Measurement and Tuning of Computer Systems

DOMENICO FERRARI
GIUSEPPE SERAZZI
ALESSANDRO ZEIGNER

MEASUREMENT AND TUNING OF COMPUTER SYSTEMS

Domenico Ferrari

Computer Science Division
Department of Electrical Engineering
and Computer Sciences
University of California, Berkeley

Giuseppe Serazzi

Istituto di Analisi Numerica del CNR
Istituto di Matematica
Universitá di Pavia, Italy

Alessandro Zeigner

Prentice-Hall, Inc. *Englewood Cliffs, NJ 07632*

Library of Congress Cataloging in Publication Data

FERRARI, DOMENICO, (date)
Measurement and tuning of computer systems.

Rev. of: Le prestazioni degli elaboratori elettronici.
Includes bibliographies and index.
1. Electronic digital computers—Evaluation.
I. Serazzi, Giuseppe. II. Zeigner, Alessandro.
III. Title.
QA76.9.E94F4813 1983 001.64 82-20432
ISBN 0-13-568519-2

Editorial/production supervision
and interior design: **Kathryn Gollin Marshak**
Cover design: **Diane Saxe**
Manufacturing buyer: **Gordon Osbourne**

Cover art: © 1981 IEEE. Reprinted, with permission,
from COMPUTER, vol. 5, no. 4, July/August 1972.

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-568519-2

Prentice-Hall International, Inc., *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Prentice-Hall of Southeast Asia Pte. Ltd., *Singapore*
Whitehall Books Limited, *Wellington, New Zealand*

PREFACE

Even though it can hardly be viewed as a settled discipline, system performance evaluation has made impressive progress during its relatively brief existence to date. A number of advances have been made in recent times in computer system modeling, especially in analytic techniques, as well as in workload characterization and forecasting, capacity planning, configuration design, and, to a lesser extent, in measurement techniques and tools, tuning procedures, and procurement techniques.

A number of books have appeared in the last few years covering performance analysis and evaluation topics. Most of these books, however, have been written for researchers and students rather than for the practitioners in the field or for their managers. Some other books, intended for the professionals, do not adequately cover some of the areas in which the most important advances have occurred, for instance that of analytic modeling.

Subjects that the informed practitioner could afford to ignore until a few years ago, as nothing really useful in coping with everyday problems could be extracted from them, have become important in practice, and their knowledge will soon be viewed as indispensable for technical survival in the world of performance evaluation. Like many other types of computer professionals, the successful evaluator can only remain successful by keeping abreast of the continuous developments not only of hardware and software technology, but also of performance measurement and modeling technology.

This book was motivated by the preceding considerations. It was felt that a simple, easily readable, practically oriented introduction to the performance evaluation field which would either cover or prepare the reader to understand even the most recent important advances in the area was still missing. It was also felt that students, perhaps even researchers, would benefit from a more pragmatic treatment of the subject, based on

practical experience as well as on solid conceptual foundations. Such a book could not adequately deal with all the subjects that are now considered part of the discipline. We chose to emphasize measurement techniques (which we believe are the really fundamental ones: without measurement, system evaluation is impossible) and tuning projects. The context of most of the discussions and of the case studies is one in which the system whose performance is to be evaluated exists and is running; in this context, one of the main goals of evaluation activities is to improve the cost–performance ratio of the installation.

Chapter 1 introduces the basic concepts and problems, and presents a wide-ranging discussion of the fundamental performance indexes used in improvement studies. Our emphasis on measurement is reflected in the discussion of workload characterization in Chapter 2, most of which is oriented toward the construction and validation of executable workload models to be used in performance measurement experiments. The chapter also presents some recent results and research trends in the area.

The basic techniques for measuring computer systems or their simulators are studied in Chapter 3, while Chapter 4 discusses the most effective ways to represent measurement results for ease of interpretation, and Chapter 5 the predominant types of measurement tools and their characteristics. The application of techniques and tools to the problem of tuning an installation is described in general methodological terms in Chapter 6. A number of case studies encompassing a wide variety of systems, configurations, and application environments are presented in Chapter 7. Chapter 8 illustrates the particular methods and instruments that can be used to improve the performance of programs, an important problem for programmers as well as for system tuners; since systems go out of tune because of changes in their workload, workload improvement is often a very effective way of improving their performance.

Analytic models have an increasingly important role in tuning studies: they can provide substantial help in diagnosing certain performance problems and can predict the impact of an expensive or risky modification to the system before the change is actually implemented. Both operational analysis and queuing modeling techniques are discussed in Chapter 9, whose several case studies illustrate the various uses of models in performance improvement projects. Some of the cases show how a modeling approach can be taken in tackling problems that in Chapter 7 were dealt with only by empirical techniques.

The final chapter, Chapter 10, briefly addresses an important question, which is very seldom considered in technical discussions of performance evaluation: how beneficial is performance improvement? The question is a relevant one since tuning studies are not always advantageous. First, not all tuning efforts are successful. Second, even when such efforts are technically successful, their results may not be justified by their costs. A performance improvement study should be considered as an investment and evaluated on the basis of a cost–benefit analysis. Several case studies illustrating how this analysis can be performed accompany the discussion of the general principles and methods.

Each section of the book is identified by either two or three digits. The sections with two digits introduce a subject in an often qualitative form, and are oriented toward those readers who want to gain a general understanding of the problems and of their solutions. Three-digit sections provide more in-depth and more detailed treatment of the subject, or discuss specific real-world examples of the application of previously described techniques.

The authors are grateful to Dr. G. C. Baldovini, Chief Executive Officer of Syntax S.p.A., for his encouragement and support during the writing of the book and the preparation of the manuscript. The support of Professor F. Filippazzi of Honeywell Information Systems of Italy is also gratefully acknowledged. This book was conceived, and most of it was written, while D. Ferrari was Visiting Professor at the University of Pavia, with the support of the Regents of the University of California and of the National Science Foundation under Grants MCS78-24618 and MCS80-12900. Professors I. De Lotto of the Istituto di Informatica e Sistemistica of the University of Pavia and E. Magenes of the Istituto di Analisi Numerica of the same university made that visit possible and provided an appropriate habitat for the growth of this book. Two trips to Berkeley made by G. Serazzi to complete the first draft were supported by the Consiglio Nazionale delle Ricerche under Contracts 79.02332.62 and 80.02334.01, which provides funding for the joint Berkeley–Pavia NSF–CNR performance evaluation research project. For his financial and moral support, the authors are indebted to the contract's principal investigator, Professor M. Italiani.

This book is dedicated to the authors' wives and children in recognition of the countless hours of their husbands' and fathers' time it subtracted from them.

Domenico Ferrari
Giuseppe Serazzi
Alessandro Zeigner

CONTENTS

1

PROBLEM DEFINITION

1.1 DEFINITIONS AND BASIC CONCEPTS

The term *performance* refers to services provided by people or machinery to whoever requires them. An *information-processing system* is a set of hardware and software components capable of processing data according to user-written programs. Thus the term *performance*, referred to an information-processing system, indicates all the facilities that the system is capable of providing for its users. These facilities include the programming languages that are used to communicate with the system, the tools that it offers for the design and development of the programs, the processing and fault recovery features, the level of data security provided, and so on.

However, the term performance will be used here with a more restricted meaning, the one that is used when considering other engineering systems, like automobiles. Although among car performance indexes one ought to consider a car's ease of use, comfort, and stability (and one usually does, even though the relative weights given to them depend on the buyer's requirements), the term usually refers to the car's maximum speed when fully loaded, to the time it takes to reach a given speed, to its fuel consumption, and to other quantities that somehow express the efficiency with which the car carries out its functions.

As for cars, the choice of a computer system depends on many factors, one of which is the performance required. The weight of this factor varies with the person and the type of application. Designers, buyers, installation managers, programmers, occasional users, and maintenance engineers usually consider differently the same system because they have different requirements and aims. This also applies to its performance, represented by indexes that are given different weights by different people or even by

the same person in different circumstances. Each index is quantifiable and can therefore be an object of *evaluation*. A performance index can be evaluated in various ways: it can be measured, calculated, or estimated. These evaluations will always be quantitative. But it must be remembered that many of the factors considered when a system is to be selected are really of a qualitative nature and thus difficult to quantify.

This book covers the most common techniques for the quantitative evaluation of performance, which is an important component of a system's value (in the economic sense). Performance acquires great importance once a computer system has been chosen and installed. It is always convenient to keep the processing efficiency under control since the performance depends on both the type of load and the usage modes of the system, and can often be kept at acceptable levels with relatively simple types of intervention. If this results in performance values inferior to the ones required, or if the possibility of improving the system's performance seems to exist anyway, a careful analysis of the causes of inefficiency and of the best remedies to it must be carried out. This analysis will make use of the performance evaluation tools to be described in the sequel.

1.2 EVALUATION OBJECTIVES

The reasons that make performance evaluation worthwhile have already been briefly outlined. This section discusses them in more detail. All engineering systems are subject to performance evaluation. During the design, assembly, sale, and usage phases of a system, a system's performance is evaluated by various people with different aims and viewpoints to verify that the system satisfies given efficiency requirements and can be used for a certain purpose.

This rule also applies, or should apply, to computer systems. Clearly, verification that the requirements are satisfied becomes more important with expensive systems and with more critical performance requirements. Thus, the amount of resources to be invested in the evaluation of a minicomputer will be much smaller than the amount invested in a study for the performance improvement of a large machine or in the choice of a microprocessor that will be used in a certain product to be sold in large quantities.

The evaluation of a computer system's performance is necessary not only during the system's productive life but also during its design, when it is selected by a customer, and when it is installed. The applications of evaluation techniques can be classified in four main categories:

1. *Procurement:* All the evaluation problems concerning the choice of a system, or of system components, among various existing and available alternatives belong to this class, for example, installation design and hardware and software acquisition.
2. *Improvement:* This class includes all performance problems that arise in existing, running systems. They will be discussed in detail in this book.
3. *Capacity planning:* This is the class of the problems related to the prediction of when in the future the capacity of an existing system will become sufficient to process the installation's workload with a given level of performance.
4. *Design:* All problems that designers must face during the creation of a new system belong to this class.

Evaluation techniques can be used for problems in all these classes [FE78], [KO78]. Nevertheless, we shall almost exclusively refer to improvement studies in the sequel.

Thus, in the context of this book, the improvement of a system's efficiency will be the main object of evaluation studies. This improvement usually causes effects that are very important, but difficult to measure. For example, an increase of user happiness often entails an increase in programmer productivity, faster development of new applications, and an increased number of users. However, some consequences are easier to measure in economic terms, such as:

1. Reduction of the system's daily operating time, with the possibility of reducing the number or the work schedules of the operators.
2. Abolition or reduction of backlogs caused by peaks of load.
3. Decrease in the cost of the system's configuration (for example, due to the elimination of a superfluous channel or peripheral unit).
4. Postponement of the time at which workload increases will saturate the system, forcing its expansion or replacement.

Clearly, the real purpose of these studies is the improvement of the cost–performance ratio, rather than mere performance improvement. Cost reduction is a worthwhile result of a study as long as it is not accompanied by too large a decrease in performance. Economic aspects are always of primary importance, even if in what follows we shall almost exclusively deal with the technical aspects, that is, with the system's performance.

Together with the advantages to be derived from an evaluation study, one should always consider its cost. Before undertaking a study, both its benefits and its costs are unknown. It is therefore necessary to estimate these benefits and costs even if this is often difficult. The maximum gain that can be expected in terms of the cost–performance ratio depends on how far the system is from its optimal operating conditions. However, since the notion of optimum is vague in this case, and since this distance is very often unknown, the previous statement is not very useful. Thus, only partial estimates can usually be made, perhaps concerning the activity predicted for the near future.

The typical procedure for attacking the problem of improving the cost–performance ratio is, as shown in Fig. 1.1, an iterative procedure whose cycle contains a phase of *diagnosis* and one of *therapy*. The goal of the diagnostic phase is to establish the causes of the unsatisfactory cost–performance ratio. These causes are often identified with *bottlenecks*. In this case, a therapy will be effective if it is capable of eliminating or "widening" the bottlenecks that limit system performance. Obviously, if more than one therapy capable of ensuring the required improvements exists, the one with the lowest cost–improvement ratio will be chosen. As some therapies are very expensive (for example, those requiring an expansion of the configuration, or delicate operating system modifications, or workload changes), it is convenient to try to predict the effects of these therapies on the cost–performance ratio before applying them.

Some of the techniques that can be used in the diagnostic phase can also be used for performance prediction. The diagnosis–therapy cycle of the procedure outlined must sometimes be repeated because some bottlenecks show up once others have been eliminated. Nevertheless, the improvements obtained from these iterations are usually smaller and smaller, so it is often better to limit their number to only a few. By making use of

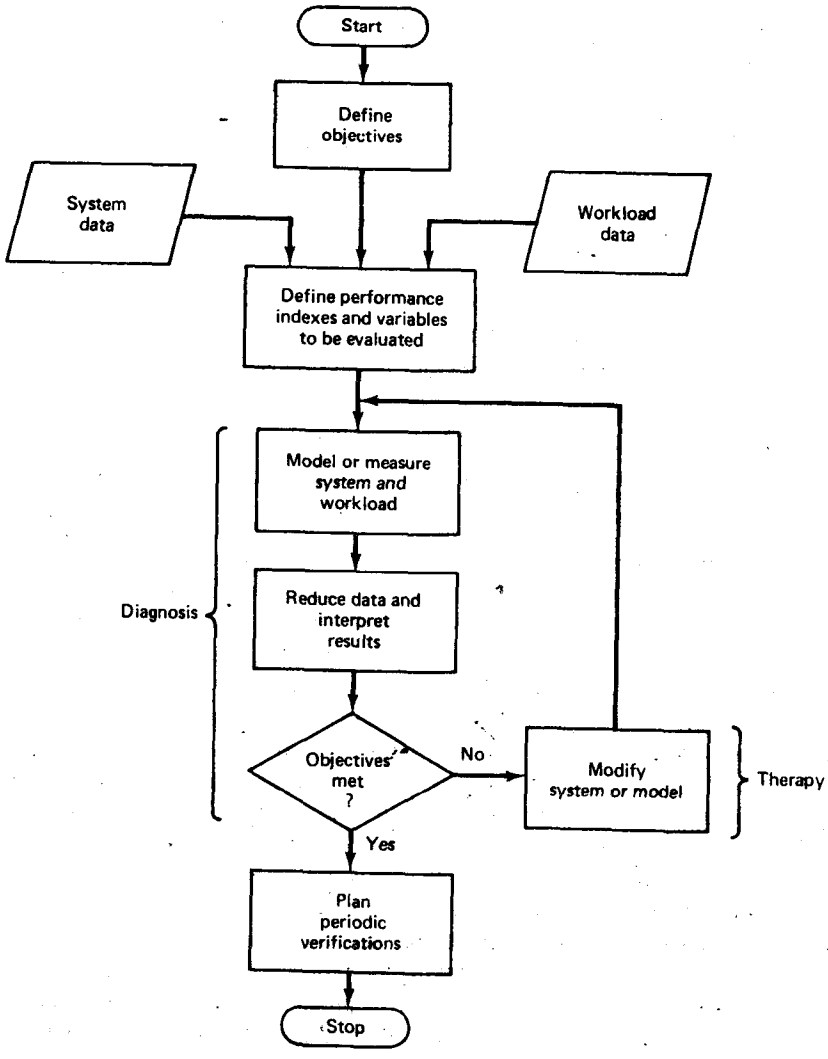

Figure 1.1 Scheme of a performance evaluation study.

evaluation techniques of the modeling type, it is also possible, although not always convenient, to postpone the application of the therapies suggested by all iterations to the end of the procedure.

1.3 THE REFERENCE SYSTEMS

The objects of the evaluation studies to be considered in this book have already been defined in Sec. 1.1; they are systems for the processing of information. To give the reader a more concrete definition, and a first approach to the viewpoint from which these systems

are to be observed for performance evaluation purposes, we shall discuss some basic types of system configurations. They are systems whose configuration is variable within certain limits. For example, the number of their I/O or mass storage devices is not rigidly specified. Their distinguishing characteristics are mainly the usage modes and the types of resource management used by the operating system. Examples of all four types of systems that have been chosen as reference systems (RS for brevity) can be found in similar configurations among existing computer systems. They are presented here in ascending order of complexity.

1.3.1 Uniprogrammed Batch-processing Reference System (UBRS)

In this system, the batch-processing mode is used and the main resources of the computer are managed in uniprogramming mode. A possible system configuration is shown in Fig. 1.2. This configuration is characterized by two channels: one (C_0) for the system console C and the peripheral units (card reader CR, line printer LP), and the other (C_1) for the mass storage units (disks D and tapes T). Users gain access to the system by punching their programs, data, and commands to the operating system on cards, which are read by the card reader CR. Channel C_0 transfers the information punched on the cards to the memory M, where the central processing unit (CPU) processes it. Each *program* usually consists of several *program steps*, for example, compilation, object program loading, execution, and storing the program in the file system. The output from these steps is sent to C_0, which controls its printing, as soon as the program produces it. Even the reading of cards is carried out in steps: only when the first step has been completed are the cards for the second step read in, and so on.

Thus, in this system, all the memory space not used by the operating system is

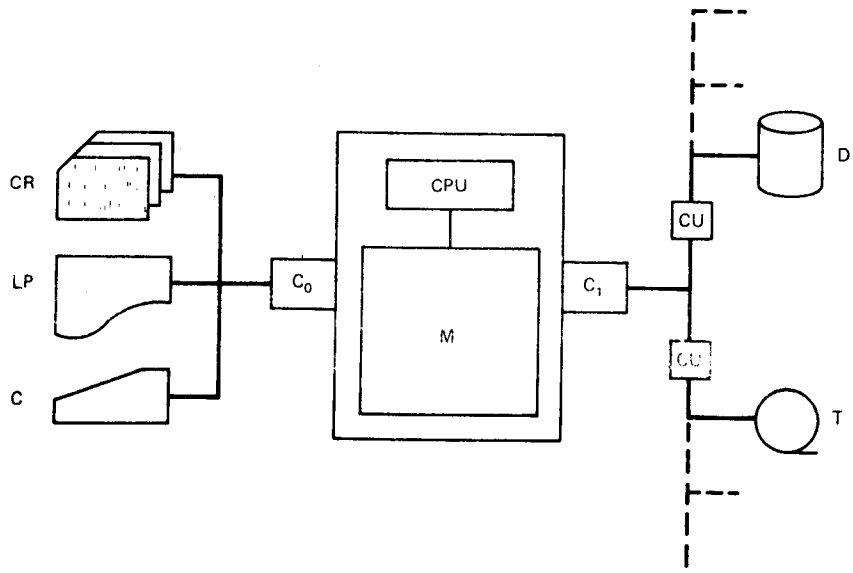

Figure 1.2 Typical UBRS configuration.