0\}\\ &= \gamma\delta(n+1)f(0),\end{aligned}$$

where the last limit follows from the independence of $Y_k$ and $L(D_{k-1})$.

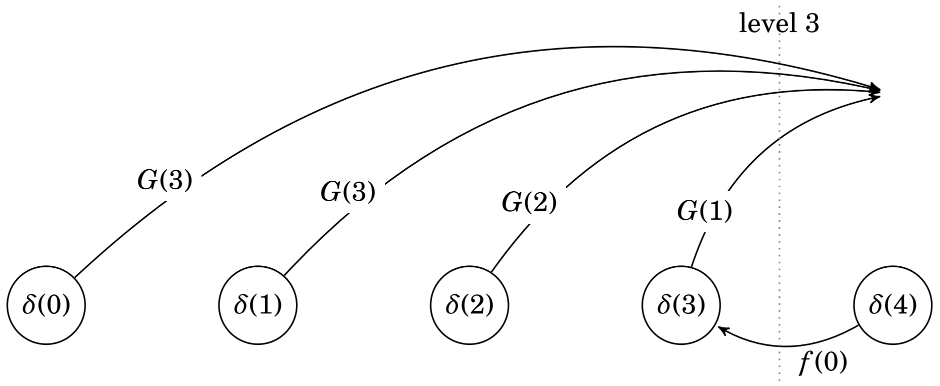


Figure 2.20: Level 3 is crossed from below with rates $\delta(0)G(3)+\delta(1)G(3)+\cdots\delta(3)G(1)$ and crossed from above with rate $\delta(4)f(0)$.

For the upcrossings, suppose first that $L(D_{k-1}) = n > 0$. If there are no arrivals during the service of job $k$, i.e., $Y_k = 0$, then job $k$ must leave $n-1$ jobs behind. Thus, no upcrossing occurs in this case. Next, if $Y_k = 1$, then job $k$ leaves $n$ jobs behind. Thus, still no upcrossing occurs. In fact, to upcross level $n$ when $L(D_{k-1}) = n > 0$, at least two new jobs must arrive during the service of job $k$. More generally, if $0 < m \leq n$,

$$L(D_{k-1}) = m, Y_k > n-m+1 \implies L(D_k) > n,$$

while if $m = 0$ (think about this),

$$L(D_{k-1}) = 0, Y_k > n \implies L(D_k) > n.$$

Similar to our earlier work, by defining proper counting functions, c.f., Exercise 1, and taking suitable limits, we find for the upcrossing rate that

$$\gamma\delta(0)G(n) + \gamma\sum_{m=1}^{n}\delta(m)G(n-m+1). \tag{2.64}$$

Equating the down-crossings and up-crossing and dividing by $\gamma$ gives

$$f(0)\delta(n+1) = \delta(0)G(n) + \sum_{m=1}^{n}\delta(m)G(n+1-m). \tag{2.65}$$

Finally, noting that $\pi(n) = \delta(n)$ from (2.50) and the fact that the $M/G/1$ queue length process has one-step transitions, we arrive at

$$f(0)\pi(n+1) = \pi(0)G(n) + \sum_{m=1}^{n} \pi(m)G(n+1-m).$$

Again, we obtain a recursion by which we can compute, iteratively, the state probabilities. Setting $\pi(0) = 1$ at first, $\pi(1)$ follows from the above recursion. Then, with $\pi(0) = 1$ and $\pi(1)$ we find $\pi(2)$, and so on, up to some limit $N$. Then, set $G = \sum_{n=0}^{N} \pi(n)$ as the normalization constant. As a practical approach to determine whether $N$ is sufficiently large, compute $p(n)$ also for a couple for values above $N$. If these values decrease when $n$ becomes larger and they are very small compared to the $p(n)$ with $n < N$, $N$ is most surely sufficiently large. However, getting formal bounds on a proper size of $N$ requires more work than we can do here.

## 2.19 A Relation Between the $\mathbf{M^X/M/1}$ and $\mathbf{M/G/1}$ Queue

There is an interesting relation between the $M^X/M/1$ and the $M/G/1$ queueing systems. As an example, consider the $M^{10}/M/1$ queue in which each job corresponds to the arrival of two items of work. Intuitevely, it is reasonable that this queueing process is quite similar to an $M/G/1$ queue in which each job requires approximately 10 time units. Below we will make this relation precise. We will also use this to find another derivation of the Pollackzek-Khintchine equation, if we consider a certain limiting case.

The basic observation is that by discretizing we can approximate the service time $S$ of the $M/G/1$ queue by the service time of a batch of small units of work. To make this more specific, assume that the distibution of the service time $S$ is given. Use a grid of size $1/n$ and take

$$\mathbb{P}\left\{X^{(n)} = i\right\} = \mathbb{P}\{S \in (i/n, (i+1)/n]\} \tag{2.66}$$

as the distribution of the batch $X^{(n)}$. Let the service time of one unit of the batch be exponentially distributed with mean $1/n$. Thus, if $n$ becomes larger, the service time $S$ is represented by ever larger batches, but such that the service time of each unit becomes ever smaller. These arguments can be made precise in a probabilistic sense, but here we rely on our intuition to conclude that, in some sense, $X^{(n)}/n \xrightarrow{d} S$.

Let us see how this limit can be used to show that the expected waiting in queue $\mathbb{E}[W_Q]$ of the batch queue leads to the Pollackzek-Khintchine equation. Writing $\mathbb{E}[L^{(n)}]$ for the batch queue with grid size $1/n$, we use (2.58) to see that

$$\frac{\mathbb{E}[L^{(n)}]}{n} = \frac{\rho}{1-\rho}\cdot\frac{1}{2}\left(1+\frac{\mathbb{V}[X^{(n)}]}{(\mathbb{E}[X^{(n)}])^2}\right)\frac{\mathbb{E}[X^{(n)}]}{n} + \frac{1}{2n}\frac{\rho}{1-\rho}. \tag{2.67}$$

By assumption[9]:

$$\frac{X^{(n)}}{n} \to S$$

[9] Here we interchange limits and expectations. A formal approach should pay due attention to the validity of all these interchanges.

so that must be that

$$\mathbb{E}\left[\frac{X^{(n)}}{n}\right] \to \mathbb{E}[S]$$

and

$$\mathbb{E}\left[\frac{(X^{(n)})^2}{n^2}\right] \to \mathbb{E}\left[S^2\right],$$

as $n \to \infty$. Hence,

$$\begin{aligned}\frac{\mathbb{V}\left[X^{(n)}\right]}{n^2} &= \frac{\mathbb{E}\left[(X^{(n)})^2\right] - (\mathbb{E}\left[X^{(n)}\right])^2}{n^2} = \mathbb{E}\left[\frac{(X^{(n)})^2}{n^2}\right] - \left(\mathbb{E}\left[\frac{X^{(n)}}{n}\right]\right)^2 \\ &\to \mathbb{E}\left[S^2\right] - (\mathbb{E}[S])^2 = \mathbb{V}[S].\end{aligned}$$

from which

$$\frac{\mathbb{V}\left[X^{(n)}\right]}{(\mathbb{E}\left[X^{(n)}\right])^2} = \frac{\mathbb{V}\left[X^{(n)}\right]}{n^2}\frac{n^2}{(\mathbb{E}\left[X^{(n)}\right])^2} \to \frac{\mathbb{V}[S]}{(\mathbb{E}[S])^2} = C_s^2,$$

as $n \to \infty$. Moreover, we expect that

$$\frac{\mathbb{E}\left[L^{(n)}\right]}{n} \to \mathbb{E}\left[W_Q\right],$$

since the number of items in the system, $\mathbb{E}\left[L^{(n)}\right]$, becomes ever larger as $n$ increases, but the service time of one item is $1/n$, which becomes ever smaller. Thus, taking the limit $n \to \infty$ in (2.74) we obtain

$$\frac{\mathbb{E}\left[L^{(n)}\right]}{n} \to \frac{\rho}{1-\rho}\frac{1+C_s^2}{2}\mathbb{E}[S].$$

This is the same as Eq. (2.61)!

## 2.20 G/G/1 and G/G/c Queue: Approximations

In manufacturing settings it is quite often the case that the arrival process at a station is not Poisson. 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. 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 consequence the Pollackzek-Khintchine formula for the $M/G/1$-queue can no longer be reliably used to compute the average waiting times. 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 queue. Thus, the $M/G/1$ waiting time formula cannot, and should not, be applied to approximate the average waiting time of the $G/G/1$ queue.

There is no formula as yet by which the average waiting times for the $G/G/1$ queue can be computed; only approximations are available. One such simple and robust approximation is based on the following observation. Recall the waiting time in queue for the $M/M/1$ queue:

$$\mathbb{E}\left[W_Q(M/M/1)\right] = \frac{\rho}{1-\rho}\mathbb{E}[S] = \frac{1_a + 1_b}{2}\frac{\rho}{1-\rho}\mathbb{E}[S],$$

where we label the number 1 with $a$ and $b$. When generalizing this result to the $M/G/1$ queue we get

$$\mathbb{E}\left[W_Q(M/G/1)\right] = \frac{1_a + C_s^2}{2}\frac{\rho}{1-\rho}\mathbb{E}[S],$$

Thus, $1_b$ in the expression for the $M/M/1$ queue is replaced by $C_s^2$ in the expression for the $M/G/1$ queue. As a second generalization, Kingman proposed to replace $1_a$ in this formula by the SCV of the inter-arrival times

$$C_a^2 = \frac{\mathbb{V}[X]}{(\mathbb{E}[X])^2}$$

resulting in

$$\mathbb{E}\left[W_Q(G/G/1)\right] \approx \frac{C_a^2 + C_s^2}{2}\frac{\rho}{1-\rho}\mathbb{E}[S].$$

This formula is reasonable accurate; for more, related, expressions we refer to **?** and **?**. With Little's law we can compute $\mathbb{E}\left[L_Q\right]$ from the above. Moreover, $\mathbb{E}[W] = \mathbb{E}\left[W_Q\right] + \mathbb{E}[S]$, and so on.

It is crucial to memorize the *scaling* relations that can be seen from the $G/G/1$ waiting time formula. Roughly:

1. $\mathbb{E}\left[W_Q\right] \sim (1-\rho)^{-1}$. The consequence is that the waiting time increases *very steeply* when $\rho$ is large, hence is very sensitive to the actual value of $\rho$ when $\rho$ is large.
2. $\mathbb{E}\left[W_Q\right] \sim C_a^2$ and $\mathbb{E}\left[W_Q\right] \sim C_s^2$. Hence, reductions of the variation of the inter-arrival and service times do affect the waiting time, but only linearly.
3. $\mathbb{E}\left[W_Q\right] \sim \mathbb{E}[S]$. Thus, working in smaller job sizes reduces the waiting time also. 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 to do '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.

For the $G/G/c$ queue we also only have an approximation:

$$\mathbb{E}\left[W_Q\right] = \frac{C_a^2 + C_e^2}{2}\frac{\rho^{\sqrt{2(c+1)}-1}}{c(1-\rho)}\mathbb{E}[S]. \tag{2.68}$$

Even though the above results are only approximate, 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.

Clearly, Kingman'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 information that is available. However, when for instance the number of arrivals per day have been logged for some time, i.e., we know

$\{a_n, n = 1,\ldots,N\}$ for some $N$, we can this information instead of the the inter-arrival times $\{X_k\}$, to obtain insight into the relevant SCVs. Here we present a relation to compute $C_a^2$ from $\{a_n\}$; of course it can be applied to $C_s^2$ as well.

**Theorem 2.20.1.** The SCV of the inter-arrival times can be estimated with the following formula

$$C_a^2 \approx \frac{\tilde{\sigma}^2}{\tilde{\lambda}},$$

where

$$\tilde{\lambda} = \lim_{n\to\infty} n^{-1}\sum_{i=1}^n a_i, \quad \tilde{\sigma}^2 = \lim_{n\to\infty} n^{-1}\sum_{i=1}^n a_i^2 - \tilde{\lambda}^2,$$

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 **?**. We use quite a bit of the notation developed in Section 2.7. 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 $\sigma$. (Observe that $\sigma$ is not the same as $\tilde{\sigma}$ above.) Note that $C_a^2$ is defined in terms of $\lambda$ and $\sigma$ as:

$$C_a^2 = \frac{\mathbb{V}[X_i]}{(\mathbb{E}[X_i])^2} = \frac{\sigma^2}{1/\lambda^2} = \lambda^2\sigma^2.$$

Next, let $A_k$ be the arrival time of the $k$th arrival. The following useful relation between $A(t)$ and $A_k$ enables us to prove our result (recall that we uses a similar relation in the derivation of the Poisson process):

$$\mathbb{P}\{A(t) < k\} = \mathbb{P}\{A_k > t\}$$

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 $\alpha$ that is yet to be determined. Thus, $\mathbb{E}[A(t)] = \lambda t$ and $\mathbb{V}[A(t)] = \alpha^2 t$.

Using that $\mathbb{P}\{N(0,1) \leq y\} = \mathbb{P}\{N(0,1) > -y\}$ (and $\mathbb{P}\{N(0,1) = y\} = 0$) we have that

$$\begin{aligned}\Phi(y) &\approx \mathbb{P}\left\{\frac{A_k - k/\lambda}{\sigma\sqrt{k}} \leq y\right\} \\ &= \mathbb{P}\left\{\frac{A_k - k/\lambda}{\sigma\sqrt{k}} > -y\right\} \\ &= \mathbb{P}\left\{A_k > \frac{k}{\lambda} - y\sigma\sqrt{k}\right\}.\end{aligned}$$

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 $\mathbb{P}\{A_k > t_k\} = \mathbb{P}\{A(t_k) < k\}$. With this we get,

$$\begin{aligned}\Phi(y) &\approx \mathbb{P}\{A_k > t_k\} \\ &= \mathbb{P}\{A(t_k) < k\} \\ &= \mathbb{P}\left\{\frac{A(t_k) - \lambda t_k}{\alpha\sqrt{t_k}} < \frac{k - \lambda t_k}{\alpha\sqrt{t_k}}\right\}.\end{aligned}$$

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 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 $\alpha$ 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),

$$\begin{aligned}\frac{\sigma_k^2}{\lambda_k} &= \frac{\mathbb{V}[A(t)]}{\mathbb{E}[A(t)]} \approx \frac{\alpha^2 t}{\lambda t} = \frac{\alpha^2}{\lambda} = \frac{\lambda^3\sigma^2}{\lambda} = \lambda^2\sigma^2 \\ &= C_a^2,\end{aligned}$$

where the last equation follows from the above definition of $C_a^2$. The proof is complete. □

## 2.21 Service Interruptions

No notes; only questions.

## 2.22 Process Batching

No notes; only exercises.

**Exercise 2.22.1.** Zijm.Ex.1.12.1

**Exercise 2.22.2.** Zijm.Ex.1.12.2

**Exercise 2.22.3.** Zijm.Ex.1.12.3

# 3 Open Queueing Networks

Read the appropriate sections of Zijm's book.

## 3.1 Deterministic Queueing Networks

The simplest possible single-server queueing system is such that job interarrival times are deterministic and that all service times are equal. In such cases it is easy to determine the maximal throughput, i.e., the capacity of the queueing system. Clearly, if the service time is $t_1$, then the service rate is $r_1 = 1/t_1$. We can also determine the minimal amount of jobs required to keep the server busy, hence achieve maximal throughput. We call this the *critical work in progress (WIP)* and denote it by $W_0$. The minimal amount of time needed to move from the start of the system to the end, also called *raw processing time* is denoted by $T_0$. With Little's law we can relate these two concepts for the single-server example. If the arrival rate is equal to the service rate, then $\lambda = r_1$, so that

$$W_0 = \lambda T_0 = r_1 T_0.$$

Since, in the single server case $T_0 = t_1$, we see that $W_0 = r_1 t_1 = t_1^{-1} t_1 = 1$. This result is evident: to get maximal throughput, it is necessary that the server is always busy, hence it must be that $W_0 \geq 1$. Given that we deal with a $D/D/1$, by assumption, we can tune the arrivals such that whenever a job leaves, a new job arrives.

When we deal with *networks*, determining $W_0$ and $T_0$ is not so simple anymore. These quantities, however, are important to know. It is not possible to organize the network such that jobs spend less time than $T_0$ in the network. The raw processing time is the sum of the required processing times and does not include, by assumption, any queueing times. The critical WIP $W_0$ is important, because if the network contains less jobs than this amount, throughput will suffer. Stated in other terms, there not sufficient jobs in the network to keep bottleneck machines fully loaded with work. For these reasons $T_0$ and $W_0$ act as lower bounds on the time jobs spend in the network and the amount of work required to achieve a certain throughput. Finally, it is important to know the bottleneck capacity $r_b$ of the network: The maximal throughput of the network cannot exceed the bottleneck capacity. Thus, when accepting jobs, one needs to respect the capacity of the network.

Before we study queueing networks in stochastic settings, we should familiarize ourselves with the behavior of networks of queueing systems in the simplest setting possible. For this we need some further definitions. A queueing network can be represented as *graph*. *Stations* form the nodes in the graph, and edges between nodes represents the possibility that jobs that can move from one station to another. A simple example is a tandem network with two stations. Jobs arrive at station 1 and after service they move to station 2. After service at station 2, they leave the network. A station can contain one or more servers. For instance, a single-server station contains just one server. Finally, we assume that job service times are constant at all servers at all stations. In other words, service times need to be same from one machine/server or station to another, but each server works at constant speed. The *utilization* of station $i$ is the rate $\lambda_i t_i$, i.e., the rate at which work arrives times the (average) service time per job. The *bottleneck* station is the station with the *highest utilization*.

The problems below are meant to help you become acquainted with networking behavior. They appear much simpler than they are; I had much less trouble making them than solving them. In fact, to ensure that I got the answer right, I often tried to find two different ways to solve the same problem. Be warned!

## 3.2 Open Single-Class Product-Form Networks

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 $\lambda_i/\gamma$ seem 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 $\gamma$. Thus, by applying Little's law to the 'box', we have that

$$\mathbb{E}[L] = \gamma \mathbb{E}[W].$$

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:

$$\mathbb{E}[L] = \sum_{i=1}^M \mathbb{E}[L_i].$$

Applying Little's law to each station separately we get that $\mathbb{E}[L_i] = \lambda_i \mathbb{E}[W_i]$. Filling this into the above,

$$\mathbb{E}[W] = \frac{\mathbb{E}[L]}{\gamma} = \sum_{i=1}^M \frac{\mathbb{E}[L_i]}{\gamma} = \sum_{i=1}^M \frac{\lambda_i \mathbb{E}[W_i]}{\gamma},$$

where we recognize the visit ratios.

## 3.3 Tandem queues

No notes; only exercises.

## 3.4 General Job Shop Manufacturing Systems

Perhaps the term 'class' needs some clarification. A network with a single class of jobs means that the all jobs have the same service distribution. In multi-class networks jobs of different classes typically have different service distributions. Note that we do not discuss multi-class queueing networks in this course.

# 4 Closed Queueing Networks

## 4.1 Gordon-Newell Networks

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.

It is perhaps the easiest to start with studying the MVA algorithm, then the MDA algorithm and finish with the convolution algorithm.

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 $VP = V$.

## 4.2 The Convolution Algorithm

In view of the later algorithm, it is more consistent to write $p_i(k|N) = \pi(n_i = k)$ for the marginal probability that station $i$ contains $k$ jobs, under the assumption that the network contains $N$ jobs.

Assume we have computed $p_i(k|N)$ with Eq.3.15, i.e., with the recursion

$$p_i(k|N) = f_i(k)\frac{G(M\setminus\{i\}, N-k)}{G(M,N)}.$$

We can then compute the rest of the performance measures as

$$\begin{aligned}
\text{TH}_i &= \mu_i \sum_{k=1}^{N} p_i(k|N), \\
\mathbb{E}[W] &= \frac{N}{TH_0}, \\
\mathbb{E}[L]_i &= \sum_{k=1} k p_i(k|N), \\
\mathbb{E}[W]_i &= \mathbb{E}[L]_i / V_i, \text{ by Little's law.}
\end{aligned}$$

The rest of the equations in Section 3.1.2 are interesting, but not necessary for an implementation in computer code.

## 4.3 MVA Algorithm

## 4.4 MDA Algorithm

A relatively easy way to obtain insight into the procedure is by comparing this case to the single-server single station case. Recall that, for the latter case,

$$\mathbb{E}\left[W_Q\right] = \rho\,\mathbb{E}[S_r \mid S_r > 0] + \mathbb{E}\left[L_Q\right]\mathbb{E}[S].$$

The first term is the probability that the server is found busy by an arriving job times the expected remaining service time of the job in service; the second is the expected time to clear the queue.

For the closed queueing network, the probability to find $k$ at station $j$ is $p_j(k|n-1)$, where we use the arrival theorem that states when $n$ jobs are present in the network, a 'jumping' job sees the stationary distribution of the same network but with one job less.

To wait in queue at station $j$ it is necessary that $k \geq c_j$, for otherwise service can start right at the arrival moment. Thus, the probability that all servers are busy is

$$\sum_{k=c_j}^{n-1} p_j(k|n-1).$$

(Compare the first $\rho$ in the above equation for $\mathbb{E}\left[W_Q\right]$). The service of the queue can start once the first job currently in service leaves. This takes $1/c\mu_j$. (Compare this to $\mathbb{E}[S_r \mid S_r > 0]$.)

The expected to clear the queue is $\mathbb{E}\left[L_{Q,j}\right]$ divided by the rate at which jobs are served. Thus, this time must be $\mathbb{E}\left[L_{Q,j}\right]/c\mu_j$. Now,

$$\mathbb{E}\left[L_{Q,j}\right] = \sum_{k=c_j}^{n-1} (k-c_j)p_j(k|n-1).$$

All in all, the total average time at station $j$ is the average time in queue plus the average service time of the job itself:

$$\mathbb{E}\left[W_j(n)\right] = \frac{1}{c_j\mu_j}\sum_{k=c_j}^{n-1} p_j(k|n-1) + \frac{1}{c_j\mu_j}\sum_{k=c_j}^{n-1}(k-c_j)p_j(k|n-1) + \frac{1}{\mu_j},$$

Up- and down-crossing of level $k$ gives that

$$\text{TH}_j(n)p(k-1|n-1) = \min(k,c_j)p(k|n). \tag{4.1}$$

# 5 Hints

**Hint 2.1.1** For the second child, condition on the event that the first does not chose the right number.

**Hint 2.1.3** Condition on the event $X > t$.

**Hint 2.1.9** 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 $A_i = \sum_{i=1}^k X_k$, 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 $X_i$.

**Hint 2.1.10** Use that $\int_0^\infty (\lambda t)^i e^{-\lambda t} \, \mathrm{d}t = \frac{i!}{\lambda}$. Another way would be to use that, once you have the moment generating function of some random variable $X$, $\mathbb{E}[X] = \frac{\mathrm{d}}{\mathrm{d}t} M(t)|_{t=0}$.

**Hint 2.1.11** Use that if $Z = \min\{X, Y\} > x$ that then it must be that $X > x$ and $Y > x$. Then use independence of $X$ and $Y$.

**Hint 2.1.12** Define the joint distribution of $A$ and $S$ and carry out the computations, or use conditioning, or use the result of the previous exercise.

**Hint 2.2.1** Recall that $\mathbb{E}[X+Y] = \mathbb{E}[X] + \mathbb{E}[Y]$. Just as a reminder, $\mathbb{E}[XY] \neq \mathbb{E}[X]\,\mathbb{E}[Y]$ in general. Only when $X$ and $Y$ are uncorrelated (which is implied by independence), the product of the expectations is the expection of the products.

**Hint 2.2.3** First find $p$, $n$, $\lambda$ and $t$ are such that the rate at which event occur in both processes are the same. Then consider the binomial distribution and use the standard limit $(1 - x/n)^n \to e^{-x}$ as $n \to \infty$.

**Hint 2.2.4** Realize that $\mathbb{P}\{N(t) = k\} = \mathbb{P}\{A_k \leq t\} - \mathbb{P}\{A_{k+1} \leq t\}$.

**Hint 2.2.5** Think about the meaning of the formula $\mathbb{P}\{N(h) = n \,|\, N(0) = n\}$. It is a conditional probability that should be read like this: given that up to time 0 we have seen $n$ arrivals (i.e, $N(0) = n$), what is the probability that just a little later ($h$) the number of arrivals is still $n$, i.e, $N(h) = n$? Then use the definition of the Poisson distribtuion to compute this probability. Finally, use Taylor's expension of $e^x$ to see that $e^x = 1 + x + o(x)$ for $|x| << 1$. Furthermore, use that $\sum_{i=2}^\infty x^i/i! = \sum_{i=0}^\infty x^i/i! - x - 1 = e^x - x - 1$.

**Hint 2.2.7** Use Bayes' law for conditional probability. Observe that

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

**Hint 2.2.8** Use a conditioning argument or use moment generating functions. In particular, for conditioning, use that $\mathbb{P}\{AB\} = \mathbb{P}\{A\,|\,B\}\,\mathbb{P}\{B\}$. More generally, if the set $A$ can be split into disjoint sets $B_i$, i.e, $A = \cup_{i=1}^n B_i$, then

$$\mathbb{P}\{A\} = \sum_{i=1}^n \mathbb{P}\{AB_i\} = \sum_{i=1}^n \mathbb{P}\{A\,|\,B_i\}\,\mathbb{P}\{B_i\},$$

where we use the conditioning formula to see that $\mathbb{P}\{AB_i\} = \mathbb{P}\{A\,|\,B_i\}\,\mathbb{P}\{B_i\}$. Now choose practical sets $B_i$.

**Hint 2.2.9** Suppose we write $N(t) = N_a(t)+N_s(t)$. Then $\mathbb{P}\{N_a(h) = 1, N_s(h) = 0 | N_a(h)+N_s(h) = 1\}$ is the probability that $N_a(h) = 1$ and $N_s(h) = 0$ *given* that $N(t) = 1$. In other words, the question is find out that, given one of the two processes 'fired', what is the probability that $N_a$ was the one that 'fired'.

**Hint 2.2.11** Condition on the total number of arrivals $N(t) = m$ up to time $t$. Realize that the probability that a job is of type 1 is Bernoulli distributed, hence when you consider $m$ jobs in total, the number of type 1 jobs is binomially distributed.

Again use that if the set $A$ can be split into disjoint sets $B_i$, i.e, $A = \cup_{i=1}^n B_i$, then

$$\mathbb{P}\{A\} = \sum_{i=1}^n \mathbb{P}\{A\,|\,B_i\}\,\mathbb{P}\{B_i\}.$$

Now choose practical sets $B_i$.

You might also 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 $\lambda p$.

**Hint 2.2.12** You might use generating functions here.

**Hint 2.4.15** Consider a numerical example. Suppose $Q_{k-1} = 20$. Suppose that the capacity is $c_k = 3$ for all $k$. Then a job that arrives in the $k$th period, must wait at least 20/3 (plus rounding) periods before it can leave the system. Now generalize this numerical example.

**Hint 2.4.18** Formuale the decision variables/controls, the objective and the constraints.

**Hint 2.5.2** Use that $W_{Q,1} = [W_{Q,0} + S_1 - X_1]^+$ and likewise for $W_{Q,2}$.

This first year probability. It should be elementary for you. Ensure you can do it.

**Hint 2.5.3** This appears to be trivial..., but I didn't find it easy... In fact, I found it a major headache, hence it is good to try yourself before looking at the answer. Check your probability book if you don't know what to do.

**Hint 2.5.5** Compare this to the definition in (2.9).

**Hint 2.5.7** Use Boolean algebra. Write, for notational ease, $A = \mathbb{1}_{A_k \leq t}$ 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 $k$.

**Hint 2.5.9** Make a plot of the function $A_{A(t)} - t$. What is the meaning of $V(A_{A(t)})$? What is $V(A_{A(t)} + A_{A(t)} - t$?

**Hint 2.6.1** When are the arrivals in slot $k$ available for service in either of the systems?

**Hint 2.6.2** Note first that from the expression for $Z_k$, $a_k - c_k = Z_k - Z_{k-1}$. Use this to get $Q_k = [Q_{k-1} + Z_k - Z_{k-1}]^+$. Subtract $Z_k$, use recursion and use subsequently,

$$\begin{aligned}
&\max\{\max\{a,b\},c\} = \max\{a,b,c\},\\
&Q_0 = Z_0,\\
&\max\{-a,-b\} = -\min\{a,b\}.
\end{aligned}$$

**Hint 2.6.3** It is actually not hard, even though the expression looks hard. Use condition to see that $\mathbb{P}_m(Z(t) = n) = \mathbb{P}\{N_\mu(t) - N_\lambda(t) = m - n\}$. Then write out the definitions of the two Poisson distributions. Assemble terms. Then fiddle a bit with the terms to get $(t\sqrt{\lambda\mu})$.

**Hint 2.6.4** Consider $Z(t) = Z(0) + N_\lambda(t) - N_\mu(t)$ at the embedded set of epochs at which either $N_\lambda$ or $N_\mu$ changes. What is the probability that $N_\lambda$ fires, given that one of the two Poisson processes fires? Relate this to $\{a_k\}$ and $\{c_k\}$.

**Hint 2.7.3** Remember that $\{X_k\}$ and $\{S_k\}$ are sequences of i.i.d. random variables. What are the implications for the expectations?

**Hint 2.7.4** Let $T_1 > A_1$ be the first time after the arrival of job 1 that arrives at an empty system. (Observe that job 1 also arrives at an empty system.) Suppose, for ease of writing, that this job is the $n + 1$th job, so that up to time $T_1$ the number of arrivals is $n$. Since the first job arrived at time $A_1 = X_1$, the first $n$ jobs arrived during $[X_1, T_1)$. The total amount of service that arrived during this period is $\sum_{i=1}^n S_i$. What is the fraction of time the server has been busy in this cycle?

**Hint 2.7.7** What is the rate in, and what is the service capacity?

**Hint 2.7.8** 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 $S$, $\lambda_A, \lambda_B$ 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.)

**Hint 2.7.10** Use that $A_{A(t)} \leq t < A_{A(t)+1}$. 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.

**Hint 2.8.1** Consider a queueing system with constant service and interarrival times.

**Hint 2.8.3** Why is (2.27) not the same as the number of batches that see a queue length less than $m$?

**Hint 2.9.7** For the acronym, observe that the service times and interarrival are $D$eterministic and there is one server. For the computation of $Y(n,t)$, make make a plot of $L(s)$ as a function of time for $n = 1$.

**Hint 2.10.2** $\mathbb{P}\{L \geq n\} = \sum_{k \geq n} p(k)$.

**Hint 2.11.1** Find suitable expressions for $\lambda(n)$ and $\mu(n)$ and use the level-crossing equations $\lambda(n)p(n) = \mu(n+1)p(n+1)$.

**Hint 2.11.2** 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 $n \to \infty$?

**Hint 2.11.3** Use $\lambda(n)p(n) = \mu(n+1)p(n+1)$ and find suitable expressions for $\lambda(n)$ and $\mu(n+1)$.

**Hint 2.11.4** Use $\lambda(n)p(n) = \mu(n+1)p(n+1) = \min\{c, n+1\}\mu p(n+1)$.

**Hint 2.11.5** Use that for any $x$, $x^n/n! \to 0$ as $n \to \infty$.

**Hint 2.11.8** Use $\lambda(n)p(n) = \mu(n+1)p(n+1)$, and realize that for this case $\lambda(n) = (N-n)\lambda$ and $\mu(n) = \mu$.

**Hint 2.12.2** Check the definitions of $Y(0,t)$, $Y(1,t)$, $A(0,t)$ and so on.

**Hint 2.12.3** Use that $\lambda \geq \gamma$ always, and $\lambda = \gamma$ when the system is rate-stable.

**Hint 2.12.4** Think about the construction of the $M/M/1$ queue as a random walk, see Section 2.6.

**Hint 2.12.5** Check all definitions of $Y(n,t)/t$ and so on.

**Hint 2.13.1** Think about the data that is given in either situation. Check the units when applying Little's law.

**Hint 2.13.3** Make a drawing of $A(t)$ and $D(t)$ until time $T$, i.e., the first time the system is empty.

**Hint 2.14.1** Realize again that $\mathbb{E}[S_r]$ includes the jobs that arrive at an empty system.

**Hint 2.14.2** This requires some extra definitions, similar to $A(n,t)$ as defined in (2.31a).

**Hint 2.14.4** Think about the consequences of memoryless service times.

**Hint 2.14.13** Use the level-crossing equations of the $M(n)/M(n)/1$ queue.

**Hint 2.14.14** This is a queueing system with loss, in particular the $M/M/1/1+2$ queue.

**Hint 2.14.19** Interpret each component at the right hand side of the equation and generalize it to a multi-server queueing system.

**Hint 2.15.2** This exercise is just meant to become familiar with the notation.

**Hint 2.15.3** What is the distribution of the batch size $B$ for the $M/M/1$ queue?

**Hint 2.15.4** Use Eq. 2.58. What are $\mathbb{E}\left[B^2\right]$, $\mathbb{E}[B]$ and $\mathbb{V}[B]$ for this case?

**Hint 2.15.6** $f_k = q^{k-1}p$ with $q = 1-p$. Use generating functions to compute $\mathbb{E}[B]$ and $\mathbb{E}\left[B^2\right]$

**Hint 2.15.8** Realize that the inventory process $I(t)$ behaves as $I(t) = S - L(t)$ where $L(t)$ is a suitable queueing process.

**Hint 2.15.9** Write $\sum_{n=0}^{\infty} G(n) = \sum_{n=0}^{\infty}\sum_{i=n+1}^{\infty} \mathbb{P}\{B=i\}$ and then do the algebra.

**Hint 2.15.10** $\sum_{i=0}^{\infty} iG(i) = \sum_{n=0}^{\infty} \mathbb{P}\{B=n\}\sum_{i=0}^{\infty} i1\{n \geq i+1\}$.

**Hint 2.15.11** $\mathbb{E}[S] = \int_0^{\infty} x\,\mathrm{d}F = \int_0^{\infty}\int_0^{\infty} 1_{y\leq x}\,\mathrm{d}y\,\mathrm{d}F(x)$.

**Hint 2.15.12** $\int_0^{\infty} yG(y)\,\mathrm{d}y \int_0^{\infty} y \int_0^{\infty} 1\{y \leq x\} f(x)\,\mathrm{d}x\,\mathrm{d}y$.

**Hint 2.16.11** Integrate $A^{-1}\int x\,\mathrm{d}x$, and likewise for the second moment.

**Hint 2.16.12** What is the average time between two arrivals? Observe that the interarrivals are memoryless, hence the average time until the next arrival after the server becomes idle is also $1/\lambda$.

**Hint 2.16.14** The job in the system is also in service (it is an $M/G/1/1$ queue...). The average idle time is $1/\lambda$. The average busy time is $\mathbb{E}[S]$–as there are no jobs in queue. The load $\rho$ must also be equal to the fraction of time the server is busy.

**Hint 2.16.15** 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 $\pi(0)$.

**Hint 2.16.16** Provide an example.

**Hint 2.17.4** Substitute the recursion, and carry on with the algebra. We urge the reader to try it, its good (necessary) practice. Use also the results of the exercises of Section 2.15.

**Hint 2.17.5** Divide both sides by $\mu$.

**Hint 2.17.6** This exercise tests your creativity and modeling skills, it is not analytically difficult. The best approach to problems like this is to try some simple cases first. For instance, consider the case with batch sizes of 1 first, then batches of sizes 1 or 2, and so on. If system contains $K$ jobs, which batches can be accepted? If the system contains $K-3$, say, which batch sizes can be accepted, which will be refused, under which acceptance policy?

**Hint 2.17.7** This is an important problem, it helps to check (numerically) the algebraic results we have been deriving up to now. Implementing the recursion is not hard, just try and see how far you get.

**Hint 2.18.2** 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 $k$.

**Hint 2.18.3** If $s$ is deterministic, the number of arrival during a fixed period of time with length $s$ must be Poisson distributed.

**Hint 2.18.8** Define shorthands such as $\alpha = \lambda/(\lambda+\mu)$, so that $1-\alpha = \mu/(\lambda+\mu)$, and $\alpha/(1-\alpha) = \lambda/\mu = \rho$. Then, with the previous exercise that $f(n) = \alpha^n(1-\alpha)$ and $G(n) = \alpha^{n+1}$. (I did not get this result for 'free'. Indeed, it's just algebra, but please try to derive this yourself. Its good to hone your computational skills.)

**Hint 2.19.2** What is the expected service time of one unit of a job in the $M^X/M/1$ queue in the limiting case?

**Hint 2.20.1** Take $c = 1$.

**Hint 2.20.6** Let $S_n = \sum_{k=1}^n X_k$. Then, under conditions you can find on the internet, $(S_n - n\mathbb{E}[X])/\sqrt{n\mathbb{V}[X]} \to N(0,1)$ as $n\to\infty$, where and $N(0,1)$ is a normally distributed random variable with $\mu = 0$ and $\sigma = 1$. But then, $S_n \sim \sqrt{n\mathbb{V}[X]}N(0,1) + n\mathbb{E}[X]$.

**Hint 3.1.3** Is the new machine placed in parallel or in series?

**Hint 3.1.4** Check your reasoning very carefully here. It is easy to do it wrong.

**Hint 3.2.1** The final result is $f_D(t) = f_{X+S}(t)\mathbb{P}\{I\} + f_S\,\mathbb{P}\{B\} = (1-\rho)f_{X+S}(t) + \rho\mu e^{-\mu t}$. Now use the above to simplify and see that $f_D(t) = \lambda e^{-\lambda t}$.

**Hint 3.3.2** Try to adapt the ideas behind Figure 2.2 of Zijm to this case.

**Hint 3.4.1** Check the formulas in the synthesis carefully.

**Hint 4.1.8** First write down all different states, and then use Zijm.Eq.3.2 and 3.3.

**Hint 4.2.7** Compare Zijm.Question 3.1.5.

**Hint 4.3.1** Start with $n = 1$, then consider $n = 2$ and so on.

**Hint 4.4.1** Just follow the description of the algorithm of the book of Zijm. I expect that you have read the solution of Question 1.

# 6 Solutions

**Solution 3.1.5** Because $T_0$ increases and/or the bottleneck rate increases. In both cases, we need more $W_0$ to fill the network and achieve that the throughput can be equal to $r_b$ in the best case.

**Solution 3.1.6** Now the slow machine at station 2 becomes superfluous. The fast machine at station 2 can cope with all jobs that arrive from station 1.

$$\begin{aligned} r_b &= \frac{1}{2} \\ T_0 &= 2 + 1 + 2 = 5 \\ W_0 &= r_b T_0. \end{aligned}$$

Thus, why would you assign part of the jobs to the slow machine at station 2? There is no queue, the fast machine can cope with all the jobs. Moreover, assigning any job to the slow machines just adds to the cycle time.

Taking $T_2 = (1+3)/2 = 2$ is plainly silly.

**Solution 3.1.8**

$$r_2 = \frac{1}{3} + \frac{1}{7} = \frac{10}{21} < \frac{1}{2}$$

hence station 2 is still the bottleneck

$$\begin{aligned} T_2 &= \frac{3}{10}7 + \frac{7}{10}3 = \frac{42}{10} = \frac{21}{5}\text{hour} \\ T_0 &= 2 + \frac{21}{5} + 2 \\ r_b &= \frac{10}{21} \\ W_0 &= r_b T_0. \end{aligned}$$

Note that we add capacity, so that we can process more jobs per unit time, but the raw processing time increases!

**Solution 3.1.10** Now stations 1 and 3 are bottlenecks with $r_b = 1/2$. For $T_0$ we have

$$T_0 = \frac{1}{5}7 + \frac{4}{5}4 = \frac{23}{5}.$$

Thus,

$$W_0 = r_b T_0 = \frac{1}{2}\frac{23}{5} = \frac{23}{10}.$$

# 7 Notation

| | | |
|---|---|---|
| $a_n$ | = | Number of arrivals in the $n$th period |
| $A(t)$ | = | Number of arrivals in $[0,t]$ |
| $A_k$ | = | Arrival time of $k$th job |
| $c_n$ | = | Service/production capacity in the $n$th period |
| $d_n$ | = | Number of departures in the $n$th period |
| $c$ | = | Number of servers |
| $C_a^2$ | = | Squared coefficient of variation of the interarrival 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 $[0,t]$ |
| $D_n$ | = | Departure time of $n$th job |
| $D_{Q,n}$ | = | Departure time of the queue of $n$th 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 $t$ |
| $\mathbb{E}[L]$ | = | Long run (time) average of the number of jobs in the system |
| $\mathbb{E}[Q]$ | = | Long run (time) average of the number of jobs in queue |
| $\mathbb{E}[L]_S$ | = | Long run (time) average of the number of jobs in service |
| $N(t)$ | = | Number of events in $[0,t]$ |
| $N(s,t)$ | = | Number of events in $(s,t]$ |
| $p(n)$ | = | Long-run time average that the system contains $n$ jobs |
| $Q_n$ | = | Queue length as seen by the $n$th job, or at the *end* of the $n$th period |
| $S_k$ | = | Service time required by the $k$th job |
| $S(t)$ | = | Total service time available in $[0,t]$ |
| $S$ | = | generic service time of a job |
| $t$ | = | time |
| $W_n$ | = | time in the system of $n$th job |
| $W_{Q,n}$ | = | time in the queue of $n$th job |
| $\mathbb{E}[W]$ | = | sample average of the sojourn time |
| $\mathbb{E}[W]_Q$ | = | sample average of the time in queue |
| $X_k$ | = | inter-arrival time between job $k-1$ and job $k$ |
| $X$ | = | generic interarrival time between two consecutive jobs |
| $\gamma$ | = | departure rate |
| $\lambda$ | = | arrival rate |
| $\mu$ | = | service rate |
| $\pi(n)$ | = | Stationary probability that an arrival sees $n$ jobs in the system |
| $\rho$ | = | Load on the system |

# 8 Formula Sheet

$$\rho = \lambda \frac{\mathbb{E}[S]}{c}$$

$$\mathbb{E}\left[W_Q\right] = \frac{C_a^2 + C_s^2}{2} \frac{\rho^{\sqrt{2(c+1)}-1}}{c(1-\rho)} \mathbb{E}[S]$$

$$C_e^2 = C_0^2 + 2A(1-A)\frac{m_r}{\mathbb{E}[S_0]}$$

$$\sigma_e^2 = \sigma_0^2 + \frac{\sigma_s^2}{N_s} + \frac{N_s - 1}{N_s^2}(\mathbb{E}[S_s])^2$$

$$\sigma_e^2 = \sigma_0^2 + \sigma_r^2 f_r + f_r(1-f_r)(\mathbb{E}[S_r])^2$$

$$C_{sB}^2 = \frac{\mathbb{V}[S]_s + N\,\mathbb{V}[S]_0}{(\mathbb{E}[S_s] + N\,\mathbb{E}[S_0])^2}$$

$$f_i(n_i) = \begin{cases} G(i)^{-1}(c_i\rho_i)^{n_i}(n_i!)^{-1}, & \text{if } n_i < c_i, \\ G(i)^{-1}c_i^{c_i}\rho_i^{n_i}(c_i!)^{-1}, & \text{if } n_i \geq c_i \end{cases}$$

$$G(i) = \sum_{n=0}^{c_i-1} \frac{(c_i\rho_i)^n}{n!} + \frac{(c_i\rho_i)^{c_i}}{c_i!}\frac{1}{1-\rho_i}$$

$$\mathbb{E}[L_i] = \frac{(c_i\rho_i)^{c_i}}{c_i!G(i)}\frac{\rho_i}{(1-\rho_i)^2} + c_i\rho_i$$

$$C_{di}^2 = 1 + (1-\rho_i^2)(C_{ai}^2 - 1) + \frac{\rho_i^2}{\sqrt{c_i}}(C_{si}^2 - 1)$$

$$C_{ij}^2 = P_{ij}C_{di}^2 + 1 - P_{ij}$$

$$Q_{ij} = \frac{\lambda_{ij}}{\lambda_j}$$

$$C_{aj}^2 = w_j \sum_{i=0}^{M} Q_{ij}C_{ij}^2 + 1 - w_j$$

$$w_j = (1 + 4(1-\rho_i)^2(v_j - 1))^{-1}$$

$$v_j = \left(\sum_{i=0}^{M} Q_{ij}^2\right)^{-1}, j = 1, \ldots, M,$$

$$f_i(n_i) = \frac{1}{\Pi_{k=1}^{n_i}\min\{k, c_i\}}\left(\frac{V_i}{\mu_i}\right)^{n_i}$$

$$G(m,n) = \sum_{k=0}^{n} f_m(k)G(m-1, n-k)$$

$$G(m,0) = 1, G(0,n) = \mu_0^{-n},$$

$$\mathbb{E}[W]_j(n) = \mathbb{E}\left[L_j(n-1) + 1\right]\mathbb{E}[S]_j$$

$$\mathrm{TH}_0(n) = \frac{n}{\sum_{j=0}^{M} V_j \mathbb{E}\left[W_j(n)\right]}$$

$$\mathrm{TH}_j(n) = V_j \mathrm{TH}_0(n)$$

$$\mathbb{E}\left[L_j(n)\right] = \mathrm{TH}_j(n)\mathbb{E}\left[W_j(n)\right]$$

$$p_j(0|0) = 1$$

$$\mathbb{E}\left[W_j(n)\right] = \sum_{k=c_j}^{n-1} \frac{k - c_j + 1}{c_j\mu_j} p_j(k|n-1) + \frac{1}{\mu_j}$$

$$\mu_j \min\{c_j, k\} p_j(k|n) = \mathrm{TH}_j(n) p_j(k-1|n-1)$$

$$p_j(0|n) = 1 - \sum_{k=1}^{n} p_j(k|n).$$