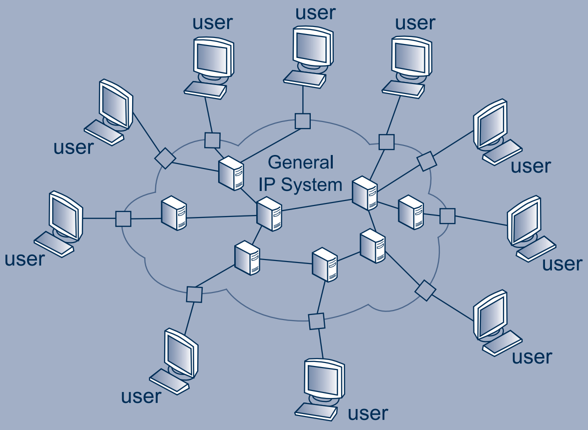
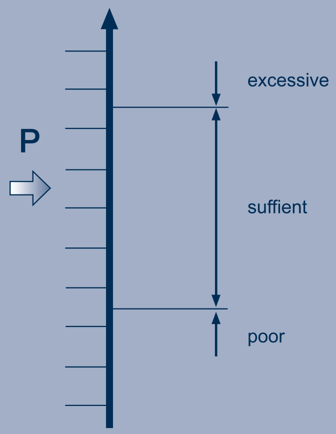
Werner Dirlewanger

Measurement and Rating of Computer Systems Performance and of Software Efficiency

An Introduction to the ISO / IEC 14756 Method and a Guide to its Application





CD-ROM included

kassel university press

About this book

ISO has developed a new method to describe and measure performance for a wide range of data processing system types and applications. This solves the problems arising from the great variety of incompatible methods and performance terms which have been proposed and are used. The method is presented in the International Standard ISO/IEC 14756. This textbook is written both for performance experts and beginners, and academic users. On the one hand it introduces the latest techniques of performance measurement, and on the other hand it is a guide on how to apply the standard.

The standard also includes additionally two advanced aspects. Firstly, it introduces a rating method to compute those performance values which are actually required by the user entirety (reference values) and a process which compares the reference values with the actual measured ones. This answers the question whether the measured performance satisfies the user entirety requirements. Secondly, the standard extends its method to assess the run-time efficiency of software. It introduces a method for quantifying and measuring this property both for application and system software.

Each chapter focuses on a particular aspect of performance measurement which is further illustrated with exercises. Solutions are given for each exercise. As measurement cannot be performed manually, software tools are needed. All needed software is included on a CD supplied with this book and published by the author under GNU license. This software is not intended to be a tool for performing professional measurements, but enables the reader to realize all steps of a performance or software efficiency measurement.

About the author

After receiving his doctorate at Erlangen University, Werner Dirlewanger became familiar with computer performance problems when he was head of various computer centres. Later, as Professor at Kassel University, his fields were software technology and computer architecture. This included active participation in computer performance measurement, evaluation and assessment. He became involved from the outset in the development of ISO/IEC 14756. In this book the author not only gives an introduction to the method but also shares with the reader his experiences of measurement projects of all types and sizes of data processing systems which he planned and performed as a consultant or which were performed under his leadership in his laboratories at the Department of Mathematics and Computer Science of Kassel University.

CD-ROM
included

ISBN-10: 3-89958-233-0
ISBN-13: 978-3-89958-233-8

Extract

Bibliographic information published by Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at http://dnb.ddb.de

ISBN-10: 3-89958-233-0
ISBN-13: 978-3-89958-233-8

2006, kassel university press GmbH, Kassel
www.upress.uni-kassel.de

Cover layout: Bettina Brand Grafikdesign, München
Printed by: Unidruckerei, University of Kassel
Printed in Germany

Preface

The performance of a data processing system is one of its most significant properties. For the user of such a system, a critical question is whether its performance will be adequate for the intended application. It is therefore desirable to be able to measure the performance of a data processing system. This question arises for all types and sizes of data processing systems. However, the measurement is difficult because of differing types, sizes and the high complexity of systems. A great variety of methods have been proposed and are being used to describe and measure performance. Each method was developed for a specific data processing system type and its use in a specific environment. Each has advantages and disadvantages. An additional problem is that the results of the different methods are not comparable.

To solve these problems ISO has developed a new method which is applicable for a wide range of data processing system types and applications. It is presented in the International Standard ISO/IEC 14756 "Information Technology - Measurement and rating of performance of computer-based software systems". Although this standard describes the method, it is not a tutorial. A textbook is desirable both for performance experts and beginners, and academic users. The aim of this book is to supply this need. On the one hand it introduces the latest techniques of computer performance measurement and of measuring software (run time) efficiency contained in the standard. And on the other hand it is a guide on how to apply the standard. However, you are recommended to buy the original ISO standard and read it in parallel.

This book focuses on measurement. It is not intended to be a general or broad overview of the wide field of all aspects of performance evaluation. For instance modelling, queuing techniques and simulation are not explained (many good books are available for those fields, for instance [ALLEN01], [BOLCH01], [JAIN01]). But the book discusses its own field in depth: i.e. measurement of system performance and of software (run time) efficiency. Additionally the author gives his experiences of many series of ISO-type measurements, of data processing systems of all sizes, which he planned and performed as a consultant or which have been performed in his laboratories.

This book is structured as follows: in each chapter the principles and methods are first explained, then illustrated with examples. Additionally there are exercises using the simple ISO-type measurement system DEMO, a simple demonstration software. These exercises can be performed on any UNIX operating system. The book shows how to do this using the well known and inexpensive LINUX. All needed software is included on a CD supplied with the book and is published by the author under GNU license. The demonstration software DEMO is not intended to be a tool for performing professional measurements. It only enables the reader to realise all steps of the measurement procedure and to observe all details as they occur. DEMO is deliberately not highly automated. It works interactively. In its native mode each step of the measurement procedure has to be manually controlled in order to show the trainee what happens.

The reader can use this book in different ways. For instance, if he is only interested in seeing some of the basic ideas of system performance measurement of the ISO method, then he should only read the corresponding chapters. But should he, for instance, be interested in a deeper understanding of the method and its applications then it would be mandatory to read all chapters thoroughly and to perform all exercises at the end of the chapters. Undoubtedly this will cause him to invest several hours of work, with possibly increased resources. It is up to the reader to decide what to do.

There is an additional benefit in using the DEMO demonstration software. As it shows in detail the principles of an ISO-type performance measurement tool, it helps industry and science in implementing an ISO-type performance measurement system for professional use. It must be stressed, however, that DEMO is only a demonstration tool on how an ISO-type measurement system works. It is neither a reference implementation nor a tool for professional measurement.

Although this book introduces the reader to modern performance measurement and to ISO/IEC 14756, it is not intended to replace this standard. The principles and methods are shown and explained, but for all normative details the reader must refer the original text of the standard. Should the reader be interested in mastering the complete mathematical presentation, this book can prepare him for more intensive study of ISO/IEC 14756.

The ISO standard uses conventional mathematical terminology and presentation. The author was tempted to do so in his book. Although he prefers this style, he resisted the temptation in order to enable less mathematically trained readers to follow the material.

ISO contributed an essential support to the realisation of this book by granting permission to reproduce, in the Appendix of this book, the workload examples specified in Annex "F" of the standard. I am most grateful for this permission.

Many individuals have stimulated and supported my work in preparing this book. I am really indebted to all of them. My gratitude goes to Annette and Robin Calkin for their many strenuous hours of proof reading and for their many recommendations to the text and to my children Christine and Christian for designing the graphics. My thanks also go to Reinhold Schlüpmann for his encouragement and editorial support in the development of the ISO standard; to Sascha Gortzewitz and Piotr Szulc for checking and testing the ISO method in my laboratories, and implementing, checking and testing the ISO workloads and for their contributions to the initial development of the DEMO software; to Eberhard Rudolph for writing Section 14.8 on applying the ISO method to function point measurement; and to Wolfgang Eichelberger, Reinhold Grabau and Stefan Kucera for their contributions when developing predecessors of the ISO method, putting them into practice and building up experience before publishing the ISO standard.

To all these people I extend my sincerest thanks. But finally I would like to especially thank my wife and children for their support and understanding when I was writing this book. It is dedicated to them in recognition of their patience during my countless hours absorbed in preparing the manuscript.

Werner Dirlewanger

Contents

1 General basics

This book introduces the modern principles of computer performance measurement and of measuring software run time efficiency. It refers especially to the method defined in the ISO/IEC 14756, see [ISO14756], and it is a guide of how to apply this standard. Finally it helps programmers when designing and implementing software tools for measurements.

Note: The author makes no warranty of any kind with regard to the material of this book and shall not be liable for defects contained herein or for incidental or consequential damages with the use of this material, either expressed or implied.

1.1 Computer performance evaluation and Benchmarking

What is "performance" ? We used to say: our computer has a high performance. In more detail one could say: it is fast and reliable; it forgets anything; it makes no errors; it is always available for work; it can do many different tasks; it is never tired. Looking further at these statements we find that the characteristics of a computer system can be divided into three classes.

First class: This class describes what the computer is able to do. It is the set of activities, the correctness of operation and of the computed results and, additionally, user friendliness (ergonomics).

Second class: This class describes how stable and consistent the computer is in its operation. It refers to the reliability of operation in the widest sense.

Third class: This class refers to the speed of operation. One aspect is the speed of executing the tasks, i.e. the time for delivery of the task results. Another aspect is the number of tasks which can be executed in a given time.

The classification assumes that a computer system is not restricted to a single-user stand-alone machine, such as a historical personal computer. Much more it is a general information processing system (IP system) including multi-user, multi-processor, or networking architectures.

The third class cited above means computer performance in the proper sense, i. e. the speed of operation. This book focuses on this meaning of performance.

What is performance evaluation ? This activity involves all evaluation methods of performance of data processing systems where performance is expressed by numbers and not only by words like high or low. Additionally evaluation can include the assessment of how good the performance satisfies the user requirements.

This book focuses on numerical performance values, determined by measurement and their assessment. Assessment also results in numerical values.

What is benchmarking ? The word benchmark is taken from the marks showing measures like feet or inches on the work bench of former craftsmen. In the data processing field several principles are in use for performing an evaluation. Some of them use a model of the IP system. Another principle is to investigate the real IP system by loading it with concrete applications, programs and data. Every method of determination performance values in such an experiment with the real system under test is benchmarking. This book is concerned with benchmarking and focuses on the ISO method.

1.2 System performance compared to component performance

The obvious way for characterising the performance of an IP system is simple. First decide which is the most significant component of the system, for instance the central processing unit (CPU). Then define a performance term, for instance the number of executed instructions per time unit, or a set of performance terms. Determine the values of this performance terms and take it (or them) as the performance of the IP system. This is the principle of "component performance".

This way may have been appropriate in earlier times when the IP systems had been simple single-user type machines having for instance, no multiprogramming features, no networks and only one central processor. In contrast with these early computers a present day IP system is much more complex. It is no longer sufficient to describe its performance by values of its most important hardware component. The IP system has to be seen in its entirety. Suitable performance terms have to be defined with regard to it. This consists of all system and network components producing the term "system performance". An important practical experience is that it is usually not possible to compute the system performance values from the component performance values. Therefore a separate method for determinating the system performance is needed. This book focuses on system performance and not on component performance.

Note: The component performance values and the system performance values often are named "external performance values". Contrary to these terms are the "internal values" as, for instance, the CPU utilisation, the storage utilisation, the utilisation of data busses, the length of job queues and the multiprogramming factor. In the past, those internal values were often used for performance values. They are no longer suitable. They only describe internal load situations of IP system components. In contrast to the external values they do not characterise the speed of operation. Internal values are not used in the ISO method and are not considered in this book. ■

1.3 ISO scope of system performance

The method defined in ISO/IEC 14756 [ISO14756] follows strictly the idea of system performance as described above. The system is the set of all co-operating hardware, software and network components. This set is regarded as a black box, which is connected via the set of its interfaces to its users (see Fig. 1-1). The users are typically humans, but some of them can be machines which submit tasks to the system via an interface.

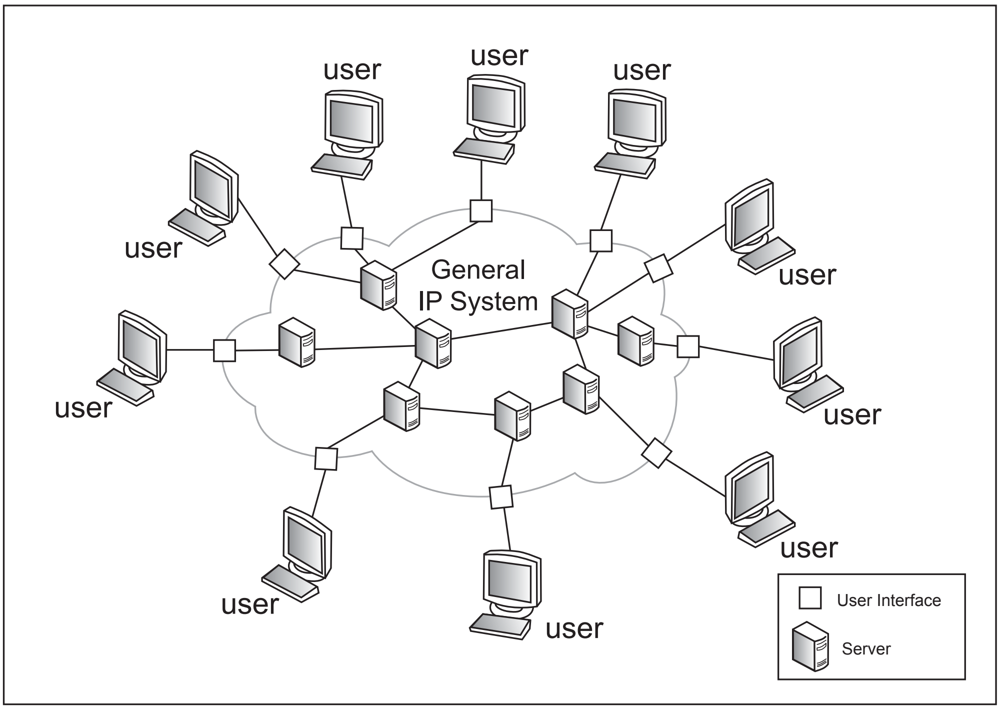


Fig. 1-1 The general IP system and its user entirety

1.4 Measurement of computer performance compared to prediction methods

The field of computer performance evaluation has the three subfields measurement, simulation and modelling.

Measurement means carrying out a real experiment with the real IP system operating in real time with real users. A monitoring feature records all necessary data during the experiment. Performance values are computed from the recorded data.

For simulation a mostly simplified functional model of the IP system and its users is developed. A computer program is then written which runs the model. This program may run in slow motion, in real time or time-lapse mode. It does not matter which one of these three modes is used. All necessary data during this simulated run are recorded by a software monitor. Performance values are computed from the recorded data.

For modelling a very simplified functional model of the IP system and its users is developed. From this a mathematical model is derived by means of queuing theory. This model can be analysed by solving the so-called state equations merely numerically. But sometimes also the explicit formulae of the interesting performance terms are found. Then the performance values can be computed by use of the formulae.

In contrast to this in a measurement the real IP system is investigated and tested. Simulation and modelling use only models of the system under test. Therefore the last two methods deliver performance values of models. These are estimated and not measured values. Consequently simulation and modelling deliver only predictions of performance values. This book is not concerned with prediction methods. It focuses on real measurement (as represented by the ISO method).

1.5 What is rating of performance and why is it needed ?

The results of a performance determination, independent of whether it is done by a measurement method or by a prediction method, are performance values. They are values of physical properties of the IP system under test (SUT), i.e. physical values. They are not information about a more far-reaching aspect of using data processing systems. This is the question if a regarded IP system fulfils the performance requirements of its user entirety.

The requirements of the user entirety are primarily non numeric values such as "poor", "sufficient" or "excessive". We have to define which ranges of performance values correspond to each of these three non numeric values. I.e. we have to relate the non numeric values to the scale of the performance values. Entering the performance values of the SUT into this scale delivers the rating result, for instance "the performance is sufficient". This is shown in Fig. 1-2 .

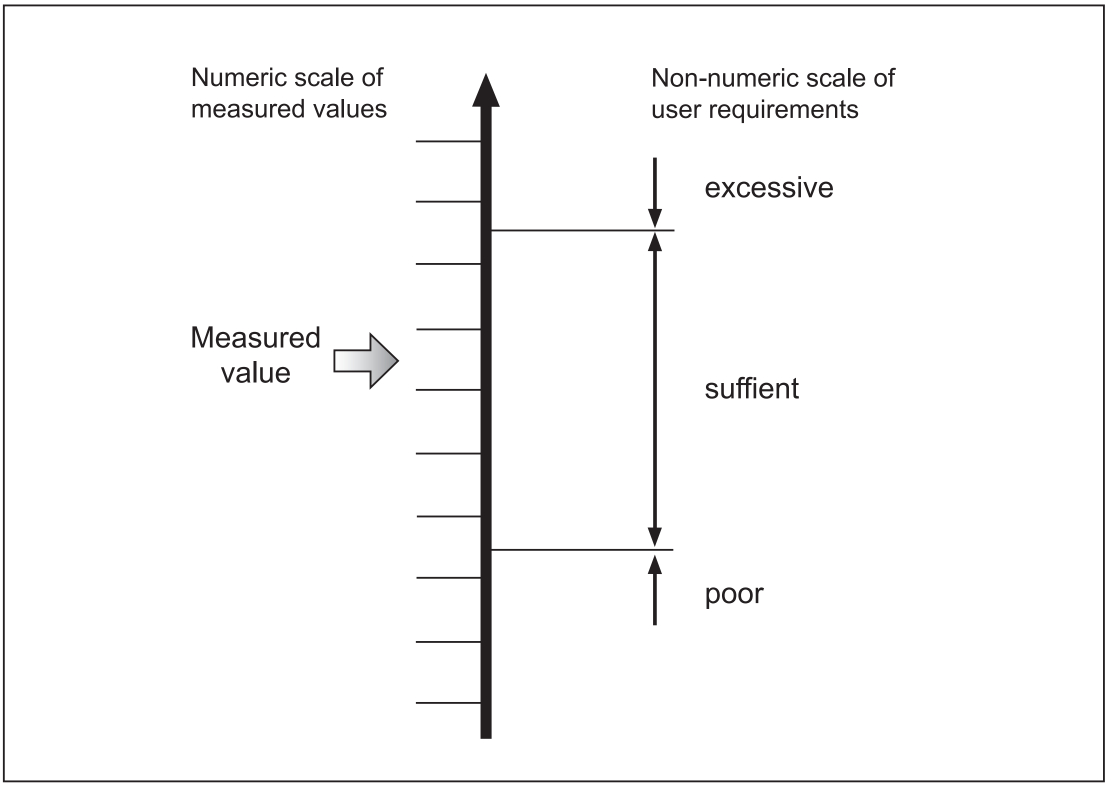


Fig. 1-2 Rating of performance values

The ISO method (which is the focus of this book) contains a comfortable rating procedure. It is explained further in chapter 7.

1.6 Basic principles and philosophy of ISO/IEC 14756

1.7 Overview of ISO/IEC 14756

1.8 Exercises

<u>Exercise 1:</u> Making available a demonstration test bed

The ISO method cannot be performed manually. It needs a computer-aided tool, which is the RTE, and a suitable SUT.

Part 1
You need two Pentium compatible computers, each running a LINUX operating system. One is used as the platform of the RTE, the other for the SUT. These two machines have to be connected by an Ethernet line with a speed of at least 10 kbits/second. The RTE machine should have a CPU with a speed of at least 500 Mhz, and be significantly faster than the SUT machine.

Part 2
Create in the RTE a user named "operator" and in the SUT a number users (for instance 25) named "user1", "user2",..... . Make sure that the "operator" can access via the network all users of the SUT with the UNIX commands "telnet", "ftp" and "rsh".

<u>Exercise 2:</u> Installing the RTE software DEMO

This book includes a complete ISO-type performance measurement system: the DEMO. This software is available under GNU License and is contained on the CD-ROM as part of this book. The DEMO is not a professional system. See Section 14.9.3 for its limitations. But it is suitable for learning and teaching. Install DEMO on the RTE machine according to the instructions on the CD-ROM.

Remark: Check if your computer uses a LINUX operating system release needed by DEMO (see file CD/DEMO-20/release.txt). If not, a few alterations will be necessary to DEMO.

<u>Exercise 3:</u> Installing the system software components on the SUT

Compilers (including those for C, FORTRAN77, COBOL ANS85) have to be installed and also the editor "ed". They are needed for running the ISO workload examples in the following chapters of this book. For economy you are recommended to use low cost software (e. g. free or GNU licensed), which is suitable for running the workloads.

<u>Exercise 4:</u> Documentation of the test bed configuration

Write down the documentation of the hardware and software of the test bed.

<u>Solutions</u>
For solutions see file
`CD/Solutions/Solutions-Section1-8.pdf .`

5 Computing the ISO performance values from the measurement result file (logfile)

5.1 Overview of the ISO performance terms

Performance P is a triple of three terms

$$\boxed{P = (B, T_{ME}, E)} \qquad (5.1)$$

where

- B is the (total) throughput vector
- T_{ME} is the mean execution time vector
- E is the timely throughput vector

It is important to understand that performance is the set of these three terms. The reason is that no term can be computed from the others. In the general case of the SUT being a black box the three terms are independent of each other.

These terms are determined for each task type. I.e. each is a tuple of as much values as there are task types. For instance, for m = 4 task types the performance is described by the three terms each subsuming four subterms, i.e. by 12 subterms. In general the following holds:

$$B = (B(1), B(2), \ldots, B(m)) \qquad (5.2)$$
$$T_{ME} = (T_{ME}(1), T_{ME}(2), \ldots, T_{ME}(m)) \qquad (5.3)$$
$$E = (E(1), E(2), \ldots, E(m)) \qquad (5.4)$$

P consists of 3*m subterms. This is a complex but very powerful definition of performance.

The values of P are computed from the logfile records, concerning the rating interval (RI). If following the recommendation of Sections 3.4 and 3.5 the logfile is separated into two parts. For the computation of P is only the RI part of the logfile is needed.

5.2 The "total throughput vector" B

B is the set of m terms B(j) where j is the current number of the task type. Every B(j) refers to the duration T_R of the RI

$$\boxed{T_R = t_2 - t_1} \quad . \qquad (5.5)$$

B(j) is the number of type j tasks submitted to the SUT by the RTE per time unit.

$$\boxed{B(j) = b(j) / T_R} \quad (5.6)$$

In the above formula $b(j)$ is the total number of tasks of type j submitted by all emulated users to the SUT during the RI. The set of the m terms $B(j)$ yields the throughput vector as defined in equation (5.2) above.

The computation of B is simple. Analyse the RI part of the logfile. Counting the number of tasks of each task type yields the m values $b(j)$. According to equation (5.6) above each of these values has to be divided by T_R . This yields the m values $B(j)$. This set of values is the throughput vector B as defined in equation (5.2) .

5.3 The "mean execution time vector" T_{ME}

T_{ME} is the set of m terms $T_{ME}(j)$ where j is the current number of the task type. Every $T_{ME}(j)$ refers to the RI and $T_{ME}(j)$ is the mean execution time of all type j tasks submitted to the SUT by all emulated users within the RI.

$T_{ME}(j)$ is simple to compute. Analyse the RI part of the logfile. The execution times of all type j tasks within the RI. These times are $t_{ET}(j,1)$, $t_{ET}(j,2)$,..... . The total number of those values is $b(j)$. Therefore the last element in the series will have the symbol $t_{ET}(j,b(j))$. Adding them and dividing by $b(j)$ yields the mean value $T_{ME}(j)$.

$$\boxed{T_{ME}(j) = ((t_{ET}(j,1) + t_{ET}(j,2) + ...+t_{ET}(j,b(j))) / b(j)} \quad (5.7)$$

Performing this simple procedure for all m task types yields the "mean execution time vector T_{ME} in equation (5.3) above.

5.4 The "timely throughput vector" E

5.4.1 The principle of E

Although, E is a somewhat unusual term, it is easy to understand.

As explained in Section 5.2 $b(j)$ is the number of type j tasks submitted to the SUT during the RI. With regard to the according timeliness function (see Section 2.5) some tasks may not have been executed in time. Let $e(j)$ be the total number of timely executed tasks of type j tasks during the RI. The maximum value of $e(j)$ may be achieved if all type j tasks have been executed in time. Therefore,

$$\boxed{e(j) \le b(j)} \quad . \quad (5.8)$$

The "timely throughput" $E(j)$ of type j tasks is the number per time unit. Therefore,

$$\boxed{E(j) = e(j) / T_R} \quad . \tag{5.9}$$

From this definition and equation (5.8) above follows that

$$\boxed{E(j) \leq B(j)} \tag{5.10}$$

This inequality is the mathematical representation of the fact that the rate of timely executed tasks cannot exceed the total rate of tasks (with respect to j type tasks).

The procedure for computing $e(j)$ (for getting $E(j)$ by use of equation (5.9) above) is somewhat sophisticated. It is explained in the Section 5.4.2 .

5.4.2 Computing e(j)

The procedure for computing $e(j)$ ....

It is important to understand that this algorithm only counts the number of timely executed tasks. It does not decide individually whether a task is executed timely, except for those beyond the uppermost time class. They are clearly tasks which are not timely.

The algorithm is implemented in DEMO. For subroutines see [DIN01], [DIRLE02] and [ISO14756].

5.4.3 The "timely throughput vector" E

The number `e(j)` of timely executed type j tasks is determined according to Section 5.4.2. The timely throughput `E(j)` of type `j` tasks is to be computed according to equation (5.9) above. Performing this for all `m` task types yields the "timely throughput vector" `E` as defined in equation (5.4) above.

5.5 Exercises

<u>Exercise 1:</u> Computing P from the logfile

Compute P (performance) manually from the logfile below. It consists of the two files `ZEIT` and `DIN.DAT`. It was produced by the measurement of Exercise 1, Section 4.5 . These files are found in directory

`CD/Sol-files/Mment-ch4/ARCHIVE-TTxs-Mment/` .

For the sake of simplicity overlook that this logfile was produced by applying so-called individual rating intervals and use for duration T_R of the RI the mean value of the individual rating intervals of the 2 users.

Apply the following timeliness function TF_1 for all task types:

TF_1 :

z = 2

k	$g_T(k)$	$r_T(k)$
1	1.5 sec	0.80
2	2.5 sec	1.00

file ZEIT

```
                  1    1            116.427 t_1
                  2    1            112.521 t_1
   *
                  1    1            233.050 t_2
                  2    1            225.125 t_2
   *
```

file DIN.DAT

```
   1       1   1   1   1   116.449   124.467   125.509 0       X
   2       1   1   1   2   125.513   135.544   137.573 0       X
   3       1   1   2   2   137.576   154.599   156.618 0       X
   4       1   1   2   3   156.622   166.637   169.668 0       X
   5       1   1   1   1   169.672   188.686   189.745 0       X
   6       1   1   1   2   189.748   195.780   197.799 0       X
   7       1   1   1   1   197.803   202.859   203.884 0       X
   8       1   1   1   2   203.888   211.903   213.925 0       X
   9       1   1   1   1   213.928   214.943   215.990 0       X
  10       1   1   1   2   215.994   231.025   233.048 0       X
  11       2   1   2   2   112.544   112.568   114.616 0       X
  12       2   1   2   3   114.619   121.646   124.675 0       X
  13       2   1   1   1   124.679   140.695   141.721 0       X
  14       2   1   1   2   141.724   156.738   158.778 0       X
  15       2   1   2   2   158.782   165.841   167.873 0       X
  16       2   1   2   3   167.877   174.905   177.933 0       X
  17       2   1   2   2   177.937   179.951   181.973 0       X
  18       2   1   2   3   181.977   193.987   197.029 0       X
  19       2   1   1   1   197.032   211.060   212.082 0       X
  20       2   1   1   2   212.086   223.102   225.123 0       X
*
```

Note: It is expected that your measurement produces results that differ a little from those above because of differences due to probabilistic events in detail. ■

Exercise 2: Similar workload (slower users)

Repeat the measurement of Exercise 1, Section 4.5 but double all preparation times in the task lists. Compute `P` manually from the logfile. For the sake of simplicity overlook - as in Exercise 1 - that the logfile is produced by applying so-called individual rating intervals. For duration T_R of the RI use the mean value of the rating intervals of the 2 users. Apply the timeliness function TF_1 above for all task types. Compute `P` and explain all changes.

Exercise 3: Similar workload (faster users)

Repeat the measurement of Exercise 1, Section 4.5 but halve all preparation times in the task lists. Compute `P` manually from the logfile. For the sake of simplicity overlook - as in Exercise 1 - that the logfile is produced by applying so-called individual rating intervals. For duration T_R of the RI use the mean value of the rating intervals of the 2 users. Apply the timeliness function TF_1 above for all task types. Compute P and explain all changes.

Solutions

For solutions see file
`CD/Solutions/Solutions-Section5-5.pdf` .

7 Rating the measured performance values

7.1 The principle of the ISO rating

The measured performance $P = (B, T_{ME}, E)$ is a set of physical values. The user of the IP system is interested in these values but much more whether they satisfy the user entirety requirements.

Therefore ISO/IEC 14756 introduced a rating process which compares P with those performance values which are actually required by the user entity (reference values). These values are

1. the total throughput reference vector B_{Ref}
2. the mean execution time reference vector T_{Ref}
   and
3. the requirement "all tasks are completed timely" .

The result of the rating is the final decision on the SUT: "satisfactory" or "unsatisfactory".

7.2 The ISO theoretical reference machine

The ISO theoretical reference machine does not really exist. It is defined as a fictive SUT that just fulfils the timeliness functions stated in the workload (when the SUT is driven by this workload). No task will be executed faster than necessary but all tasks are executed in time. The performance of this fictive SUT represents the requirements of the user entirety and is named P_{Ref} .

$$\boxed{P_{Ref} = (B_{Ref},\ T_{Ref},\ E_{Ref})} \tag{7.1}$$

$$B_{Ref} = (\ B_{Ref}(1),\ B_{Ref}(2),\ \ldots, B_{Ref}(m)\ ) \tag{7.2}$$
$$T_{Ref} = (\ T_{Ref}(1),\ T_{Ref}(2),\ \ldots, T_{Ref}(m)\ ) \tag{7.3}$$
$$E_{Ref} = (\ E_{Ref}(1),\ E_{Ref}(2),\ \ldots, E_{Ref}(m)\ ) \tag{7.4}$$

P_{Ref} can be determined as follows.......

8 The performance measure N_{max}

8.1 Maximum number of timely served users (N_{max})

People like to describe the performance of IP systems by a scalar value such as the old-fashioned "million instructions per second" (MIPS) of mainframe machines or "giga floating point operations per second" (GFLOPS) of super computers. Contrary to this the ISO method makes it clear that system performance measure is not a scalar but a vector (or even a set of three vectors).

To approximate the theoretical unreachable goal of a scalar performance measure we can take an ISO workload and modify it stepwise. There are many ideas of doing so.

For instance we can modify the activity types by using a replication factor (for examples see Section 11.3.1). We can perform several measurements increasing the replication factor and determine the rating values. We can use the replication factor as a performance term. Performance is the limiting value of the replication factor when the ISO rating changes from "satisfactory" to "unsatisfactory" (see Section 14.3).

One idea of modifying a workload stepwise seems to be both attractive and very practical. It is to increase the number of users of a defined workload while keeping constant all other parameters of the WPS. When the rating changes from "satisfactory" to "unsatisfactory", N_{max} is the number of users. But it is important not to change the basic characteristics of the workload when increasing the number of users. This implies that we eventually cannot increase the number of emulated users by an arbitrary increment. This aspect is detailed pointed out in Section 8.2 .

N_{max} is a powerful performance measure, but it is not an absolute measure. It always refers to a defined workload. N_{max} is a not normative performance measure of ISO/IEC 14756 but it is tolerated by the standard. It is derived from the ISO terms. It is only applicable to multi-user SUTs.

8.2 Incrementing the number of users

If the workload has only one user type (i.e. $n = 1$) the lowest possible number of users is 1. We can set $N_{user}(1) = 1$. Then $N_{tot} = 1$ and the workload is a basic workload. The smallest step for increasing N_{tot} is 1. We are free to increase N_{tot} in steps of 1 or more.

If the workload has more than one user type (i.e. $n > 1$) the situation is more complicated. As an example we take a workload with two user types, i.e. $n = 2$, each having one user. Obviously $N_{tot} = 1$ is impossible. The minimum value of N_{tot} is 2. Increasing N_{tot} from 2 to 3 users would violate the principle of keeping the basic characteristics of the workload. The relative percentages of the chain types would be changed. But increasing N_{tot} from 2 to 4 would keep the percentages the same. The minimum increment for changing N_{tot} is therefore a multiple of 2. Consequently N_{max} would be an even number.

Generally the basic workload of a given workload is found by determining the greatest common divisor N_{GCD} and then dividing all $n(i)$ by N_{GCD} .

.... Example: ....

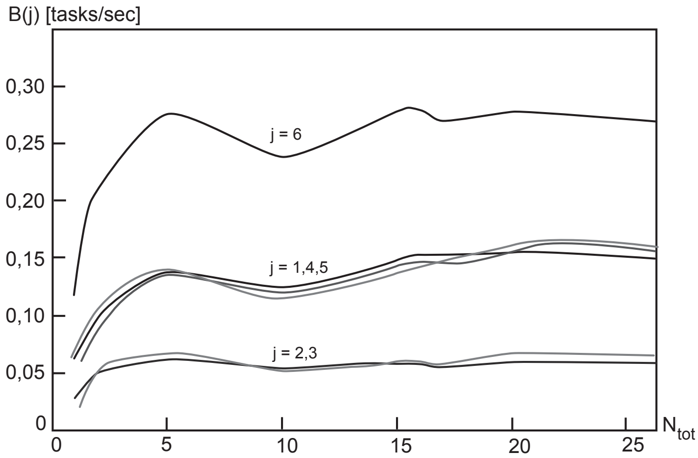
8.3 Measurement series

.
.
.... Figures 8-1 and 8-2 show an example of the result of a measurement series.
.
.
.

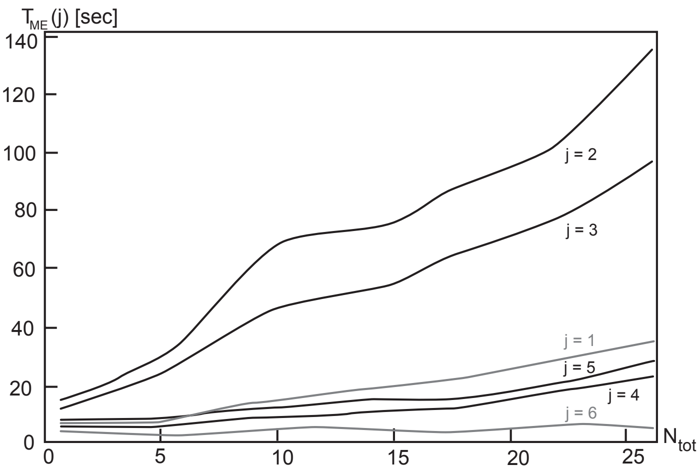
8.4 Acceptable tolerances of N_{max}

8.5 Experiences from various measurement series

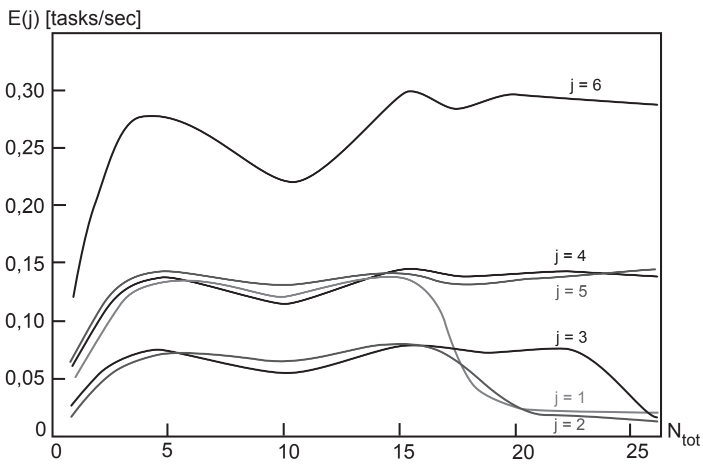
.
.
.
.



8-1a Total throughput

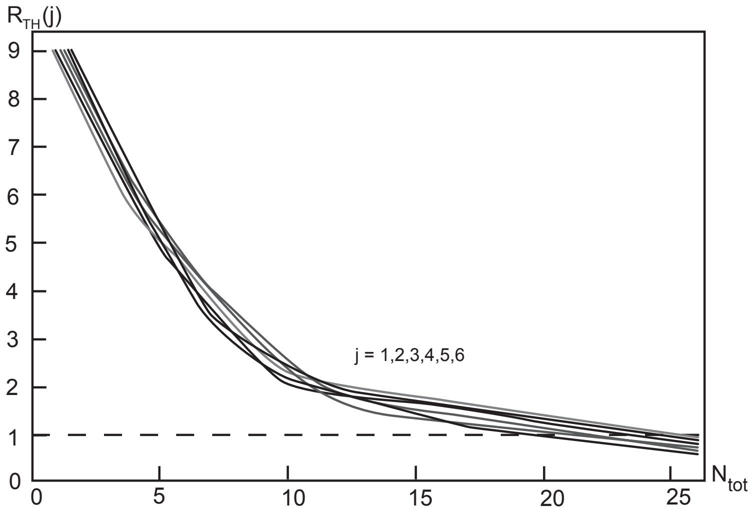


8-1b Mean execution time

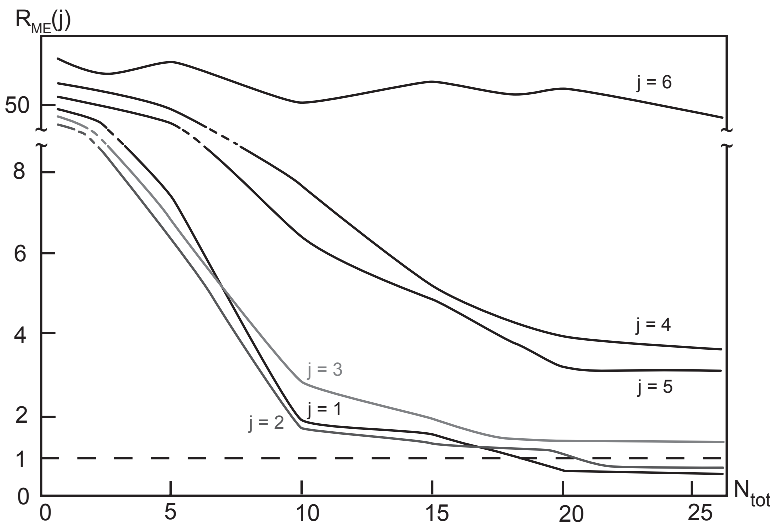


8-1c Timely throughput

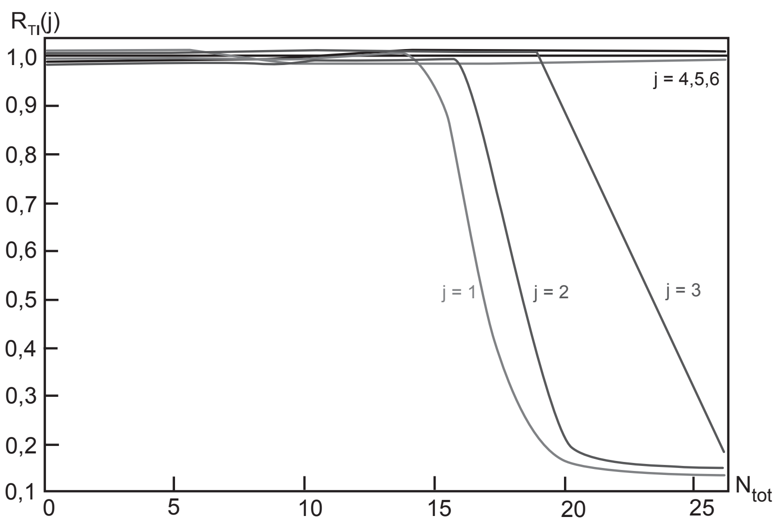
Fig. 8-1 Measured performance values of a measurement series



8-2a Throughput rating



8-2b Mean execution time rating



8-2c Timeliness rating

Fig. 8-2 Rating values of a measurement series

But if most task types defined in the WPS have the task mode value of 0 then typically the R_{TH} values are the criteria that determine the value of N_{tot} . I.e. the throughput is the deciding criterion (and not mean execution times or timeliness). Examples of such workloads are the ISO workloads COMPCENTER1, Version "B" and COMPCENTER2, Version "B" (see Annex F in ISO/IEC 14756 and also Sections 11.3.2, 11.3.3, 11.5.1.1 and 11.5.2.1 of this book).

Although in principle the duration of the rating interval T_R increases with increasing N_{tot} this effect is not very strong marked. Usually T_R is not much longer for $N_{tot} = N_{max}$ than for N_{tot} = value of the basic workload. T_R only increases sharply when N_{tot} is clearly greater than N_{max} .

8.6 Exercises

Exercise 1: Measurement using a basic workload

Preparation

Use the WPS of Exercise 1 of Section 6.6 but change the activity types to `TT1a`, `TT2a` and `TT3a` as shown below.

Part 1
Write down WPS and `SAG.DAT` .

Part 2
Check whether this workload is a basic workload.

Part 3
Determine the increment of N_{tot} for performing a measurement series.

Part 4
Set $N_{tot} = 2$ and perform a measurement using `REP` = 1.
Report the measured performance and rating values.

Exercise 2: Performing a measurement series

Become familiar with the operator utility `"./runMeasurement x"` of DEMO.
Perform a measurement series and determine N_{max}. Use a REP value suitable for your SUT. Store the measurement result files using the operator utility
`"./saveMment ddd"` or `"./saveMment2 ddd"`.
All files are stored in the directory `"ddd"` .

Note: For instance if your SUT has a 1.2 Mhz Intel CPU, a value of `REP`=30 can be suitable. If your SUT uses a faster CPU the following problem can arise. The maximum number of users to be emulated by DEMO 2.0 is 99. But N_{max} can be greater than 99. Then use a value greater than 30 for `REP`. There is another problem that can limit the maximum number of users emulated by DEMO. Most of the LINUX operating systems support less than 99 active Xterminals or remote shells when using the default values of the operating system parameters. When using DEMO in this exercise, the easiest way is to set `REP` to a value that does not cause N_{max} to exceed 25 or 30.

Exercise 3: Repeat the measurement series with a modified WPS

Use the same workload as in Exercise 2. But modify its WPS by setting the task modes of all task types to 0 (NO WAIT). Use the same `REP` value as in Exercise 2. Observe the effect of increasing N_{tot} on `B` and the rating values.

Exercise 4: Greater REP value

Repeat the measurement series of Exercise 2 with a significantly greater REP value.

Solutions
For solutions see file
`CD/Solutions/Solutions-Section8-6.pdf` .

Chapter 10   Measurement of software (runtime) efficiency

10.1 A hierarchical model for information processing systems

10.2 The reference environment and the term run time efficiency

10.3 The measurement procedure and measures of software efficiency

10.4 Examples

10.4.1 Application software efficiency

10.4.2 System software efficiency

Chapter 14 Miscellaneous aspects

Appendix A: CD as a part of this book

1. Foreword
   See file `CD/Foreword-of-this-CD.txt`

2. Contents
   File `CD/Contents-of-this-CD.txt`

3. GNU General Public license
   See file `CD/GNU-gpl.txt`

4. ISO-IEC 14756 original workloads
   See directory `CD/iso14756-orig-workloads/`

5. Logical steps of the OSCPs of the ISO computer centre workloads
   See file `CD/Supplement-to-ISO14756.pdf`
   Contents:
   - Workload COMPCENTER1
   - Workload COMPCENTER2
   - Workload COMPCENTER3

6. Two ISO workloads converted for LINUX SuSE 9.1

6.1 Workload COMPCENTER1
   See directory `CD/Linux-workloads/CC1-Linux9/`
   Installation:
   see file `CD/Linux-workloads/Install-CC1-Linux9.txt`

6.2 Workload COMPCENTER2
   See directory `CD/Linux-workloads/CC2-Linux9/`
   Installation:
   see file `CD/Linux-workloads/Install-CC2-Linux9.txt`

7. Sketch of the ISO workload COMPCENTER1,
   converted for NT 4.0
   See directory `CD/NT-workloads/`

8. Sketches of some ISO type individual workloads
   See directories
   `CD/Workload-sketches/ERP-WL/`
   `CD/Workload-sketches/ExternalDWH/`
   `CD/Workload-sketches/KS-Web/`
   `CD/Workload-sketches/Mainframes/`
   `CD/Workload-sketches/Web99/`

9. Measurement system DEMO 2.0 (implemented for LINUX SuSE 9.1)
9.1 Manual
See directory `CD/DEMO-20/DEMO-manual/`
9.2 Software
See directory `CD/DEMO-20/DEMO-sw/`
9.3 XDEMO
See file `CD/DEMO-20/XDEMO.txt`

10. Detailed documentation of a measurement using DEMO
See directory `CD/Mexample/`

11. Solutions of exercises
See directory `CD/Solutions/`

12. Files of the exercises
See directory `CD/Sol-files/`

Note 1: This compact disc was created using a LINUX operating system. If using another operating system for reading it is not guaranteed that correct data are obtained or displayed, but "`.txt`" files, "`.email`" files and files named `readme` or `README` can be displayed also by WINDOWS Editor or WordPad . "`.pdf`" files can be opened by ACROBAT reader. ■

Note 2: For extracting the "tar" archives
see file `CD/Contents-of-this-CD.txt` . ■

References

Abbreviations

Symbols

Index