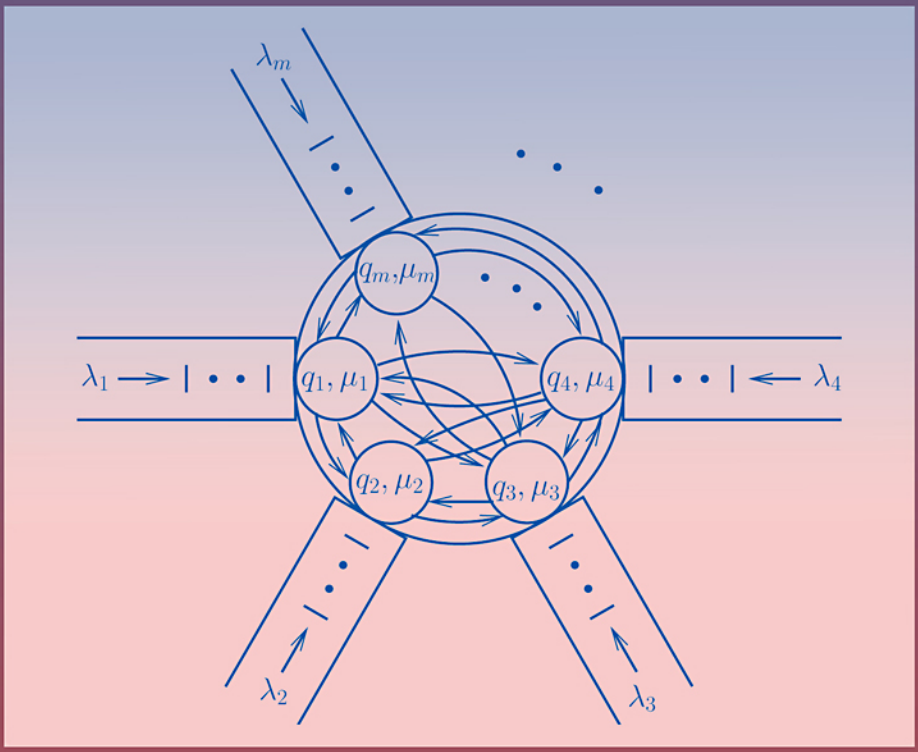
CHAPMAN & HALL/CRC COMPUTER and INFORMATION SCIENCE SERIES

Performance Analysis of Queuing and Computer Networks



G. R. Dattatreya

CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Performance Analysis of Queuing and Computer Networks

CHAPMAN & HALL/CRC
COMPUTER and INFORMATION SCIENCE SERIES

Series Editor: Sartaj Sahni

PUBLISHED TITLES

ADVERSARIAL REASONING: COMPUTATIONAL APPROACHES TO READING THE OPPONENT'S MIND
Alexander Kott and William M. McEneaney

DISTRIBUTED SENSOR NETWORKS
S. Sitharama Iyengar and Richard R. Brooks

DISTRIBUTED SYSTEMS: AN ALGORITHMIC APPROACH
Sukumar Ghosh

FUNDEMENTALS OF NATURAL COMPUTING: BASIC CONCEPTS, ALGORITHMS, AND APPLICATIONS
Leandro Nunes de Castro

HANDBOOK OF ALGORITHMS FOR WIRELESS NETWORKING AND MOBILE COMPUTING
Azzedine Boukerche

HANDBOOK OF APPROXIMATION ALGORITHMS AND METAHEURISTICS
Teofilo F. Gonzalez

HANDBOOK OF BIOINSPIRED ALGORITHMS AND APPLICATIONS
Stephan Olariu and Albert Y. Zomaya

HANDBOOK OF COMPUTATIONAL MOLECULAR BIOLOGY
Srinivas Aluru

HANDBOOK OF DATA STRUCTURES AND APPLICATIONS
Dinesh P. Mehta and Sartaj Sahni

HANDBOOK OF DYNAMIC SYSTEM MODELING
Paul A. Fishwick

HANDBOOK OF PARALLEL COMPUTING: MODELS, ALGORITHMS AND APPLICATIONS
Sanguthevar Rajasekaran and John Reif

HANDBOOK OF REAL-TIME AND EMBEDDED SYSTEMS
Insup Lee, Joseph Y-T. Leung, and Sang H. Son

HANDBOOK OF SCHEDULING: ALGORITHMS, MODELS, AND PERFORMANCE ANALYSIS
Joseph Y.-T. Leung

HIGH PERFORMANCE COMPUTING IN REMOTE SENSING
Antonio J. Plaza and Chein-I Chang

PERFORMANCE ANALYSIS OF QUEUING AND COMPUTER NETWORKS
G. R. Dattatreya

THE PRACTICAL HANDBOOK OF INTERNET COMPUTING
Munindar P. Singh

SCALABLE AND SECURE INTERNET SERVICES AND ARCHITECTURE
Cheng-Zhong Xu

SPECULATIVE EXECUTION IN HIGH PERFORMANCE COMPUTER ARCHITECTURES
David Kaeli and Pen-Chung Yew

Performance Analysis of Queuing and Computer Networks

G. R. Dattatreya
University of Texas at Dallas
U.S.A.

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
A CHAPMAN & HALL BOOK

Cover graphic represents the queing network for the contention-free channel access problem in Exercise 20, Chapter 5.

Chapman & Hall/CRC
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed in the United States of America on acid-free paper
10 9 8 7 6 5 4 3 2 1

International Standard Book Number-13: 978-1-58488-986-1 (Hardcover)

Library of Congress Cataloging-in-Publication Data

Dattatreya, G. R.
Performance analysis of queuing and computer networks / author, G.R. Dattatreya.
p. cm. -- (Chapman & hall/CRC computer and information science series)
"A CRC title."
Includes bibliographical references and index.
ISBN 978-1-58488-986-1 (hardback : alk. paper) 1. Computer networks--Evaluation. 2. Network performance (Telecommunication) 3. Queuing theory. 4. Telecommunication--Traffic. I. Title.

TK5105.5956D38 2008
004.6--dc22 2008011866

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the CRC Press Web site at
http://www.crcpress.com

To my family

Contents

Preface

The principles used in the design, operation, and interconnections of data communication networks have been mature for well over a decade. The technology is very pervasive and upgrades to the equipment are very frequent. Therefore, a first course on the topic of computer networks is very useful for students intending to professionally work with this technology. Indeed, the vast majority of undergraduate students majoring within and bridging the electrical engineering and computer science disciplines study a course on computer networks. Simultaneously, a course on probability theory, required for such students, has generally expanded to include some material on queues, a fundamental topic in performance analysis of data communication networks. Alternatively, many undergraduate degree programs within these disciplines offer a follow up course, after probability theory, covering related topics including queues. However, in both these scenarios, a common observation is that queues are not taught with a systematic development of even the elementary results. Even if the subject has a chapter on Markov chains, the balance equations are written in a hurried fashion and students get a false impression that it is a rigorous development. Two examples of additional pitfalls are the following. Students get the false impression that they have formally derived the result that a stable queue reaches equilibrium. They also find it obvious that the departure process of an M/M/1/∞ queue is Poisson. While many such results are indeed true, there is a dangerous tendency to believe that the results extend to other similar but more general cases of queues and Markov chains.

Books and formal courses on stochastic processes or queuing theory generally dwell on the systematic development of the mathematical principles governing various types of Markov chains to force conclusions on when such desirable results are true and when they are not. This approach appears to be abstract, long-winded, and even graduate students in applied sciences and engineering tend to feel lost in a maze. Also, in such an approach, at the end of an abstract approach to Markov chains, simple queues are trivial examples and are not treated at length. Furthermore, in both the above approaches, only very simple examples from the application area of computer networks are introduced. The typical student completes the course with the frustration that only some formulas were given in the course. Instructors, on the other hand, form the following erroneous opinions about students. (*a*) They are impatient and do not realize the value of the mathematical principles governing even the simplest of queues. (*b*) They don't realize that practical systems are more complicated variations or interconnections of simple systems and that simple systems should be thoroughly understood first. (*c*) They just want some magical formulas not only for simple queues, but also for practical telecommunication systems they will encounter

in their job-related activities. (*d*) They don't realize that each practical application system is different, and without a complete specification, it cannot be analyzed, even if such an analysis is feasible with skills available to students.

This book attempts to strike a balance between (*i*) mathematical skills of incoming students, (*ii*) mathematical skills that can be taught as part of this course, (*iii*) generality, (*iv*) rigor, (*v*) focus, (*vi*) details, and (*vii*) model formulation for application systems in computer networks.

Its prerequisites are well specified as follows. College mathematics including differential and integral calculus, elementary matrix theory (but not linear algebra), and a course on elementary probability theory. Principles of stochastic processes and advanced matrices (such as eigenvalue theory) are *not* assumed to be known to students. Throughout the book, the development is motivated and illustrated by examples and exercises in computer systems and networks. Mathematical derivations are part of the material; however, focus is maintained by splitting the development of a sequence of results into smaller tasks and discussing the role of the results in the big picture at every step. Also, final results are prominently restated with the appropriate conditions for their validity. Examples that violate the conditions and hence do not enjoy the corresponding results are included. Therefore, the book is self contained and can also serve as a reference for practicing engineers. As a consequence, only a short bibliography of mostly unreferenced books is included.

An additional advantage of this approach is that instructors and students can opt for detailed coverage of some topics while summarily browsing through the mathematical development of others and quickly moving onto applications. That is, the instructor can choose the level of detail and emphasize on different sets of subtopics. Therefore, even though the material may appear to be too vast for a one semester course, selection of topics is easy.

Many concepts and results of probability theory and stochastic processes are developed with the help of queues as applications. This avoids unnecessary abstractness and allows treating many different types of queues that appear in computer networks over a shorter time. This approach gives students motivation to study the needed principles and results. Every such development uses no more than the stated college mathematics (listed above) and principles thus far developed in the book, except in the final two chapters on advanced material. The book uses alternative and simpler techniques, in many places, to avoid using results from higher (say graduate level) mathematics. This avoids undue generality and keeps the focus on necessary results.

The material in the book begins by describing queues and with fairly extensive descriptions of activities in computer systems and networks resulting in various types of queues to motivate the students. Appendix A is a brief but rigorous and self contained review of elementary probability theory with examples and exercises.

Chapter 2 is devoted to traffic models. Pareto random variable is introduced as a model for either inter-arrival time or for service time in some computer network queues. The development also serves as a warm-up exercise in the use of probability theory. Poisson and exponential random variables are systematically developed from a practical source that emits jobs or electrons at random and with a constant

rate. All their properties are developed. Simulation is introduced and the transformations from a uniformly distributed random variable to generate other important random variables are developed. Simple concepts of parameter estimation are also developed. Mean square convergence of a sequence of random variables is introduced as a natural topic in estimation. This finds use later in the analysis of sample functions of Markov chains and in the development of the Little's result. A very simple model for error-prone data channels is developed. The model is fully specified if the bit error rate at any data transmission rate is known. It is demonstrated with a throughput optimization example.

Chapter 3 is on equilibrium M/M/1/∞ queue. Properties of Poisson and exponential random variables developed in Chapter 2 are heavily used. The equilibrium solution is systematically developed (without using any concepts from stochastic processes). To retain interest in equilibrium solution, it is shown that if such a system is in equilibrium at some time instant, it will remain so for all the time to come. To illustrate that we can construct practical models from simple (but not necessarily practical) models, a round robin version of M/M/1/∞ queue with non-vanishing piecemeal service times is introduced and all the results are systematically developed. This also allows for a simple analysis of a data link affected by erroneous packets which are required to be retransmitted. The Poisson nature of the departure stream of an M/M/1/∞ system is proved without using reversibility. This result is important to students for two reasons. It validates the assumption that packet arrivals into a queue can be Poisson even if bits and hence packets arrive over nonzero time intervals. Also, that the output stream can be fed in its entirety or through a probabilistic split to another queue as Poisson inputs. That is, a feed-forward network of M/M/1/∞ queues can be analyzed with the help of results on individual M/M/1/∞ queues. The non-Poisson nature of the merged stream of customers arriving at the waiting line of a round robin scheme is also shown. The probability density function and the Laplace transform of the busy time periods in an M/M/1/∞ queue are systematically developed. All the results on M/M/1/∞ queues are mathematically developed without using (and before introducing) the concept of stochastic processes. Any use of the term " average" of a random variable refers to its expectation and is clear from the context. As a consequence of the use of random variables only (and not random processes), Little's result, which is on time averages, is not introduced or used in this chapter.

Chapter 4 is on continuous time, state dependent single Markovian queues. The definitions and elementary concepts of stochastic processes are easily developed with the help of a queue as an application example. Continuous parameter Markov chains are introduced with the M/M/1/∞ queue as an example. Balance equations for the equilibrium state probabilities of an irreducible chain are derived by first deriving the differential equations, just as is done for the case of M/M/1/∞ queue. This is rigorous, and it also reinforces the concepts developed earlier. The conclusion is that if the balance equations result in a unique solution for the state probabilities, we have a nice Markov chain that can be in equilibrium and whose equilibrium performance figures can be evaluated. The general development of uniqueness of solution for a positive recurrent Markov chain is deferred to a later chapter. This decision is motivated by

the desirability of an early introduction of a rich class of application systems in the computer networks area. An intuitive approach to develop the results for long-term time averages is followed by a thorough and rigorous development. Little's result is proved for FIFO and non-FIFO systems. In addition to the usual state dependent application examples with finite buffers and multiple servers, a very simple model of analysis of a heavily loaded Carrier Sense Multiple Access with Collision Detection (CSMA/CD) system is developed. Justification for the heavily loaded assumption is made by arguing that the individual stations attempt to transmit control packets when payload packets are absent in the buffer. The model and its utility from this example are comparable to the simplistic analysis of continuous time ALOHA to derive the maximum possible throughput, taught in a first course on computer networks. A similar system for CSMA/CA wireless LANs is completely described in exercises for students to analyze. A contention-free CSMA LAN performance analysis problem with a finite number of transmitting stations and heterogeneous arrival rates is similarly formulated. Its analysis and performance optimization is carried out. Other interesting examples in computer systems and networks are also included. Illustrative exercises on computer network performance analysis are listed.

Chapter 5 is on the M/G/1 queue. The recurrence equations for the state sequence of the imbedded (embedded) Markov chain of an M/G/1/∞ queue are developed. The uniqueness of solution to the resulting equilibrium balance equations is easily shown. The equilibrium state probabilities at departure time instants being the same as the expected long-term time averages of state occupancies is shown with the help of the PASTA property, which is also developed. The Pollackzec-Khinchin mean value formula is completely derived without developing or using the corresponding transform formula. The expected time averages of state occupancies for a finite buffer M/G/1 queue are also developed. The contention-free LAN performance analysis problem with heterogeneous arrival rates, first studied in Chapter 4, is generalized in the exercises here, to allow for heterogeneous packet sizes. This is a useful feature in Voice Over IP (VOIP) application.

Chapter 6 is on discrete time queues. A detailed analysis of timing within and across slots is very important to understand the various possible and impossible events concerning arrivals to and departures from empty and full systems. The analysis leads two different Markov chains, for the states, at slot centers and slot edges, respectively. State classification is developed with practical examples from computer systems. Existence and uniqueness of the solution of equations for equilibrium state probabilities is shown without using advanced linear algebra or advanced matrix theory. Interrelationships between these Markov chains are developed for students to clearly identify the correct quantities to be used to obtain the performance figures. Interesting examples from synchronous digital systems are used to illustrate the topic. Examples and exercises on the topic of slotted networks and sensor networks are also included.

Chapter 7 is on continuous time Markovian queuing networks. The case of open queuing networks is studied first. The Markovian nature of such systems is pointed out. Balance equations and traffic equations are developed. The product form solution is verified to hold. Illustrative properties and examples are included. For closed

queuing networks, in addition to the verification of the product form solution, convolution algorithm, performance figures, and mean value analysis are developed with the necessary details. Illustrative properties and application problems are included.

Chapter 8 is on G/M/1 queues. The imbedded Markov chain of the G/M/1/∞ queue is analyzed. Results are specialized to Pareto interarrival times (IAT). The effective load as a function of normalized load and the Hurst parameter of the Pareto IAT are very illustrative; the average buffer occupancies are considerably worse than those in M/M/1/∞ queues for the same load. Furthermore, these averages steeply increase as the Hurst parameter increases towards 1. These results bring out the bursty nature of data traffic with Pareto IAT. The derivations use no results from outside and are fairly easy to follow, although obtaining the Laplace transform for a Pareto IAT is somewhat lengthy. Evaluation of equilibrium state probabilities at arrival time instants in a finite buffer G/M/1 queue is straightforward and included. From these, packet drop rates (due to the finite buffer), expected response time, and average queue size are easy to evaluate.

Chapter 9 introduces and analyzes a few bursty traffic models and their effects on queues. Chapter 10 introduces fluid-flow models and their analyses. These topics are considered somewhat advanced and the treatment here does use matrix theory and systems of ordinary differential equations. The motivation, model development, and relations to other models are nevertheless simple to follow, as are the final developed results. A conscious attempt is made to develop the advanced mathematical results as and when needed. Only very occasionally is a reference made to a specific advanced result in the literature, listed in the short bibliography.

Chapter 9 is devoted to bursty traffic and corresponding queues. Principles of smooth and bursty traffic are introduced with the help of simple probability theoretic principles. In the literature, exact results on queues input with some models of bursty traffic have been elusive even with sophisticated mathematical tools. A tractable approximation to self-similar traffic is developed as follows. Merging numerous (theoretically, unbounded number of) streams of traffic with heavy-tailed IAT is known to result in a self-similar data source. In this chapter, the heavy-tailed Pareto random variable is approximated by a hyperexponential random variable. Merging several such data packet streams (each with a hyperexponential IAT) results in a Markovian Arrival Process (MAP) with a very large number of states. This Markov chain is shown to sport a product form solution which is evaluated with the help of an efficient algorithm. This also introduces state dependent closed queuing networks. A queue fed by such a packet source is analyzed. The complexity of the solution for the queue depends only on the number of states in the Markov chain of the data source. Matrix inversion is not required here. The complete analysis of such a queue is based on the original work of Marcel Neuts which deals with a more general system. Queues fed by data packet streams generated by a Markov modulated Poisson process (MMPP) are similarly but briefly analyzed. Evaluation of results on a queue input by an MMPP requires inversion of a square matrix with the number of rows equal to the number of states in the MMPP. Some results are left for students to develop and are listed in exercises. The product form solution developed here for closed networks with stations that offer immediate service expands the applicability

of closed networks. Some interesting application problems on the topic of cognitive radio networks are formulated in exercises.

The final chapter, Chapter 10, is on fluid flow models. Data packets are considered to flow into a buffer at a rate that can switch from one value to another over a countable set of rates. The output from the buffer has similar features. These rates change in a continuous time Markov chain fashion. The analysis technique is first introduced with a two state ON-OFF Markov chain model of a packet train feeding into a leaky-bucket with a constant draining rate. An illustrative example demonstrates all the aspects of solution development for this two state Markov chain fluid input problem. Differential equations for the cumulative distributions of the buffer content in the general case of multistate Markov chain controlling the input and draining rates are formally developed. Solution follows the earlier developed eigenvalue-eigenvector approach. Little's result for the general case of a stable fluid flow system is systematically developed. If the number of states of the Markov chain controlling the flow rates is infinity, a matrix-method solution is not possible, in general. The simplest case of an infinite state Markov chain controlling the flow rates is the output of an M/M/1/∞ queue feeding a constant rate leaky bucket. This is analyzed and illustrated with a variation of the first example. Comparison of the two different but similar systems is very illustrative.

I would like to express my appreciation and gratitude to many people who have directly and indirectly helped me through the development and preparation of this book. My wife Manorama has been very supportive and freed me from the many day-to-day concerns that would otherwise have impeded progress. She has willingly endured my unpredictable hours of work day and night. I thank her from the depths of my heart. My son Madhur's eagerness to see this book published provided additional motivation. Growing up, my parents, brothers, and sisters instilled in me a deep appreciation for education and critical thinking. I am indebted to all of them.

I have taught several sections from the first seven chapters to numerous students at the University of Texas at Dallas. Discussions with them and their questions and feedback have contributed to the way I treat the topics in this book. I have used some material from the research publications of my former Ph.D. students Sarvesh Kulkarni and Larry Singh. They were my teaching assistants for a few semesters each and have helped me in other ways with this book. Early versions of sections from some of the chapters were prepared as notes for an online course through a grant from the Telecampus program of the University of Texas System. Larry Singh prepared those electronic notes. R. Chandrasekaran and Shun-Chen Niu have spent a lot of time with me answering my questions on mathematics in general and on queues and Markov chains in particular. I am very thankful to them.

I thank Marwan Krunz of the University of Arizona, Sartaj Sahni of the University of Florida, and Medy Sanadidi of the University of California at Los Angeles for their early reviews on a few chapters. I thank Sartaj Sahni, the series editor, additionally, for including this book in the Series on Computer and Information Science. Finally, I thank the editorial and publishing staff of Taylor & Francis, in particular, Theresa

Delforn, Shashi Kumar, Amy Rodriguez, and Bob Stern, for their timely assistance and cooperation.

I am solely responsible for errors and omissions in this book. A publisher's website is planned to receive and announce errata. I will be grateful for any criticism and suggestions for corrections I receive.

G. R. Dattatreya

Short Bibliography

1. D. Gross and C. M. Harris, *Fundamentals of Queueing Theory*. Wiley Series in Probability and Statistics, 1998.

2. F. P. Kelly, *Reversibility and Stochastic Networks*. John Wiley, 1979.

3. L. Kleinrock, *Queueing Systems. Volume I: Theory*. Wiley Interscience, 1975.

4. L. Kleinrock, *Queueing Systems. Volume II: Computer Applications*. Wiley Interscience, 1976.

5. M. F. Neuts, *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. Baltimore, MD: Johns Hopkins University Press, 1981.

6. A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*. NY: McGraw Hill Higher Education, 2002.

7. K. S. Trivedi, *Probability and Statistics with Reliability, Queueing, and Computer Science Applications*. Wiley-Interscience, 2001.

8. R. W. Wolff, *Stochastic Modeling and the Theory of Queues*. Prentice Hall, 1989.

Chapter 1

Introduction

1.1 Background

A queue is an arrangement for the members of a set to appear for an activity, complete it, and leave. Such appearances are called *arrivals*. The activity is called *service*. The members arriving for service are called *customers*, even though they may not be humans in every case. Customers may be physical devices, or even abstract entities such as electromagnetic signals representing a data packet. The arrangement is also called a *queueing system*. The word queueing is also spelled *queuing*, now-a-days. Queues occur extensively in all walks of life and in many technological systems. They gained importance in machine shops with a demand for quick repair turn around during World War II. The simplest examples of queues are those in banks with customers being served by tellers, calls appearing at telephone exchanges, and population dynamics of, say, rabbits and foxes in a forest.

The following are some common features in a queuing system. Arrival time instants are usually uncertain, with a statistically steady behavior of the time intervals between successive arrivals. Similarly, the service times are also usually uncertain with a statistically steady behavior. Customers may wait in a *waiting line* to receive service. In the simplest arrangement, service is provided in a *first-in, first-out* (FIFO) order. In such a system, the customer receiving service is said to be at the *head* of the queue and a fresh arrival joins the *tail* of the queue. A customer *departs* from a queue after receiving service. In another type of arrangement, service is provided in parts or *piecemeal* with a customer typically alternating between the waiting mode and the service mode, returning to the tail of the waiting line after a piece of service. The customer leaves the entire system at the end of the complete service, possibly after many time intervals of piecemeal service, separated by time intervals of waiting. Queues with *last-in, first-out* (LIFO) service, and service in *random order* are also found in practice. An LIFO arrangement is commonly referred to as a stack (instead of being called a queue). In some applications, multiple customers may receive service simultaneously, with the help of multiple servers in the system. There may also be multiple waiting lines with customers moving from one queue to another. Such systems with interacting queues are called *queuing networks*. In such queuing networks, customers may move from the departing point of one queue to the tail of another. A customer may return to the tail of the departing queue itself. A customer may also arrive at the tail of an earlier visited queue for additional service. After

possibly many such visits to multiple queues, a customer finally leaves the entire network.

Individual computers and computer networks abound with queues. Statistical averages of various quantitative criteria governing such queues are useful to assess the acceptability of the performance. Their evaluations are also useful to optimize the performance by tuning control parameters and to determine the number and qualities of processors and other servers required to achieve an acceptable degree of performance, in applications. Several examples of queuing in computers and their networks are described in the following section, to motivate a detailed study of the subject.

1.2 Queues in Computers and Computer Networks

1.2.1 Single processor systems

A computer processes jobs submitted to it by a user. Many of these jobs are ready-made computer programs that a user initiates through a keyboard command or by pointing the computer mouse pointer at a representative icon and clicking it. Internally, the main monitor program, called the *operating system* (OS) itself keeps the computer busy to a certain extent with housekeeping operations, even when there is no external job to process. For example, checking to see if any program is initiated by a user is a house-keeping operation. If a user strikes a key on the keyboard, that information stays in a memory buffer; the fact that the computer's attention has been called to the data-input device (keyboard) is stored in another buffer. The OS lets the computer to frequently check these buffers called the input ports. Input and output (I/O) between the computer and the external devices are through organized handshake procedures with the computer and the I/O device having a full knowledge of whose turn it is to respond and how, for every step of the process. When an external input device has submitted a request, the OS invokes one or more programs to examine the request and processes the same.

Most individual computer systems are built around a single processor each. Such a processor is called the Central Processing Unit (CPU). Even if the processor has pipelined or vector processing hardware, machine instruction executions are completed one by one in such machines. However, the CPU gives attention to segments of many different programs, in sequence. That is, whereas the machine instructions are executed one after another, the execution of program jumps from one subsequence of instructions in a program to another subsequence of a *different* program. The scheduling algorithm for such jumps between different programs is influenced by a variety of factors such as which Input/Output (I/O) device becomes active during an execution period. Even when there is no such external stimuli during a time period, the OS changes the CPU's attention from one program to another, with the help of internal timers. This feature is deliberately incorporated so that the execution of a short program is not completely held up while the CPU completes the execution

of a very long program.

The machine instruction execution is relatively very fast in comparison with the usual speed at which the external requests draw the attention of the computer. Therefore, many times, the user feels that the computer is processing all the requests simultaneously, and hence the terms "multiprogramming" and "time sharing" are used to describe the operation of such a single computer system.

The queue in such a single computer system consists of arrivals of external jobs or requests submitted by the user. The server is the CPU giving piecemeal attention to the requests. Partially processed requests are sent back to the tail of the queue, whenever the CPU decides to change its attention to the next job in the queue. Such processing and queuing systems are referred to by the name *round robin*. More complicated queuing systems can be formulated by accounting for the interaction of the I/O devices and the CPU.

1.2.2 Synchronous multi-processor systems

Multiple computers are synchronously interconnected in some specialized systems to allow parallel processing. In such systems, all activities and data movement are controlled by a single master-clock that ticks at a constant rate. There may be other clocks synchronized with the master-clock. There may be a single or multiple service points. The number of master clock cycles, also known as *slots*, can vary from one invocation of a program to another. Statistical averaging of the performance metrics are useful to assess the overall systems. In such a system, a sequence of programs arrives and processing is FIFO, leading to a simple queue. However, the slotted operation requires the quantity "time" to be treated as a discrete variable.

1.2.3 Distributed operating system

In many other applications, several computers, terminals, and workstations, all generally referred to as *clients*, are connected to one or a few high performance computers called the servers. Client machines may process many jobs themselves. They may also ship jobs to the servers when deemed necessary. All the activities are controlled by a loosely coupled *distributed operating system* (DOS). There is no master-clock controlling the movement of customers; hence the time variable is a continuous one. In this configuration, jobs or requests may wait for various types of service at multiple locations. Therefore, there are several queues in such a system. Jobs may also visit service points repeatedly, due to the time sharing organization mentioned earlier. The overall organization is a network of queues.

1.2.4 Data communication networks

1.2.4.1 Data transfer in communication networks

In data communication networks, computers, called host machines are interconnected by a system of communication links. The interconnected system of links, not

including the host machines, is known as the subnet. The host machines run application programs that require movement of data between different computers. All the computers are independent devices and there is no single DOS controlling the computers. The primary purpose is data transfer between computers which are possibly geographically separated by hundreds or thousands of kilometers. The process of data transfer requires running computer programs such as format conversion, proper I/O, etc., but the applications themselves are not generally computation-intensive. The level of cooperation is at a higher level in the sense that data transfer of every single item is not a tightly controlled handshake procedure. The following example illustrates the above situation. In an ongoing data transfer, the computer receiving data from an incoming data link is generally ready for the task. However, over a particular short time interval, it may not have processed all the received data available on its input ports. Several bytes of additional data may arrive in a quick sequence. In such a case, the newly arrived data may write over existing data in the input ports. If the recipient computer is configured not to accept data on input ports until existing data are processed, the newly arriving data will simply not be entered into any input ports and vanish! This demonstrates that such a computer network is less reliable than a tightly controlled interconnection between a single computer and its I/O devices. Another source of lack of reliability is the bit errors possibly introduced due to noise over long data links, especially over wireless networks. Such lack of reliability is taken into consideration and programs running on the computers attempt to compensate for the same through the use of error detection, acknowledgments, and retransmissions. These slow down the overall data transfer processes creating the necessity of queuing. If the overall data movement is not efficient enough, queuing delays will accumulate. The long queues necessitate very large buffers in which to hold waiting data. This becomes impractical, even if we resign ourselves to tolerate longer overall delays. Therefore, data transfer in practical computer networks is required to be very efficient.

1.2.4.2 Organization of a computer network

The overall network has a hierarchical structure with a backbone subnet made of a small number of high data rate links. A data link connects two routers. A router is a high speed special purpose computer, but it is not a host machine. A router can support multiple links, going in different directions. Each link is usually bidirectional, and can be equivalently considered to be two unidirectional links in opposite directions. Each router in the backbone subnet in turn feeds into different portions of the network. Each such portion itself is an interconnection of routers realized with the help of data links. Each of these routers feeds into one or more *local area networks* (LANs). A LAN uses a single broadcast medium through which several host computers communicate among themselves. One single computer on the LAN also functions as a LAN server to facilitate communication between the other host computers on the LAN and the rest of the world.

In data networks, communication between host machines is not in a contiguous stream of bits. An overall communication of a large file is accomplished by splitting

the file into individual *data items*, with each data item consisting of a stream of several bits. The number of bits in a data item can range from hundreds to several thousands. Data items are *transmitted* from one point to the next over links. All the data items belonging to a file to be transferred do not necessarily go through the same sequence of links and do not appear at the eventual destination in the exact same order of transmission at the original source. Software in the original source host and the eventual destination host cooperate to reassemble data items to reconstruct the original file. Such software at each of the hosts of the origin of the file and the destination are called *transport layer* software. Thus, even the software for the data communication over a computer network is organized in a hierarchical way with different software modules responsible for different activities. Each *layer* of the overall software appends additional bits called *headers* to a data item to manage the transfer of a data file to the eventual destination. Several headers are added and removed in the course of the overall transfer of a data item. At the transport layer, a data item including its header is called a *transport data unit* or TPDU. The network layer is responsible for decisions on which data link a data item should be transmitted. At the network layer, a data item is called a *packet*. Between the end points of a single link, the *datalink layer* (DLL) software manages error correction, verification of successful transfer, etc. The data items in this layer are called *data frames*. The *medium access control layer* (MAC) manages data transfer over a broadcast link such as a LAN. The primary problems encountered by the MAC layer are cooperative access of the common communication channel, managing *collisions* which are unintended destructive overlapping transmission by multiple hosts, etc.

1.2.5 Queues in data communication networks

The total number of data links in such a vast network is very small in comparison with the number of host computers. In the case of a LAN, only one of the many host computer can successfully transmit data over the broadcast medium at any time. Therefore data communication over such an enormous and complicated network is required to be very efficient. Let us now understand some of the queuing that occurs in computer networks. A router receives data frames on incoming links, from another router. The network layer processes each packet very minimally and gives it to the DLL corresponding to another link over which the packet should be retransmitted. Following are some details. The DLL at the receiving router performs error detection and keeps track of whether or not all transmitted frames from the preceding router are received. The DLL strips the frame header and gives the packet to the network layer. The network layer examines the packet header. It determines the link over which the packet should be retransmitted (forwarded) towards the eventual destination. A few fields of the packet header, such as the number of hops may be updated and the packet is passed onto the DLL. The DLL introduces

- redundancy bits for error detection,
- serial number to track whether or not all the packets are successfully received by the router on the other side of the forward link, and

- frame boundary bits to determine the start and end of a frame.

The resulting data frame is transmitted on the forward link. The entire process at the router can be approximated to be a single FIFO queuing system. The real situation is a little more complicated. A router uses a more involved *data link protocol* over each of its links. As mentioned above, the activity includes using (a finite field) serial numbering of the data frames, acknowledgments, and retransmissions if necessary. Therefore, after transmitting a data frame, the router needs to hold it in another queue. It can be deleted only after the router receives an explicit or implicit acknowledgment from the frame receiving router. Thus, a better approximation uses two *interacting queues*.

A host computer connected to a common LAN maintains data frames for transmission in a queued buffer. When transmitted, a packet can collide with another, if a different host computer also starts transmitting a packet, in an overlapping time interval. Thus we have multiple queues with *interacting servers*, in a LAN.

1.3 Queuing Models

A model of a physical system is a mathematically precise representation of the interaction of several variables and functions governing the original system. The spirit behind the mathematical representation is two-fold as follows. We would like the representation to duplicate the functioning of the original system as closely as our knowledge of the system and our knowledge of mathematics allow us to do. We would also like the mathematical representation to be simple enough for us to analyze the same, with our limited knowledge of mathematics, and evaluate the required performance characteristics. Therefore, in most cases, these precise mathematical models are approximations of the real characteristics of the systems being modeled. These desirable features are often contradictory and therefore lead to multiple models with a simple model on the one hand and a more accurate but complicated one on the other, for the same physical system. A simple queuing model is a single FIFO queue. Such a model may be an adequate representation for a single database server and an acceptable approximate representation of a network router. Figure 1.1 shows a usual pictorial representation of a single FIFO queue. The circle at the right is the service area. At most one customer can be in the service area at any time. The server is required to be busy, serving, if there is at least one customer in the system. Customers are represented by short vertical lines. Waiting customers are in the buffer to the left of the service area. The mathematical behaviors of the *arrival time instants* and *service time* intervals for different customers are parts of the model. The arrival time instants are equivalently represented by *inter-arrival times* (IATs) and the time instant of the first arrival. The amount of time a customer spends in the entire system is called the *response time* which is the sum of the *waiting time* and the service time. Response time is also called sojourn time. Typical performance characteristics

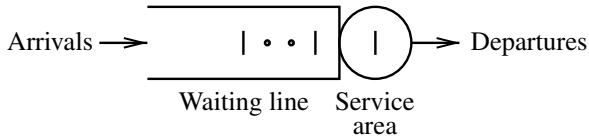


FIGURE 1.1: FIFO queue representation

of interest in such a simple queue include the following. The *average number of customers* found in the system. This is defined as follows. The number of customers in the system is a function of the continuous time variable. The average of this time varying function, over a long time interval, is the required performance figure. The average response time is the average of the response time intervals experienced by all the customers over the long time interval. The average waiting time and the average service time are similarly defined. The *fraction of time* the server is busy is also an important performance criterion. It corresponds to the total of the time intervals that the server is busy, divided by the total time of the queue operation. This fraction is known as the *utilization* of the server. The number of customer positions in a waiting line may be finite in some application systems. In such cases, a customer attempting to arrive is not allowed to wait in the waiting line. Such queues are known as finite buffer queues.

David George Kendall (1918–2007) introduced a notation to represent different classes of single waiting line queues in the year 1953. The $A/B/m/k/n$ queue has interarrival times of type A and service times of type B. The parameter m is the number of servers, k is the maximum number of customers allowed to be in the queue (including any being serviced) at any time, and n is the size of the population from which customers arrive. Classes of A and B are distinguished by their statistical properties.

The behavior of a queue is cumulative, in the sense that the number of customers found at any time instant is affected by previous activity. Clearly, the future behavior of the queue is affected by the the number of customers found at the current time instant. In general, the time instant of the next arrival may depend on the past, for example, on the time instant of the most recent arrival. Similarly, time instant at which the customer being currently served will depart may depend on when the time instant the previous customer departed after service. However, it turns out we can construct simple mathematical models of IATs and service times wherein the statistics of the future behavior of a queue depends only on the number of customers in the system at the present time instant and not even on the time instants of the most recent arrival and departure. These are developed in the chapters to follow.

Many complicated queuing systems can be modeled with the help of modifications

of simple models, or with interconnections of simple models or with both. Therefore, it is very important to study very simple models in the beginning, even if they appear to be unrealistically ideal. A study of a variety of simple models and some of their modifications and interconnections also helps us to develop more realistic models for physical systems. Such a study also enhances the level of our mathematical knowledge and helps us to attempt analysis of more realistic, complicated models. Occasionally, it turns out that some performance characteristics of a more involved

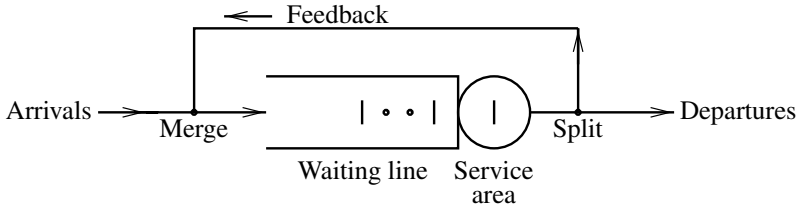


FIGURE 1.2: Round robin queue

model are the same as the corresponding ones for a simple model. For example, consider a round robin scheme, the model for which is obtained by using a feedback path in the simple FIFO model. A pictorial representation is shown in Figure 1.2. The wperating system's timer decides when to pause the service for a job and feed it back to the queue's tail. The time for feedback is usually negligible in comparison with each continuous service time intervals. Therefore, the number of customers in the FIFO and in the corresponding round robin models are identical, all the time. This implies that the two models have the same average number of customers and server utilization. The following describes a few examples of models for queues for different systems constructed by making modifications to simple models. A model for the queue for multiple servers in a DOS with a few computation intensive servers is shown in Figure 1.3. Job arrivals are those submitted by many client computers. They queue up for FIFO service. Each server has its own service area. There can be at most one customer in each service area. The DOS must use a scheduling policy on which server to send an arriving job to, if there are multiple servers free to serve, when an arrival comes in. In some client server systems, a client may be allowed to submit only one job to the server and is not allowed to submit another job until the previously submitted job is complete. In such a system, the arrivals are functions of the number of jobs in the servers' queue. In a more general system of multiple processors, jobs queue up in front of all servers. The DOS may ship jobs from the output of one queue to the tail of another or to the tail of the original

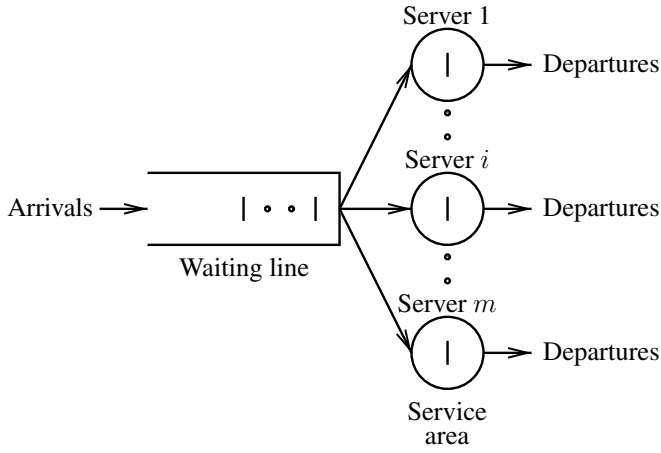


FIGURE 1.3: A queue with multiple servers

queues. At some time, possibly after visiting several queues multiple times, a job finally departs. Such a system is called an *open queuing network* and is depicted in Figure 1.4. Alternatively, in a DOS, we can model all the processes of the DOS as customers that move from one queue to another depending on the data received. External programs now function as data to the DOS. In such a case, the number of customers in the queuing network is a constant all the time. Such systems are called *closed queuing systems*. The model for a queuing system is not complete without a precise mathematical specification of the behavior of interarrival times and service times. The model may also require a scheduling policy for system operation. The interarrival times and service times are usually uncertain quantities; they vary from one job to another. But they also usually possess statistically steady behavior over a long time of operation. Therefore, we use probability theoretic models for these. In some cases, the scheduling policy can be varied to optimize some performance criterion of the system.

1.4 Conclusion

Many real computer networks' queuing models are very complicated. However, in many cases, approximate models can be developed with the help of either the variations of simple models or some interconnections of simple models. Examples

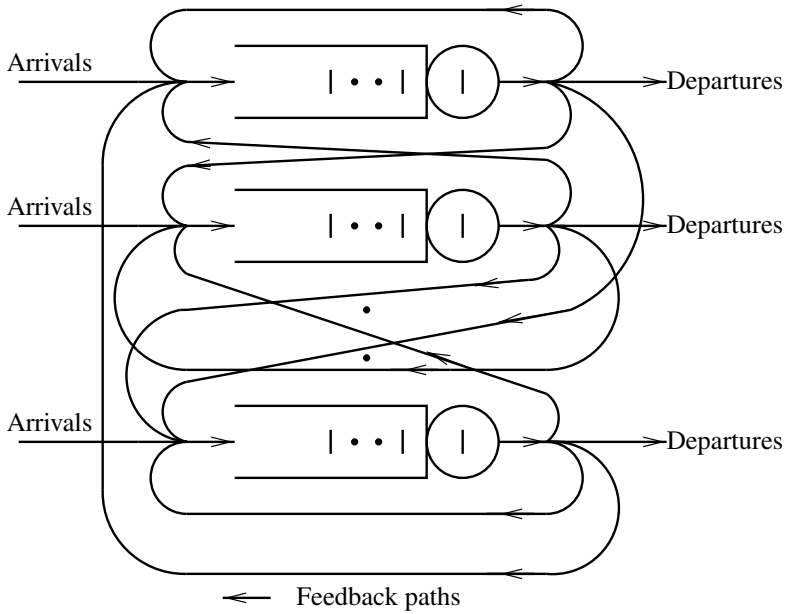


FIGURE 1.4: Open queuing network

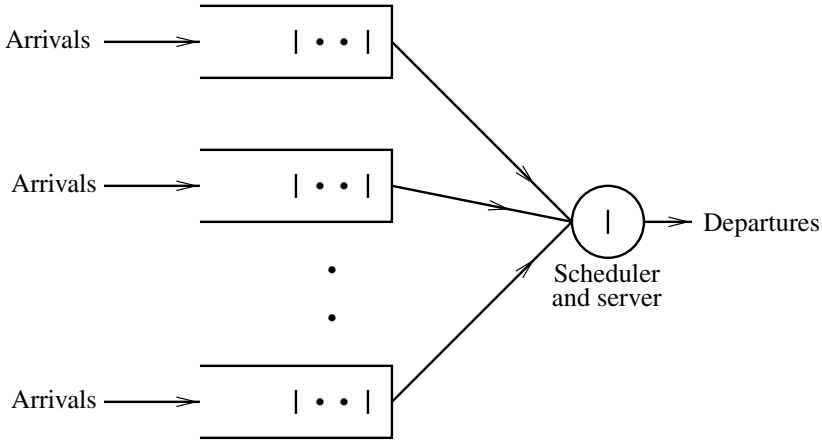


FIGURE 1.5: Multiple queues with a single scheduler and server

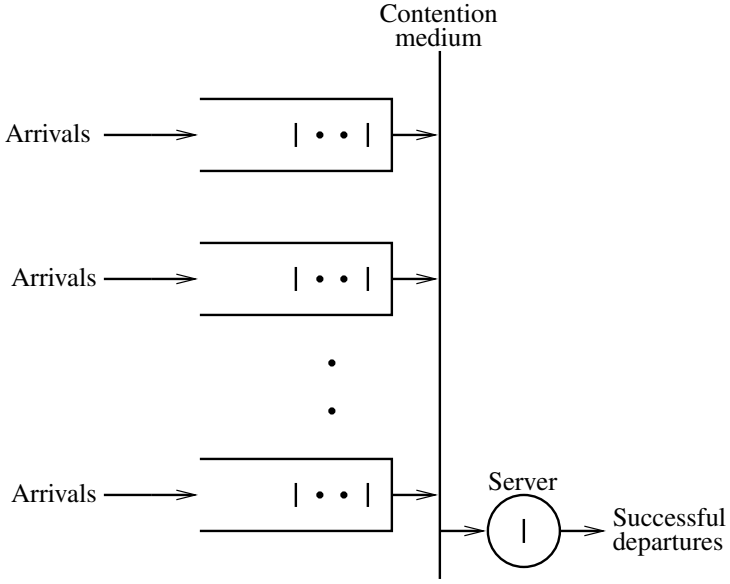


FIGURE 1.6: Multiple queues with contention based service

of these were included at the beginning of this chapter to motivate a detailed study of queuing models starting from very simple models. The next chapter deals with the introduction and detailed analysis of simple traffic models. Simulation of these traffic patterns is also a topic there. In addition, simple principles and procedures of parameter estimation are included. They are very useful in the analysis of real or simulated traffic patterns.

Many of computer networks' diverse performance metrics are statistical averages. therefore, by and large, analyses of queues are applications of probability theory and stochastic processes. These are functions of the behavior of time periods of internal activities and external load or request patterns. Typically, requests for service wait in queues. Therefore, queuing theoretic principles are the main set of tools in our performance analysis. Statistical averaging of the quantities affecting the performance requires the study of the variations of those quantities as they occur repeatedly. Evaluation of such statistical averages is facilitated by the extensive use of Probability Theory and Random Processes, in queuing theory. Many advanced principles of probability theory and elementary principles of random processes are easier to grasp with the help of the applications in which they find use. They are introduced and covered in the necessary detail, as needed, in the following chapters. A review of Probability Theory appears in the Appendix at the end of the book.

Chapter 2

Characterization of Data Traffic

2.1 Introduction

Data traffic is the sequence of movement of data items through a point or a physical device. A typical data item is a contiguous sequence of bits forming a data packet. When these data items pass through a physical device, there is usually some impediment in the form of reception, processing, and forwarding. Such an impediment results in queuing and causes time delays. In general, queues have successive arrivals of customers as inputs. These arrivals experience possible waiting and service before being output as successive departures. This chapter introduces important random variables that constitute models for arrival and service disciplines. The statistical nature of arrivals can be expressed in different ways. For example, if successive interarrival times (IATs) are independent, a specification of the initial condition in the form of the time instant at which the operation of the queue starts and the probability density function (pdf) of IATs are sufficient to completely describe the nature of arrivals. The Pareto random variable for IATs is one such model. This random variable exhibits some important variations in its characteristics, based on the values of the parameters of its pdf. Its variance can be finite or infinite. Infinite variance random variables find applications in characterizing bursty data traffic. Therefore Pareto random variables are studied in this chapter. Since its study is a valuable review of elements of probability theory, it is introduced first.

The number of arrivals over a time interval is another important way of characterizing the nature of arrivals. In general, this requires the specification of the initial condition and the time instants of the start and end of the interval over which the random variable number of arrivals is characterized. There is an important class of arrival disciplines for which this specification can be considerably simplified; the initial condition of the starting time and the exact time instants constituting the time interval over which the number of arrivals is being characterized are not important. The only important quantity influencing the number of arrivals is the amount of time in the time interval. This class of arrivals is known as Poisson arrivals, named in honor of Simeon Denis Poisson (1781–1840), a French scientist. The IATs in a stream of Poisson arrivals are independent and identically distributed (iid) exponential random variables. This class of random variables possess a very interesting property known as "memorylessness." The exponential random variable is a very useful model for service times since the memoryless property greatly simplifies the analysis

of queues. Poisson and exponential random variables are studied in detail, following a study of the Pareto random variable.

One of the practical problems encountered in data communication networks is the errors in received data packets. Errors are caused by noise in physical links. A simple model of noise and its effects on bit errors and data packet errors is introduced. A particular advantage of this model is that if the packet error rate at a particular data transmission rate is given, the corresponding packet error rate at a different data transmission rate can be evaluated. This helps in optimizing the data transmission rate.

The basic approach to simulation of a queue is to generate outcomes of random variables corresponding to data traffic and use them in the way the queue operates. Therefore, simulation of random variables corresponding to data traffic is fundamental to the simulation of queues. Computer simulation of random variables is most commonly implemented by attempting to repeatedly generate iid outcomes of a very simple random variable and subjecting them to the needed transformations. Unfortunately, computers execute algorithms in a deterministic way. Therefore, if a simulation algorithm is run repeatedly with identical external data input, it produces identical results for every run. There is nothing random about this. If the external inputs themselves form all of the extensive random data, we are not using the computer to simulate; we would only be using it to operate a system, possibly a queue, to which random data from elsewhere are input. The best we can hope to achieve is to use the computer to generate a long sequence of numbers that "appear" to have the properties of the outcome of a sequence of iid random variables. There are excellent algorithms for this purpose. Typically they approximate the generation of iid uniformly distributed random variables. The length of the sequence of such generated numbers is typically $2^k - 1$ where k is the number of bits in the computer word the the algorithm uses. If the algorithm is run to generate more than $2^k - 1$ random numbers, the sequence repeats. The algorithms also accept an external input called the seed that determines the starting point in the cyclic sequence of generated numbers. Thus, by giving different seeds, practically different simulation trials are realized.

The next step in simulation of queues is to generate outcomes of random variables for different data traffic models. This is usually accomplished by using mathematical transformations of a uniformly distributed random variable (that can be simulated) to the desired random variables. This is also a topic studied in this chapter.

Finally, analysis of simulation results require an understanding of the basic principles of parameter estimation from random samples. Only some very elementary principles of parameter estimation are included in this chapter.

2.2 The Pareto Random Variable

The Pareto random variable is named in honor of Vilfredo Federico Damaso Pareto (1848–1923), a French-Italian scientist. It is characterized by a pdf which varies as a negative power of the outcome and a value of zero for pdf for small values of the outcome. That is, if X is Pareto, its pdf

$$f_X(x) = \begin{cases} v\,x^{-u}, & x \geq w \\ 0, & x < w. \end{cases} \tag{2.1}$$

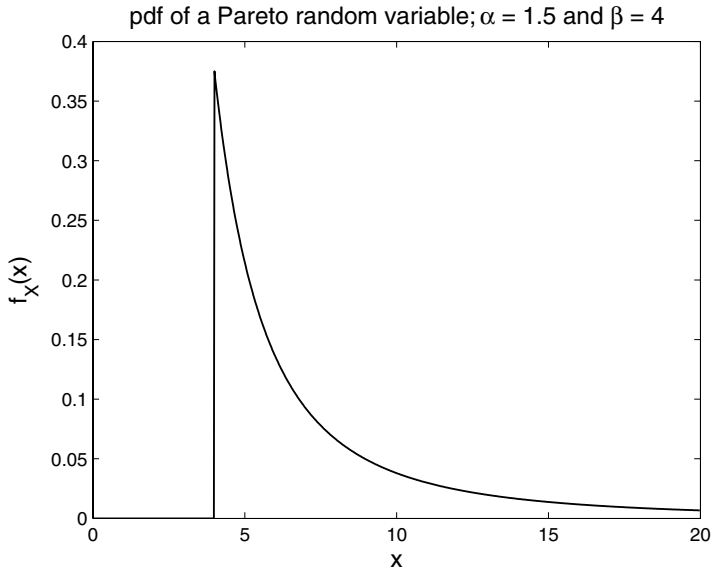


FIGURE 2.1: Density function of a Pareto random variable; $\alpha = 1.5$, $\beta = 4$

To make this a valid pdf, we need

$$\int_{-\infty}^{\infty} f_X(x)\, dx = 1. \tag{2.2}$$

Now,

$$\int_{-\infty}^{\infty} f_X(x)\, dx = v \int_{w}^{\infty} x^{-u}\, dx \tag{2.3}$$

$$= \frac{v}{1-u} \left[x^{-u+1}\right]_{w}^{\infty}. \tag{2.4}$$

We need $u > 1$ and $w > 0$ for this integral to be finite. Then,

$$\int_{-\infty}^{\infty} f_X(x)\, dx = \frac{v}{u-1}\, w^{-(u-1)} = 1. \tag{2.5}$$

Therefore,

$$v = (u-1)\, w^{u-1} \tag{2.6}$$

and

$$f_X(x) = (u-1)\, w^{u-1}\, x^{-u} \tag{2.7}$$

$$= \frac{u-1}{w} \left(\frac{w}{x}\right)^{u}. \tag{2.8}$$

We introduce new constants, $\alpha = u - 1 > 0$ and $\beta = w > 0$ in order to express the density function in a commonly represented form. We have

$$f_X(x) = \begin{cases} \frac{\alpha}{\beta} \left(\frac{\beta}{x}\right)^{\alpha+1}, & x \geq \beta \\ 0, & x < \beta. \end{cases} \tag{2.9}$$

An alternative common form of representation uses the Hurst parameter H instead of α. Harold Edwin Hurst (1880–1978) was a British hydrologist. He studied long term storage capacities of reservoirs based on empirical observations on the river Nile. The Hurst parameter for a Pareto random variable is given by

$$H = \frac{3-\alpha}{2}. \tag{2.10}$$

Let us evaluate the properties of the above valid density function. The cumulative distribution function (cdf) is

$$P[X \leq x] = \int_{\beta}^{x} f_X(x)dx, \quad x \geq \beta \tag{2.11}$$

$$= 1 - \left(\frac{\beta}{x}\right)^{x}, \quad x \geq \beta \tag{2.12}$$

$$= 0, \quad x < \beta. \tag{2.13}$$

The expectation

$$E[X] = \int_{-\infty}^{\infty} x\, f_X(x)\, dx \tag{2.14}$$

$$= \int_{\beta}^{\infty} \frac{\alpha}{\beta}\, x \left(\frac{\beta}{x}\right)^{\alpha+1} dx \tag{2.15}$$

$$= \alpha\, \beta^{\alpha} \int_{\beta}^{\infty} x^{-\alpha}\, dx \tag{2.16}$$

$$= \alpha\, \beta^{\alpha} \left[\frac{x^{-\alpha+1}}{-\alpha+1}\right]_{\beta}^{\infty}. \tag{2.17}$$

Now, α needs to be larger than 1 for finite $E[X]$. Therefore, for $\alpha > 1$

$$E[X] = \frac{\alpha\, \beta^{\alpha}}{\alpha - 1}\, \beta^{-\alpha+1}$$

and finally, we have

$$E[X] = \begin{cases} \frac{\alpha\, \beta}{\alpha - 1}, & \text{if } \alpha > 1 \\ \\ \infty, & \text{if } \alpha \leq 1. \end{cases} \tag{2.18}$$

The variance Pareto random variable is evaluated as

$$\text{var}[X] = \int_{-\infty}^{\infty} (x - E[X])^2 \, f_X(x) \, dx \tag{2.19}$$

$$= E[X^2] - E^2[X]. \tag{2.20}$$

Equation (2.20) follows by expanding the square in equation (2.19).

$$E[X^2] = \int_{-\infty}^{\infty} x^2 \, f_X(x) \, dx \tag{2.21}$$

$$= \alpha \, \beta^{\alpha} \int_{\beta}^{\infty} x^{-\alpha+1} \, dx \tag{2.22}$$

$$= \alpha \, \beta^{\alpha} \left[\frac{x^{-\alpha+2}}{-\alpha+2} \right]_{\beta}^{\infty}. \tag{2.23}$$

For $E[X^2]$ to be finite, we need $\alpha > 2$. If $\alpha > 2$,

$$E[X^2] = \frac{\alpha \beta^{\alpha} \beta^{-\alpha+2}}{\alpha - 2} = \frac{\alpha \beta^2}{\alpha - 2}. \tag{2.24}$$

Summary
The Pareto random variable X can have any physical dimension, such as length, mass, time, or bits (approximating number of bits by a real number). The parameter α is dimensionless, and β has the same dimension as X.

$$f_X(x) = \begin{cases} \frac{\alpha}{\beta} \left(\frac{\beta}{x} \right)^{\alpha+1}, \text{ if } x \geq \beta \\ 0, \qquad\qquad \text{if } x < \beta, \end{cases} \tag{2.25}$$

$$F_X(x) = \begin{cases} P[X \leq x] = 1 - \left(\frac{\beta}{x} \right)^{\alpha}, \text{ if } x \geq \beta \\ 0, \qquad\qquad\qquad\qquad \text{if } x < \beta, \end{cases} \tag{2.26}$$

$$E[X] = \begin{cases} \frac{\alpha\beta}{\alpha-1}, \text{ if } \alpha > 1 \\ \\ \infty, \quad \text{if } \alpha \leq 1, \end{cases} \tag{2.27}$$

$$\text{var}[X] = \begin{cases} \frac{\alpha\beta^2}{\alpha-2} - \left(\frac{\alpha\beta}{\alpha-1}\right)^2, \text{ if } \alpha > 2 \\ \\ \infty, \qquad\qquad\quad \text{if } \alpha \leq 2. \end{cases} \tag{2.28}$$

The range $\alpha \in (1, 2]$ is of interest to us. In this range, the mean is finite but the variance is infinity. Data traffic in present day LANs is very bursty, despite having an overall finite average value. Modeling interarrival times between successive data packets by a Pareto random variable with $\alpha \in (1, 2]$ is gaining popularity. It turns out that we can easily simulate Pareto random numbers, as discussed later in this Chapter.

Example 2.1

In an Ethernet, successful packets (those that are transmitted without collisions) appear as the presence or absence of a successful packet, over a time interval. An example of such a trace is shown in Figure 2.2. Extensive experiments with Ethernet traffic have led to a model in which the time intervals between the end of one packet and the beginning of the next are Pareto with $\alpha = 1.2$ as an estimate. Let the average of such OFF times in the data packet train be 1 millisecond (msec and ms are also used to denote millisecond). Find the minimum time interval between successive packets. Find $P[X > 10\,\text{msec}]$, i.e., the probability of finding no arrival in 10 msec since the end of the previous packet. □

Solution

$$E[X] = \frac{\alpha\beta}{\alpha - 1} \tag{2.29}$$

$$\beta = \frac{E[X](\alpha - 1)}{\alpha} = \frac{1\,\text{msec}\,(1.2 - 1)}{1.2} \tag{2.30}$$

$$= \frac{1}{6}\ \text{msec}. \tag{2.31}$$

Since $f_X(x) = 0$, for $x < \beta$, the OFF time is always $\frac{1}{6}$ msec or higher. Note the

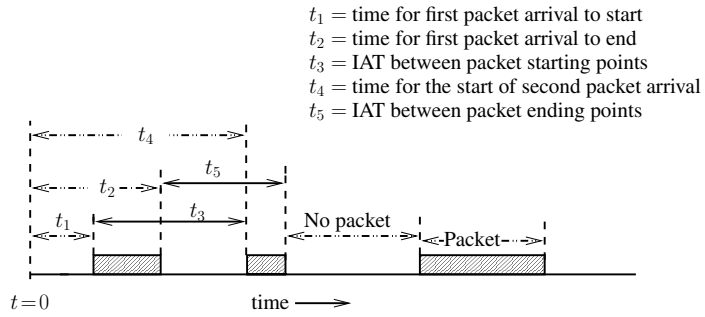


FIGURE 2.2: ON-OFF model of a packet train.

difference between two random variables associated with the stream of packets, the OFF times and the IATs.

$$P[X > 10\,\text{msec}] = 1 - F_X(10) \tag{2.32}$$

$$= 1 - \left[1 - \left(\frac{1}{60}\right)^{1.2}\right] \tag{2.33}$$

$$= \left(\frac{1}{60}\right)^{1.2} \approx 0.007. \tag{2.34}$$

The probability of OFF time to be larger than or equal to 10 times the mean is still not too small! This is the heavy-tailed property of this random variable. Later on, we will compare this with the probability of the same event for an exponential random variable with the same mean. □

Example 2.2

The probability density function of the time for the next bus arrival starting at 8 AM as zero time is Pareto with $\alpha = 1.7$ and $\beta = 1$. Time is measured in minutes. At 8:05 AM, the bus had not arrived. Determine the probability that the bus will not arrive for at least t more minutes after 8:05 AM. □

Solution
Starting from equation (2.13) for $P[X \leq x]$ of a Pareto random variable, we have

$$P[X > x] = \left(\frac{\beta}{x}\right)^{\alpha}, x > \beta. \tag{2.35}$$

Let T be the absolute arrival (random) time

$$P[T > 8:05 + t | T > 8:05] = \frac{P[T > 8:05 + t]}{P[T > 8:05]} = \frac{P[X > 5 + t]}{P[X > 5]} \tag{2.36}$$

$$= \left(\frac{5}{5+t}\right)^{1.7}. \tag{2.37}$$

□

Example 2.3
An agent in a train A is required to give a key to another agent in train B. It is known that Train B will be parked at a station S between 3:00 PM and 4:00 PM. Train A starts from a distant point at 1 PM the same day. Its travel time to reach station S is a Pareto random variable with $\alpha = 3$ and $\beta = 1$ hour. It will stop next to where train B would be in station S for a negligible amount of time and proceed. What is the probability that the hand-over of the key will be successful? Ignore the time for agents to walk to each other if and the two trains stop next to each other. □

Solution
Let X be the random variable of the time in hours it takes for train A to travel to station S. We need

$$P[2 < X < 3] = \int_{2}^{3} \frac{\alpha}{\beta} \left(\frac{\beta}{x}\right)^{\alpha+1} dx. \tag{2.38}$$

This evaluates to $\frac{1}{8} - \frac{1}{27} = \frac{19}{216} = 0.088$. □

Example 2.4
A Pareto random variable X has $\alpha = 1.5$ and $\beta = 2$. We would like to construct a new random variable for IATs in the form of the random variable $Y = X - a$, with a constant a such that Y is nonnegative but its density is nonzero starting from the outcome 0 itself. Determine a and completely specify the probability density function of Y, its mean and variance. □

Solution
If we draw a rough figure (or even imagine one) with the density function pushed so that it starts to be nonzero from the 0 point itself, we find that $a = 2$. Substitute $x = y + 2$ in the expression for the density function. The density of Y is zero for $y < 0$. More systematically,

$$P[y \leq Y < y + dy] = P[y + 2 \leq X < y + 2 + dx], \;\; y \geq 0 \;\; \text{and } dy = dx. \tag{2.39}$$

Therefore, $f_Y(y) = f_X(y + 2), \;\; -\infty < y < \infty$. That is,

$$f_Y(y) = 0.75 \left(\frac{2}{y+2} \right)^{2.5}, \;\; y \geq 0 \tag{2.40}$$

$$= 0, \;\; y < 0. \tag{2.41}$$

$$E[Y] = E[X] - 2 = \frac{1.5 \times 2}{1.5 - 1} - 2 = 4. \tag{2.42}$$

Variance of a random variable does not change with translation. Therefore,

$$\text{var}[Y] = \text{var}[X] = \infty. \tag{2.43}$$

□

2.3 The Poisson Random Variable

Let us study the traditional and "smooth" interarrival times model. The properties of this random variable can be formally derived by three simple and appealing assumptions. Consider an electron gun shooting out electrons in a narrow beam. This is a random phenomenon. Let us assume that the electrons' arrival times at a particular point follow the three randomness properties below.

1. In a narrow time interval, the probability of an arrival is proportional to the time interval.

2. In a narrow time interval, the probability of two or more arrivals is negligible in comparison with the probability of one arrival.

3. Numbers of arrivals in nonoverlapping time intervals are mutually independent of one another.

Note that $(0, t_1]$ and $(t_1, t_2]$ are nonoverlapping. As an example, if firing times of electrons are independent and statistically steady, then these assumptions are intuitively appealing. Mathematically, the assumptions imply the following.

1.

$$\lim_{\delta t \to 0} \frac{P[\text{ one arrival in } \delta t\,]}{\delta t} = \lambda, \text{ a constant} \tag{2.44}$$

2.

$$\lim_{\delta t \to 0} \frac{P[\text{ two or more arrivals in } \delta t\,]}{P[\text{ one arrival in } \delta t\,]} = 0 \tag{2.45}$$

3.

$$\begin{aligned} &P\left[k_1 \text{ arrivals in } (t_1, t_2] \text{ and } k_2 \text{ in } (t_2, t_3]\right] \\ &\quad = P\left[k_1 \text{ arrivals in } (t_1, t_2]\right] \cdot P\left[k_2 \text{ arrivals in } (t_2, t_3]\right]. \end{aligned} \tag{2.46}$$

From these three defining assumptions, we can derive the probability mass function (pmf), $P\left[k \text{ arrivals in } (0, T]\right]$. The pmf will be a function of only one parameter value λ, which is found in the defining assumptions (and the time interval T).

2.3.1 Derivation of the Poisson pmf

Consider a time interval $(0, T]$. Divide this interval into n equal parts. As n increases and tends to ∞, $\frac{T}{n} \to 0$ and we have a narrow sub-interval tending to 0. Therefore, in each such sub-interval, we have one arrival with probability $\frac{\lambda T}{n}$ and zero arrivals with probability $1 - \frac{\lambda T}{n}$. Two or more arrivals occur with zero probability. These arguments are accurate in the limit, as $n \to \infty$. The number k of sub-intervals with arrivals in a total of n sub-intervals is binomially distributed.

$$P[k \text{ arrivals in } (0, T]] = P[k \text{ in } T], \text{ for brevity} \tag{2.47}$$

$$= \lim_{n\to\infty} \frac{n!}{k!(n-k)!}\left(\frac{\lambda T}{n}\right)^k \left[1-\frac{\lambda T}{n}\right]^{n-k}. \quad (2.48)$$

We just need to evaluate the above limit.

$$P[k \text{ in } T] = \lim_{n\to\infty} \frac{(\lambda T)^k}{k!}\left[1-\frac{\lambda T}{n}\right]^{-k}\left[1-\frac{\lambda T}{n}\right]^{n} \frac{n!}{n^k(n-k)!}. \quad (2.49)$$

The quantity

$$\left[1-\frac{\lambda T}{n}\right]^{-k} \to 1 \text{ as } n \to \infty. \quad (2.50)$$

Therefore,

$$P[k \text{ in } T] = \lim_{n\to\infty} \frac{(\lambda T)^k}{k!}\left(1-\frac{\lambda T}{n}\right)^{n} \times \left[\frac{n\cdot(n-1)\cdot(n-2)\cdots(n-k+1)}{n^k}\right]. \quad (2.51)$$

In the last fraction, each $(n-i)$ in the numerator cancels with an n in the denominator, as $n \to \infty$, for any finite k. Therefore,

$$P[k \text{ in } T] = \frac{(\lambda T)^k}{k!} \lim_{n\to\infty}\left(1-\frac{\lambda T}{n}\right)^{n}. \quad (2.52)$$

Concentrate on

$$\lim_{n\to\infty}\left(1-\frac{\lambda T}{n}\right)^{n} = \lim_{n\to\infty}\left[\left(1-\frac{\lambda T}{n}\right)^{\frac{n}{-\lambda T}}\right]^{-\lambda T} \quad (2.53)$$

$$= \left[\lim_{a\to 0}(1+a)^{\frac{1}{a}}\right]^{-\lambda T} \quad (2.54)$$

$$= \left[e^{\left\{ \lim_{a \to 0} \frac{1}{a} \ln(1+a) \right\}} \right]^{-\lambda T}. \tag{2.55}$$

Consider

$$\lim_{a \to 0} \frac{\ln(1+a)}{a}.$$

Apply L'Hospital's rule. This rule is named in honor of Guillaume Francois Antoine de L'Hospital, a French mathematician (1661–1704).

$$\lim_{a \to 0} \frac{\ln(1+a)}{a} = \lim_{a \to 0} \frac{1}{(1+a) \cdot 1} \tag{2.56}$$

$$= 1. \tag{2.57}$$

Therefore,

$$\exp\left[\{ \lim_{a \to 0} (1+a)^{\frac{1}{a}} \} \right] = e \tag{2.58}$$

and,

$$\lim_{n \to \infty} \left[1 - \frac{\lambda T}{n} \right]^n = e^{-\lambda T}. \tag{2.59}$$

Finally,

$$P[k \text{ in } T] = \frac{(\lambda T)^k}{k!} e^{-\lambda T}. \tag{2.60}$$

This is the Poisson pmf. This pmf gives the probabilities of finding various numbers of possible arrivals in a given time interval, if the arrival scheme satisfies the previously mentioned three properties.

2.3.2 Interarrival times in a Poisson sequence of arrivals

Let X be the random variable corresponding to the time for the next arrival, soon after one arrival. Such a random variable is appropriately called the interarrival time.

$$P[X > t] = P[\text{no arrivals in } (0, t]] \tag{2.61}$$