Peter J. B. King

Computer and Communication Systems Performance Modelling

PRENTICE HALL INTERNATIONAL SERIES IN COMPUTER SCIENCE

C. A. R. HOARE SERIES EDITOR

Computer and Communication Systems Performance Modelling

Peter J. B. King

Prentice Hall
New York London Toronto Sydney Tokyo Singapore

First published 1990 by
Prentice Hall International (UK) Ltd
66 Wood Lane End, Hemel Hempstead
Hertfordshire HP2 4RG
A division of
Simon & Schuster International Group

Printed and bound in Great Britain
at the University Press, Cambridge.

Library of Congress Cataloging-in-Publication Data

King, Peter J.B., 1952—
Computer and communication systems performance modelling/Peter J.B. King.
p. cm.
ISBN 0-13-162984-0: $67.95
1. Electronic digital computers — Evaluation.
2. Telecommuncation systems — Evaluation. I. Title.
QA76.9.E94K56 1989 89-28401
004.2-dc20 CIP

British Library Cataloguing in Publication Data

King, Peter J.B., 1952—
Computer and communcation systems performance modelling.
1. Computer systems. Networks, Queues. I. Title
004.6

ISBN 0-13-162984-0
ISBN 0-13-163065-2 pbk

1 2 3 4 5 94 93 92 91 90

Computer and Communication Systems Performance Modelling

C.A.R. Hoare, Series Editor

BACKHOUSE, R.C., *Program Construction and Verification*
BACKHOUSE, R.C., *Syntax of Programming Languages: Theory and practice*
DEBAKKER, J.W., *Mathematical Theory of Program Correctness*
BARR, M. and WELLS, C., *Category Theory for Computing Science*
BEN-ARI, M., *Principles of Concurrent Programming*
BIRD, R. and WADLER, P., *Introduction to Functional Programming*
BJÖRNER, D. and JONES, C.B., *Formal Specification and Software Development*
BORNAT, R., *Programming from First Principles*
BUSTARD, D., ELDER, J. and WELSH, J., *Concurrent Program Structures*
CLARK, K.L. and McCABE, F.G., *micro-Prolog: Programming in logic*
CROOKES, D., *Introduction to Programming in Prolog*
DROMEY, R.G., *How to Solve it by Computer*
DUNCAN, F., *Microprocessor Programming and Software Development*
ELDER, J., *Construction of Data Processing Software*
ELLIOTT, R.J. and HOARE, C.A.R., (eds.) *Scientific Applications of Multiprocessors*
GOLDSCHLAGER, L. and LISTER, A., *Computer Science: A modern introduction (2nd edn)*
GORDON, M.J.C., *Programming Language Theory and its Implementation*
HAYES, I. (ed). *Specification Case Studies*
HEHNER, E.C.R., *The Logic of Programming*
HENDERSON, P., *Functional Programming: Application and implementation*
HOARE, C.A.R., *Communicating Sequential Processes*
HOARE, C.A.R., and JONES, C.B. (ed), *Essays in Computing Science*
HOARE, C.A.R., and SHEPHERDSON, J.C. (eds). *Mathematical Logic and Programming Languages*
HUGHES, J.G., *Database Technology: A software engineering approach*
INMOS LTD. *occam 2 Reference Manual*
JACKSON, M.A., *System Development*
JOHNSTON. H., *Learning to Program*
JONES, C.B., *Systematic Software Development using VDM (2nd edn)*
JONES, C.B. and SHAW, R.C.F. (eds), *Case Studies in Systematic Software Development*
JONES, G., *Programming in occam*
JONES, G. and GOLDSMITH, M., *Programming in occam 2*
JOSEPH, M., PRASAD, V.R. and NATARAJAN, N., *Multiprocessor Operating System*
LEW, A., *Computer Science: A mathematical introduction*
MACCALLUM, I., *UCSD Pascal for the IBM PC*
MARTIN, J.J., *Data Types and Data Structures*
MEYER, B., *Introduction to the Theory of Programming Languages*
MEYER, B., *Object-oriented Software Construction*
MILNER, R., *Communication and Concurrency*
MORGAN, C., *Programming from Specification*
PEYTON JONES, S.L., *The Implementation of Functional Programming Languages*
POMBERGER, G., *Software Engineering and Modula-2*
REYNOLDS, J.C., *The Craft of Programming*
RYDEHEARD, D.E. and BURSTALL, R.M., *Computational Category Theory*
SLOMAN, M. and KRAMER, J., *Distributed Systems and Computer Networks*
SPIVEY, J.M., *The Z Notation: A reference manual*
TENNENT, R.D., *Principles of Programming Languages*
WATT, D.A., *Programming Language Concepts and Paradigms*
WATT, D.A., WICHMANN, B.A. and FINDLAY, W., *ADA: Language and methodology*
WELSH, J. and ELDER, J., *Introduction to Modula-2*
WELSH, J. and ELDER, J., *Introduction to Pascal (3rd edn)*
WELSH, J., ELDER, J. and BUSTARD, D., *Sequential Program Structures*
WELSH, J. and HAY, A., *A Model Implementation of Standard Pascal*
WELSH, J. and McKEAG, M., *Structured System Programming*
WIKSTRÖM, Å., *Functional Programming using Standard ML*

For Rosemary

Preface

Outline of book

This book is an introductory text in the field of quantitative analysis of computer and communication system performance. Knowledge of elementary probability theory and simple stochastic processes is used, but the material is also briefly outlined in the early chapters. Although no knowledge of queueing theory is assumed, a number of aspects which only receive passing mention in other books will make this a useful reference book for the professional as well as a student text.

The first chapter introduces Kendall's (extended) notation for describing queueing systems, and sample path arguments that lead to Little's result, which relates queue lengths to arrival rates and waiting times. Various queueing disciplines are discussed qualitatively. The elements of probability theory are reviewed in the second chapter. This should be an element of revision for the student, since it is by no means a full introduction to the theory but covers all that is needed for this book. Chapter 3 gives a brief introduction to the theory of stochastic processes, in particular Markov processes, which play a central role in the analyses presented in later chapters. Little's theorem is proved more formally.

The body of the book starts in chapter 4, with simple queues. $M/M/1$ queues are examined in detail, starting with the time dependent formulation, and then leading on to the steady state solution. Generalisations such as the machine-repairman problem and the finite source queue are considered. Simple discrete time queues are also considered. The next chapter treats queues in which the service times can be generally distributed, and three different approaches are used to derive the Pollaczek-Khintchine formula. The following chapter treats systems which are unreliable, and have servers that break down and can be repaired. This study involves non-trivial manipulations of the generating functions and is perhaps the simplest system in which such calculations arise naturally. Priority queues, in which some jobs have priority over others form the subject of chapter 7. Both preemptive and non-preemptive priorities are considered. Kleinrock's conservation law, which provides a measure of the cost of different priority assignments, is proved. The next chapter uses the Laplace transform to evaluate busy period distributions. These non-trivial manipulations of Laplace transforms also allow waiting time distributions to be found for several different scheduling disciplines.

Chapter 9 considers multiserver Markovian queues. The analysis of $M/M/1$ queues is extended to $M/M/c$ queues, and taken to the limit as a study of $M/M/\infty$ queues. A number of two server systems with unusual characteristics are analysed.

Systems in which the two servers have different speeds are studied, and also systems in which the second server becomes available only after a delay. Multiserver priority systems are analysed.

Chapter 10 looks at networks of queues. The effect of connecting the output of one queue as the input to another is investigated. The ideas of local balance and time reversibility allow a large class of networks to be analysed. Solutions to networks in this class have the property known as *product form*, since the steady state probability of a network state is the product of the steady state probabilities of the nodes in the network considered in isolation. The following chapter examines some of the elegant algorithms for computing various performance metrics for networks of the product form class. We examine some of these and discuss their limitations. Chapter 12 treats approximation techniques based on product form solutions, and derives a number of bounds on throughput and delay which can be calculated more cheaply than full solutions.

Numerical methods for the solution of queueing problems form the subject of the next chapter. These include Gaussian elimination on the balance equations, eigen-analysis of the transition matrix, and Courtois type decomposition methods. Finally, a chapter on local area network performance is included.

Most chapters contain exercises which may aid understanding. Each chapter has a bibliography, giving the sources for the techniques described, and suggestions for further reading. I have endeavoured to reference only widely available journals and books, rather than technical reports, even when these are in fact the original publication.

The book is designed to be suitable for final year undergraduates in computer sciences, or conversion course MSc students. It will also find use in electrical engineering courses on communication systems analysis, and in mathematics departments as a source of examples in the field of operations research.

Prerequisites for the study of this book are an introductory course in probability theory, such as is given to most science and engineering students. An understanding of both discrete and continuous distributions is required, along with the elementary properties of generating functions, distribution and density functions, Laplace transforms, and moments. The necessary material is briefly covered in chapters 2 and 3.

Acknowledgments

This book would not have been possible without the support and encouragement of many people. I have been privileged to work closely with Isi Mitrani for many years, and he taught me most of what I know in this subject area. He will recognise some of our joint work appearing here in book form. Nachum Shacham will also

see his influence on many pages. The Department of Computer Science at Heriot-Watt University provided a working environment that encouraged me to write the book and allowed the class testing of some portions of the material. The computer facilities were also provided by Heriot-Watt University. The major portion of the book was typed by the author, although first drafts of chapters 5 and 8 were typed by Mrs Shirley Storer. Leslie Lamport's LaTeXsystem was used to typeset the book, with the final camera-ready copy being produced at the University of London Computer Centre on the Linotronic L-300 phototypesetter. Helen Martin from Prentice-Hall attempted to keep me on schedule and did not despair when deadline after deadline was passed.

The book was read at various stages by Ilze Ziedins, Stan Zachary, Peter Thanisch, Rob Pooley, Jane Hillston, and Saqer Abdel-Rahim. They clarified many points, and corrected errors. Their suggestions, and those of the publisher's anonymous reviewers, have been invaluable. Despite their efforts, errors undoubtedly remain, for which the responsibility rests with the author.

Finally, I must thank my wife, Rosemary, and children, Robert and Anna, for putting up with many husbandless and fatherless evenings and weekends when I was writing. Without their tolerance and encouragement this book would not have been finished.

Edinburgh PJBK

Contents

Chapter 1

Introduction

Computer and communication systems are expensive resources, and it is important that the most efficient use is made of them. Although the necessity to keep the CPU active as much as possible is perhaps less important today, the increasing use of sophisticated communications systems has led to many demands to share scarce resources. It is important to be able to quantify the performance of systems both existing and at the design stage, so that cost/benefit type of analyses can be used to choose between design alternatives.

This book is an introduction to the performance modelling of computer and communication systems. We shall be examining such questions as:

1. How much buffer space should be provided for users' input?
2. Will adding a second CPU to the system be more effective than upgrading the current CPU to a higher speed one?
3. What response time can we expect to simple queries on the database?
4. How many processes will be waiting for execution on average?
5. What retransmission policy will optimise throughput in a CSMA local area network?

There are three approaches to this type of problem. First, the system can be built, or modified, and then measured. If the measurement is performed while a standard set of tasks is running, this technique is known as *benchmarking*. This has the advantage that performance estimate is perfectly accurate. No errors have been introduced that are not there in the real system. The disadvantages of benchmarking are the cost and inflexibility of the technique. It will be expensive to acquire new equipment just for the purposes of evaluation. Even if it can be borrowed, there will be expense involved in configuring software to use the new equipment. Each benchmark will usually run for a fairly long period in order that its measurements are statistically reliable. If there are a large number of design alternatives, it will be very time consuming to investigate them all.

The second approach is to build a simulation model of the system. This model can be validated against the existing system and then altered to reflect the proposed modifications. In common with benchmarking, simulation is an experimental science; parameters are changed and then the benchmark is performed again or the

simulation model is rerun and the performance measured. Simulation models have the property that arbitrary levels of detail can be included. This is both an advantage and a disadvantage. The advantage is that any system feature can be included, to see if that feature has an impact on system performance. The major disadvantage is that there is a great temptation to include many features of the real system in the simulation model, which while adding to realism, also add to the running time of the model and hence the cost of the performance study.

This book deals only with the third approach to performance evaluation and prediction. A mathematical model of the system, or of parts of it, is constructed. This model can then be validated in the same way as a simulation model. After validation, changes to the system can be evaluated by changing appropriate model parameters and reevaluating the model solution. The expense here is in the time of the skilled modeller to construct a model which can both be solved and capture the significant behaviour of the system.

Although computers are completely deterministic machines, our analyses will use the theory of probability extensively. In fact, computer performance modelling has been one of the driving forces behind many recent advances in applied probability theory, particularly in the field of queueing theory. There are two reasons why we use probability theory so extensively. First, many of the processes involved are inherently random. For example, the time that a user takes to type a line of input will vary dramatically depending on their skill as a typist, the complexity of response expected, and many other factors. Similarly, the successful transmission of a packet in a computer network is a random event since there is a chance that there will be some interference with the communication, and it will need to be retransmitted. Secondly, it is more convenient. Even if we had complete knowledge of all the times involved at each stage of each process, the size of the data needed as input to the model would be extremely large, and, perhaps more importantly, the results and conclusions that we could draw would be needlessly specific. By representing the essential characteristics of the system with a few parameters we can evaluate the performance and hope to demonstrate the effects of changes in the parameter values.

1.1 History

The application of probability theory to these problems goes back to 1917 and the Danish engineer A.K. Erlang, who analysed the behaviour of simple queues to assist him in designing telephone exchanges. This telephonic flavour continued until the mid 1960s when the first analysis of a computer time-sharing system was undertaken by Scherr.

Many of the problems that we shall investigate are approached using some aspects of the theory of queues. This theory has been developed over the last seventy

years, and extensively so in the last twenty years. The literature is vast, with comprehensive bibliographies having over nine hundred entries in the early 1960s. We shall only touch on some of the simpler aspects of the theory. Over the last twenty years the field has blossomed, and now has several journals devoted exclusively to research papers on the subject.

1.2 Performance measures

There are many reasons why we may wish to calculate the performance of our systems, but for the most part long run costs and revenues are the interests of management. Freak occurrences that happen only in peculiar circumstances are of little interest. We will be examining the long run performance of systems. The notion of *statistical equilibrium* or *steady state* will often be used. This does not necessarily mean that the system behaves in a particularly 'steady' manner, but that the probabilities of the system being in a particular state have settled down and are not changing with time.

When a user submits a job to a system, the question that usually needs to be answered is 'How long before I have my answers?' Customer waiting time and its distribution is one of our main interests. Sometimes, particularly when priority queues are under consideration, we shall want the waiting time conditioned on the service requirement of the customer. The management of the system will be interested in the number of jobs waiting – since they must be stored somewhere – the throughput of the system, how many jobs it can process in unit time, indicates the rate at which revenue will be generated if a charge is made to each job using the system. The utilisation of the system, the proportion of the time that the system is idle, gives some indication of the scope for increasing throughput and hence revenue, without increasing the resources committed. Note that an increase in throughput may only be possible at the expense of increased delay to customers. The distribution of the lengths of idle and busy periods may also be of interest to the management.

1.3 Notation

A queueing system can be described in terms of six component parts.

1. The arrival process: a stochastic process describing how jobs arrive in the system from the outside world.
2. The service process: a stochastic process describing the length of time that a server will be occupied by a job.
3. The number of servers and their rates of service.