CHAPTER 1

INTRODUCTION AND BASIC CONCEPTS

Performance evaluation of computer and telecommunication systems has become an increasingly important issue given their general pervasiveness. An evaluation of these systems is needed at every stage in their life. There is no point in designing and implementing a new system that does not have competitive performance/cost ratio. Performance evaluation of an existing system is also essential because it helps to determine how well it is performing and whether any improvements are needed to enhance the performance.

Computer and telecommunication systems performance can be evaluated using the measurement, analytic modeling, and simulation techniques. Once a system has been built and is running, its performance can be evaluated using the measurement technique. To evaluate the performance of a component or a subsystem that cannot be measured, for example, during the design and development phases, it is necessary to use analytic or simulation modeling so as to predict the performance [1–15].

The objective of this book is to provide an up-to-date treatment of the fundamental techniques and applications of performance evaluation of computer and telecommunication systems.

Fundamentals of Performance Evaluation of Computer and Telecommunication Systems,
By Mohammad S. Obaidat and Noureddine A. Boudriga

1.1 BACKGROUND

Performance evaluation aims at predicting a system's behavior in a quantitative manner. When a new computer and telecommunication system is to be built or an existing system has to be tuned, reconfigured, or adapted, a performance evaluation can be employed to forecast the impact of architectural or implementation modifications on the overall system performance.

Today's computer and telecommunication systems are more complex, more rapidly evolving, and more pervasive and essential to numerous parties that range from individual users to corporations. This results in an increasing interest to find new effective tools and techniques to assist in understanding the behavior and performance of existing systems as well as to predict the performance of the ones that are being designed. Such an understanding can help in providing quantitative answers to questions that arise during the life cycles of the system under study, such as during initial design stages and implementation, during sizing and acquisition, and during evolution and fine tuning.

To evaluate the performance of a system, we can use the measurement technique if the system exists and it is possible to conduct the required experiments and testing on it. However, when the system does not exist or conducting the measurements is expensive or catastrophic, then we rely on simulation and analytic modeling techniques. The last two techniques try to answer important questions related to the design or tuning of the system under study, where the term "system" refers to a collection of hardware, software, and firmware components that make a computer or telecommunication system. It could be a hardware component such as an Asynchronous Transfer Mode (ATM) switch and a central processing unit (CPU); a software system, such as a database system; or a network of several processors, such as a multiprocessor computer system or a local area network (LAN) [1–21].

Examples of the type of predictions that can be made from performance analysis studies include [1–21]:

- The number of stations that can be connected to a LAN and still maintain a reasonable average frame delay and throughput
- The fraction of cells that can be discarded from an ATM system during overload
- The number of sources that can be supported in an Available Bit Rate (ABR) voice service over ATM networks so that a specific cell loss ratio (CLR) threshold is not exceeded
- The fraction of calls that are blocked on outgoing lines of a company's telephone system and how much improvement we can get if an extra line is added
- The improvement in speedup and latency that we can achieve if we add a processor or two to a multiprocessor system
- The best switch architecture for a specific application

- The improvement in mean response time of a network if the copper wires are replaced by optical fiber

All such questions and more can be answered using the three main techniques of performance evaluation. The results from one or more of these techniques can be used to validate the results obtained by the other. For example, we can use analytic results to validate simulation results or vice versa. We can also use the analytic results from a prototype version of the system, which can be designed to validate simulation results and so on.

It is worth mentioning here that validation and versification (V&V) are important procedures that should be performed for any simulation model. Also, validation is needed for analytic models. These subjects are important for performance evaluation, and many conferences and journals have dedicated tracks/section for them [1, 2, 14, 15]. We will deal with V&V in Chapter 11.

1.2 PERFORMANCE EVALUATION VIEWPOINTS AND CONCEPTS

All engineering systems should be designed and operated with specific performance requirements in mind. It is essential that all performance requirements of any system to be designed should be stated at the outset and before investing time and money in the final design stages, which include testing and implementation. The work conducted by Erlang in 1909 on telephone exchange is considered the beginning of performance evaluation as a new discipline. Even though the range of performance evaluation is now wide, the fundamentals are the same.

It is desirable to evaluate the performance of a system make sure that it is suitable for the intended applications and that it is cost effective to build it, or if it exists physically, it can be operated and tuned to provide optimum performance under given resource constraints and operating conditions. The best performance metrics and desired operational requirements of a system under study depend on the nature of applications, constraints, and environments. For example, the metrics to be considered for a LAN or a computer system that are operating in a manned space shuttle may be different from those on a campus of a company or college.

Experimentation with the real system or a prototype version of it is usually expensive, laborious, inflexible, and prohibitive. Moreover, it gives accurate information about the system under special cases or a specific set of assumptions. However, analytic modeling and simulation are flexible, inexpensive, and usually provide fast results.

In the context of modeling, we can define a model as an abstraction of the system or subsystem under study. A model can be envisioned as a description of a system by symbolic language or theory to be viewed as a system with which the world of objects can be communicated. Shannon defined a model as "the process of designing a computerized model of a system (or a process) and

conducting experiments with this model for the purpose of either understanding the behavior of the system or of evaluating various strategies for the operation of the system'' [1, 2].

In the context of performance evaluation, we can provide three possible definitions for the term ''system'' [1, 2, 14]:

- An assemblage of objects so combined by nature or human as to form an integral unit
- A regularly interacting or interdependent group of objects forming a unified whole [*Webster's Dictionary*]
- A combination of components/objects that act together to perform a function not possible with any of the individual parts [*IEEE Standard Dictionary of Electrical and Electronic Terms*]
- A set of objects with certain interactions between them

From the above definitions, we observe two major features in these definitions:

1. A system consists of interacting objects/components.
2. A system is associated with a function/work that it performs.

It is important to mention here that a system should not always be coupled with physical objects and natural laws as a set of equations that defines a function is considered a system.

Systems can be divided into the following three types:

- Continuous systems: Here the state changes continuously over time.
- Discrete systems: In this type, the state varies in fixed quanta.
- Hybrid systems: Here, the system state variables may change continuously in response to some events, whereas others may vary discretely.

We can also classify systems into stochastic and deterministic types. The stochastic systems contain a certain amount of randomness in their transitions from one state to another. A stochastic system can enter more than one possible state in response to a stimulus. Clearly, a stochastic system is nondeterministic because the next state cannot be unequivocally predicted if the current state and the stimulus are known. In the deterministic systems, the new state of the system is completely determined by the previous state and by the stimulus.

Modeling and simulation is considered one of the best instruments to predict performance as they roll data into knowledge and knowledge into experience. It is also flexible, cost effective, and risk free. In modeling and simulation, we need three types of entities: (1) real system, (2) model, and (3) simulator. These entities have to be understood as well as their interrelation to one another. The real system, if either it exists physically or its design is available, is a supply of

raw data, whereas the model is a set of instructions for data generating. The simulator (simulation program) is a tool to implement the model and carry out its instructions [1, 2, 6, 14, 15].

Moreover, systems can be divided into open and closed systems. In a closed system, all state changes are prompted by internal activities, whereas in an open system, state change occurs in response to both internal and external activities.

1.3 GOALS OF PERFORMANCE EVALUATION

The objectives of any performance evaluation study depend mainly on the interest, applications, skills, and capabilities of the analysts. Nevertheless, common goals in any performance evaluation study are typical for computer and telecommunication systems [1, 2, 4]. The major ones are briefly described below.

1. **Compare alternative system designs**. Here, the goal is to compare the performance of different systems or component designs for a specific application. Examples include deciding the best ATM switch for a specific application or the type of buffering used in it. Other examples include choosing the optimum number of processors in a parallel processing system, the type of interconnection network, size and number of disk drives, and type of compiler or operating system. The objective of performance analysis in this case is to find quantitatively the best configuration under the considered operating environments.
2. **Procurement.** In this case, the goal is to find the most cost-effective system for a specific application. It is essential to weigh out the benefit of choosing an expensive system that provides a little performance enhancement when compared with a less expensive system.
3. **Capacity planning**. This is of great interest to system administrators and managers of data processing installations. This is done to make sure that adequate resources will be available to meet future demands in a cost-effective manner without jeopardizing performance objectives. In some literature, capacity management, which is used to ensure that the available resources are used to provide the optimum performance, is included under capacity planning. In general, capacity planning is performed using the following main steps: (a) instrument the system, (b) observe it, (c) select the workload, (d) forecast the performance under different configurations and alternatives, and (e) select the best cost-effective configuration alternative.
4. **System tuning.** The objective in this case is to find the set of parameter values that produce the best system performance. For example, disk and network buffer sizes can impact the overall performance. Finding the set of best parameters for these resources is a challenge but is important to have the best performance.

5. **Performance debugging**. In some applications, you may come to a situation where the application or control software of the system is working, but it is slow. Therefore, it is essential to discover through performance analysis why the program is not meeting the performance expectation. Once the cause of the problem is identified, the problem can be corrected.
6. **Set expectation**. This is meant to enable system users to set the appropriate expectations for what a system actually can do. This is imperative for the future planning of new generations of routers, switches, and processors.
7. **Recognize relative performance**. The objective in this circumstance is to quantify the change in performance relative to past experience and previous system generations. It can also be to quantify the performance relative to the customer's expectations or to competing systems.

1.4 APPLICATIONS OF PERFORMANCE EVALUATION

The performance evaluation of computer and telecommunication systems is needed for a variety of applications; the major ones are described below [1–3, 5–7]:

- **Design of systems**. It is important that before implementing any system, we conduct a performance evaluation analysis to select the best and most cost-effective design. In general, before designing any new system, one typically has in mind specific architectures, configurations, and performance objectives. Then, all related parameters are chosen to reach the goals. This process entails constructing a model of the system or subsystem at an appropriate level of detail, and this model is evaluated using either analytic modeling or simulation to estimate its performance. It is worth pointing out that analytic modeling may give quick rough results to eliminate inadequate and bad designs; however, simulation would be an effective tool for conducting experiments that can help in making detailed design decisions and avoiding mistakes. Analytic modeling can be used to validate simulation results. In some cases, a prototype version of the system to be designed can be built to make special case validation to simulation and analytic results.
- **System upgrade and tuning**. This process is needed to upgrade or tune the performance of the system or components of the system by either replacing some components with new ones that have better capabilities or by replacing the entire system or subsystem with one depending on the required performance and capacities. The cost, performance, and compatibility dictate the chosen type of system, subsystem, or component, as well as the vendor. In such a case, analytic modeling is used; however, for large and complex systems, simulation is a must. Furthermore, this process may

entail changing resource management policies, such as the buffer allocation scheme, scheduling mechanism, and so on. In applications like these, direct testing and measurement is the best to use; however, it may not be feasible in many situations. Analytic techniques may be attractive, but we may not be able to change the aspects easily. This means that simulation analysis may be the best in such cases, especially if direct experimentation is not possible. Nevertheless, if the goal is just to get a rough estimate or to track the change in output in response to some changes in input parameters, then analytic modeling is a viable option.

- **Procurement.** In this application, the objective is to select the best system from a group of other competing systems. The main criteria are usually the cost, availability, compatibility, and reliability. Direct testing may be the best for such an application, but it may not be practical. Therefore, decisions can be made on some available data with simple modeling.
- **System analysis**. When the system is not performing as it is expected, a performance analysis is conducted to find the bottleneck device or cause of sluggish behavior. The reason for such a poor performance could be either inadequate hardware devices or system management. This means there is a need to identify and locate the problem. If the problem is caused by an inadequate hardware device, then the system has to be upgraded, and if it is caused by poor management, then the system has to be tuned up. In general, the system has to be monitored using hardware, software, or hybrid monitors to examine the behavior of various management schemes under different operating environments and conditions. A measurement technique is usually used in such cases to locate the hardware components or code in question. However, in some cases, simulation and analytic analysis are used, especially if the system is complex.

1.5 TECHNIQUES

Three methods can be used to characterize the performance of computer and telecommunication systems. These are (a) analytic modeling, (b) simulation, and (c) measurement and testing. These alternatives are arranged here in increasing order of cost and accuracy. Analytic models are always approximate: This price must be paid for tractability and obtaining closed-form solution expressions of the performance metrics in terms of design parameters and variables. However, they are usually computationally inexpensive, and expressions can be obtained in a fast manner. Simulations require considerable investment of time in deriving the model, designing and coding the simulator, and verifying and validating the model, but they are more flexible, accurate, and credible. Real measurements and experiments on a variation of a prototype or on the actual system are the most expensive of all and require considerable engineering efforts; however, these measurements are the most accurate. It is important to

note that these three methods complement one another and are used in different phases of the development process of the system [1, 5, 6]. Some of them can be used to validate the results obtained by the others.

In the early stage of the design, when the system designer/architect is searching to find the optimum system configuration, it is impossible to carry out experiments on prototype, and it is time consuming to conduct detailed simulation experiments. During this early stage of the design, the designer is interested in basic performance tradeoffs and in narrowing the range of parameters to be considered. Conducting real-time measurement on a prototype or constructing detailed simulation experiments may be tedious and not cost effective. All that is required at this early stage is approximate calculations to indicate the performance tradeoffs. Analytic performance models provide such an approximate initial quick and rough analysis. It is important to keep in mind that almost all analytic models are approximate. Also, there is often no way to bound tightly the accuracy of such models. That is, one cannot guarantee that the real performance measure is within x% of that predicted by the analytic model, for some finite y%. In most cases, the only way to assess the accuracy of the model is to conduct a few simulation runs and compare the simulation results with the analytic results. Although analytic models are approximate, they are accepted because these models themselves might be used to explore design alternatives, and it is sufficient to have approximate estimates of the expected behavior and performance. If a more accurate performance characterization is required, then the designer must turn to the simulation or measurement on a prototype version of the system, which is more expensive. It is worth noting that the accuracy of an analytic model depends on the quality of input data and on the appropriateness of the chosen performance measure. Regardless of how good the analytic model may be, it cannot give accurate results if the input data are inaccurate or not representative of the workload that the system will be subjected to in the real world. That is to say, collecting representative workload data is crucial for accurate performance modeling [1–7].

1.6 METRICS OF PERFORMANCE

The selection of performance metrics is essential in performance evaluation. These metrics or measures should be selected with the type of application and service in mind, as a performance metric for one application may not be of interest to another application. A good performance metric should have the following characteristics: (a) the performance metric should allow an unambiguous comparison to be made between systems, (b) it should be possible to develop models to estimate the metric, (c) it should be relevant or meaningful, and (d) the model used to estimate the metric should not be difficult to estimate.

In general, performance evaluation analysts are typically interested in the: (a) frequency of occurrence of a specific event, (b) duration of specific time intervals, and (c) size of some parameter [4, 5–7]. In other words, the interest is in count, time, and size measures.

If the system performs the intended service correctly, its performance can be measured by the rate at which the service is performed, the time needed to perform the service, and the resources consumed while performing the service. These are often called productivity, responsiveness, and usage metric/measures, respectively. The productivity of a multiprocessor computer system is measured by its throughput (number of packets or requests processed per unit time) or speedup (how fast the system compared with a single processor system). The responsiveness of the same system is measured by the mean packet delay, which is the mean time needed to process a packet. The utilization metric gives a measure of the percentage of time the resources of the multiprocessor system are busy for a given load level. The resource [usually a processor, but can be a memory or an input/output (I/O) device] with the highest use is called the bottleneck device [1–4].

Performance evaluation metrics of a computer and telecommunication systems can be classified into the following chief categories [1–2]:

- Higher better metrics (HB). In this category, the higher the value of the metric, the better it is. Productivity comes under this category.
- Lower better metrics (LB). Here, the lower the value of the metric, the better it is. Responsiveness is an example of this type.
- Nominal better metrics (NB). In this class, the performance metric should not be too high or too low. A value of usage between 0.5 and 0.75 is desired. Utilization is an example on such metrics.

Other performance measures that are becoming of great interest to performance analysts are availability and reliability. Availability is quantified by two known measures: (a) mean time to failure (MTTF) and (b) mean time between failures (MTBT) [1–3]. Reliability is defined as the probability that the system survives until some time t. If X is time to failure of the system, where X is assumed to be a random variable, then reliability, $R(t)$, can be expressed as $R(t) = P(X > t) = 1 - F(t)$, where $F(t)$ is the distribution function of the system lifetime X [1, 4, 8].

It is important to point out that performance of computer and telecommunication systems from the viewpoint of performance tends to be optimistic as it usually ignores the failure-repair behavior of the system. A new trend these days is to consider the performance, availability, and capacity together. This process is important because in a computer communication network, the failure of a link or router causes partial outage of the network, namely, the decrease in network's capacity that affects the system's quality of service (QoS) as well as its performance [5–8].

1.7 WORKLOAD CHARACTERIZATION AND BENCHMARKING

Regardless of which performance evaluation technique is used, we need to provide input to the model or real system under study. Many new computer and network applications and programming paradigms are constantly emerging. Understanding the characteristics of today's emerging workloads is essential to design efficient and cost-effective architectures for them. It is important to characterize web servers, database systems, transaction processing systems, multimedia, networks, ATM switches, and scientific workloads. It is also useful to design models for workloads. An accurate characterization of application and operation system behavior leads to improved architectures and designs. Analytical modeling of workloads is a challenge and needs to be performed carefully. This is because it takes significant amounts of time to perform trace-driven or execution-driven simulations due to the increased complexity of the processor, memory subsystem, and the workload domain. Quantitative characterization of workloads can help significantly in the creation and validation of analytic models. They can capture the essential features of systems and workloads, which can be helpful in providing early predication about the design. Moreover, quantitative and analytical characterization of workloads is important in understanding and exploiting their interesting features [10–12]. Figure 1.1 depicts an overall block diagram of workload characterization process.

In this context, there are two types of relevant inputs: (a) parameters that can be controlled by the system designer, such as resource allocation buffering technique and scheduling schemes, and (b) input generated by the environments in which the system under study is used such as interarrival times. Such inputs are used to drive the real system if the measurement technique or the simulation model is used. They also can be used to determine adequate distributions for the analytic and simulation models. In the published literature, such inputs are often called workloads.

Workload characterization is considered an important issue in performance evaluation, as it is not always clear what (a) level of detail the workload should have (b) aspects of the workload are significant, and (c) method to be used to represent the workload. In workload characterization, the term "user" may or

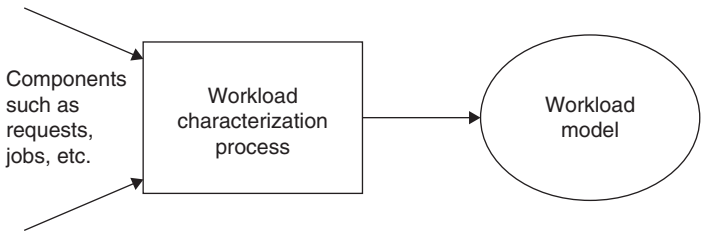

FIGURE 1.1. Overall workload characterization process.

may not be a human being. In most related literature, the term "workload component" or "workload unit" is used instead of user. This means that workload characterization attempts to characterize a typical component. Examples of workload components include (a) applications such as website, e-mail service, or program development (b) sites such as several sites for the same company, and (c) user sessions such as monitoring complete sessions from user login and logout and applications that can be run during such sessions. Measured quantities, requests, and resource demands used to characterize the workload are called parameters. Transaction types include (a) packet sizes, (b) source and destination of packets, and (c) instructions. In general, workload parameters are preferable over system parameters for the characterization of workloads. The parameters of significant impact are included, whereas those of minor impact are usually excluded. Among the techniques that can be used to specify workload are (a) averaging, (b) single-parameter histogram, (c) multiparameter histogram, (d) Markov models. (e) clustering, (f) use of dispersion measures such as coefficient of variation (COV), and (g) principal-component analysis [10–12].

The averaging is the simplest scheme. It relies on presenting a single number that summarizes the parameter values observed, such as arithmetic mean, median/mode/geometric or harmonic means. The arithmetic means may not be appropriate for certain applications. In such cases, the median, mode, geometric means, and harmonic means are used. For example, in the case of addresses in a network, the mean or median is meaningless, therefore, the mode is often chosen.

In the single-parameter histogram scheme, we use histograms to show the relative frequencies of various values of the parameter under consideration. The drawback of using this scheme is that when using individual-parameter histograms, these histograms ignore the correlation among various parameters. To avoid the problem of correlation among different parameters in the single-parameter scheme, the multiparameter scheme is often used. In the latter scheme, a k-dimensional histogram is constructed to describe the distribution of k workload parameters. The difficulty with the same technique is that it is not easy to construct joint histograms for more than two parameters.

Markov models are used in cases when the next request is dependant only on the last request. In general, we can say that if the next state of the system under study depends only on the current state, then the overall systems is behavior follows the Markov model. Markov models are often used in queuing analysis. We can illustrate the model by a transition matrix that gives the values of the probabilities of the next state given present state. Figure 1.2 shows the transition probability matrix for a job's transition in a multiprocessor computer system. Any node in the system can be in one of three possible states: (a) active state where the node (computer) is executing a program (code) using its own cache memory, (b) wait (queued) state where the node waits to access the main memory to read/write data, and (c) access state where the node's request to access the main memory has been granted. The probabilities of going from

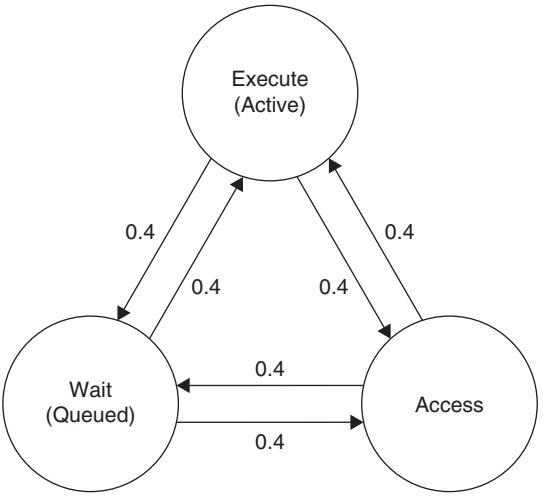

FIGURE 1.2. State transition diagram for the Markov model of the multiprocessor system.

one state to the other make what is called the transition matrix [16]; see Figure 1.2.

The clustering scheme is used when the measured workload is made of a huge number of components. In such a case, these huge components are categorized into a small number of clusters/tiers such that the components in one cluster are as akin to each other as possible. This is almost similar to what is used in clustering in pattern recognition. One class member may be selected from each cluster to be its representative and to conduct the needed study to find out what system design decisions are needed for that cluster/group.

Figure 1.3 shows the number of cells delivered to node A and the numbers delivered to node B in a computer network. As shown in Figure 1.3, the cells can be classified into six groups (clusters) that represent the six different links that they arrive on. Therefore, instead of using 60 cells for each specific analysis, we can use only 6 cells.

The use of dispersion measure can give better information about the variability of the data, as the mean scheme alone is insufficient in cases where the variability in the data set is large. The variability can be quantified using the variance, standard deviation or the COV. In a data set, the variance is given by:

$$\text{Variance} = s^2 = 1/(n-1) \sum_{i=1}^{n} (x_i - x')'$$

and COV = s/x'

where x' is the sample mean with size n. A high COV means high variance, which means in such a case, the mean is not sufficient. A zero COV means that

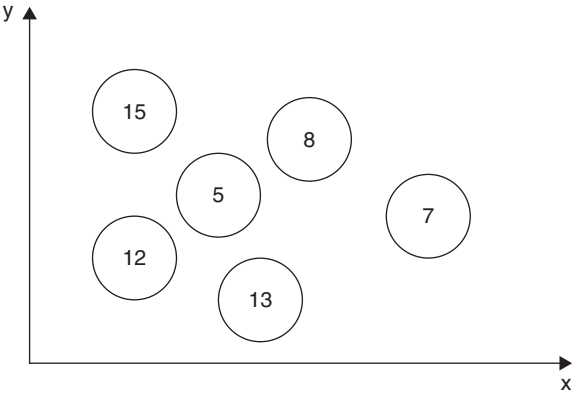

FIGURE 1.3. An example of 60 cells in 6 groups (clusters).

the variance is zero, and in such a case, the mean value gives the same information as the complete data set.

The principal-component analysis is used to categorize workload components using the weighted sum of their parameter values. If d_i is the weight for the i^{th} parameter x_i, then the weighted sum W is as follows:

$$W = \sum_{i=1}^{k} \alpha_i x_i$$

The last expression can be used to group the components into clusters such as low, medium, and high-demand classes [2, 14].

The weights to be used in such cases can be determined using the principal-component analysis that permits finding the weights w_j's such that W_i's provide the maximum discrimination when compared with other components. The value of W_i is called the principal factor or principal component. In general, if we are given a set of k parameters, such as $x1$, $x2$,…., xn, then the principal-component analysis produces a set of factors and $W1$, $W2$, ……, Wk, such that: (a) the W's are linear combinations of x's, (b) the W's form an orthogonal set, which means that their inner product is zero:

Inner Product $= \Sigma W_j \cdot W_j = 0$, and the W's form an ordered set so that $W1$ describes the highest percent of the variance in resource demands, $W2$ describes a lower highest percent, and so forth.

If the system under study is to be used for a specific application, such as airline reservation, online banking, or stock market trade, then representative application programs from these applications or a representative subset of functions for these applications should be used during the performance evaluation study. Usually, benchmark programs are described in terms of the

functions to be performed, and they exercise all resources in the system such as peripherals, databases, networks, and so on.

The term "benchmark" is often used to mean workload or kernel. Benchmarks are usually run by vendors or third parties for typical configurations and workloads. This process should be done with care as it may leave room for misinterpretation and misuse of the measures. Clearly, it is essential to perform this task accurately. A benchmark program is used as a standard reference for comparing performance results using different hardware or different software tools. It is supposed to capture processing and data movement characteristics of a category of application. Benchmarks are meant to measure and predict the performance of systems under study and to reveal their design weakness and strong aspects. A benchmark suite is basically a set of benchmark programs together with a set of specific rules that govern the test conditions and methods such as testbed platform environment, input data, output results, and evaluation metrics (measures). A benchmark family is a set of benchmark suites.

In computer systems, benchmarks can be classified based on the application, such as commercial applications, scientific computing, network services, signal processing, scientific computing, and image processing. Moreover, we can classify benchmarks into microbenchmarks, macrobenchmarks, synthetic benchmark programs, program kernels, and application benchmark programs [9, 13].

A microbenchmark tends to be a synthetic kernel. Microbenchmarks measure a specific portion of the computer system, such as the CPU speed, memory speed, I/O speed, interconnection network, and so on. A small program can be used to test only the processor-memory interface, or the floating-point unit, independent of other components of the system. In general, microbenchmarks are used to characterize the maximum possible performance that could be obtained if the overall system's performance were limited by that single component. Examples on microbenchmarks include [9]:

- LINPAC: This suite measures numerical computing, and it is a collection of Fortran subroutines that analyzes and solves linear equations and linear least-square problems.
- LMBENCH: This suite measures system calls and data movement operation. It is portable and used to measure the operating system overheads and capability of data transfer among the processor, cache, main memory, network, a disk or various Unix platforms.
- STREAM: This simple synthetic benchmark measures sustainable bandwidth of memory and the corresponding computation rate.

A macrobenchmark measures the performance of the system as a whole. Basically, it compares different systems when running a specific application on them. This is of great interest to the system buyer. Keep in mind that this class of benchmarks does not reveal why the system performs well or bad.

Usually, this class of benchmarks is used for parallel computer systems. Examples on macrobenchmark programs include [9, 16–21] the following:

- NPB suite: The Numerical Aerodynamic Simulation (NAS) Parallel Benchmark (NPB) was developed by the (NAS) program at National Aeronautics and Space Administration (NASA) at Ames for performance evaluation of supercomputers. It consists of five kernels: Embarrassing Pascal (EP), multigrid method (MG), conjugate gradient method (CG), fast fourier-based method for solving a three-dimensional (3D) partial differential equation (FT), and Integer Sorting (IS), as well as the simulated applications block lower triangular, block upper triangular (LU), scalar penta-diagonal (SP) and block tri-diagonal (BT) programs.
- PARKBENCH: This was called after the Parallel Kernel and Benchmarks committee. The current benchmarks are for distributed memory multicomputers, coded using Fortran 77 plus Parallel Virtual Machine (PVM) or Message Passing Interface (MPI) for message passing.
- STAP: The Space-Time Adaptive Processing (STAP) benchmark suite is basically a set of real-time, radar signal processing programs originally developed at MIT Lincoln Laboratory. The suite consists of computational-intensive programs that require one to perform 10^{10}-10^{14} FLOPS.
- TPC: This was developed by the Transaction Processing Performance Council. TPC has released five benchmarks: TPC-A, TPC-B, TBC-C, TPC-D, and TPC-E. The first two released benchmarks became obsolete in 1995.
- SPEC: This suite was developed by Standard Performance Evaluation Corporation (SPEC), which is a nonprofit corporation that is made of major vendors. It is becoming the most popular benchmark suite worldwide. SPEC started with benchmarks that measure CPU performance, but now it has suites that measure client-server systems, commercial applications, I/O subsystems, and so on. Among the suites, there are SPECT95, SPEChpc96, SPECweb96, SFS, SDM, GPC, SPEC SFS97, SPECjAp, and SPECjAppServer2001, which is a client/server benchmark for measuring SPEC HPC2002, and SPECviewperf 7.1. SPEC periodically publishes performance results of various systems, both in hard copy and on their website (http://www.spec.org).

  In 2003, SPECapc (SPEC application, performance, characterization) releases the new Solid Edge V12 benchmark, and SPECviewperf 7.1. SPECapc for Solid Edge Version 12 is an updated benchmark based on new features in the software's latest version. The new version increases the graphics and CPU workloads without requiring additional memory. The CPU tests now include a recompute calculation for a part with 500 features and a mass property calculation of the assembly. SPECviewperf 7.1 inserts a small amount of variation at regular intervals within its application-based test files, called viewsets. This ensures that the test system examines and processes each frame individually, as it would in typical real-world applications. For more updated information, visit SPEC website [17].

Despite the problems involved in using the instruction mix to evaluate performance of computer systems, there is still interest in them. An instruction mix is attractive for some analysts in that it abstracts many details of real application programs. An instruction mix is a specification of different instructions coupled with their usage frequency. Examples of an instruction mix include the Gibson mix, which was originally developed by Jack C. Gibson for the IBM 704 system [1, 2].

The program kernel is a generalization of the instruction mix. It is used to characterize the main portion of a specific type of application program. The Kernel benchmark is usually a small program that has been extracted from a large application program. Because the kernel is small, including a dozen lines of code, it should be easy to port it to many different systems. Evaluating the performance of different systems by running such a small kernel can provide an insight into the relative performance of these systems. Because kernels do not exercise memory hierarchy, which is a major bottleneck in most systems, they are of limited value to make a conclusive overall performance comparison or prediction of system performance. Examples of kernels include Puzzle, Tree Searching, Ackermann's Function, and Application benchmark programs are often used when the computer system under evaluation is meant to be used for a specific application, such as an airline reservation or scientific computing. These benchmarks are usually described in terms of the functions to be performed and make use of almost all resources of the system. Keep in mind that application benchmarks are real and complete programs that produce useful results. Collection of such programs is often made on emphasizing one application. To reduce the time needed to run the entire set of programs, they usually use artificial small input data sets, which may limit the application's ability to model memory behavior and I/O requirement accurately? They are considered effective in giving good results. Examples of such benchmark programs include the Debit-Credit benchmark, which is used to compare transaction processing systems [9, 17–21].

Network quality benchmarking services is designed to provide independent examination of network quality and performance. QoS is measured externally based on drive tests, whereas performance is measured internally based on network management system data. Quality benchmarking services for a network is-useful for performance target setting and instant comparison. It is also useful for long-term monitoring. Its main benefits include (a) objective evaluation of network quality, (b) end-user point of view (c) comparison with competitors, and (d) good for long-term network planning.

1.7.1 Case Study: Website Characterization

The phenomenal growth of the World-Wide Web (WWW), in both the volume of information on it and the numbers of users desiring access to it, is dramatically increasing the performance requirements for large-scale information servers.

WWW server performance is a central issue in providing ubiquitous, reliable, and efficient information access [10–12].

It is important that the WWW traffic workload be understood as it is crucial in the analysis of a server's performance. Capturing the main characteristics of such systems, such as the distributions of file sizes and buffering schemes, is vital to provide a quantitative measure of the aggregate overall advantage of a particular server system's optimization. Workload generators that can be used for such systems include SpecWeb96, WebStone, and SURGE [10–12].

In the characterization of a web server, we need to choose parameters that best describe the characteristics of the workload of the servers and system software used, monitor the systems to obtain some raw performance data, analyze performance data, and finally construct a workload model of the system under investigation. Workload characterization allows us to understand the current state of the system under investigation. Characterizing workload is also essential to the design of new system components [11, 12].

In Arlitt and Jin [10] from Hewlett-Packard (HP) have conducted a workload characterization of the of the 1998 World Cup website. Measurements from the World Cup website were collected over a 3-month period, and during this time, the site received 1.35 billion requests, which is considered large, and if not the largest Web workload analyzed to date, it is definitely one of the largest. The authors determined how Web server workloads are evolving. They found that improvement in the caching architecture of the World-Wide Web are changing the workloads of Web servers and that major improvements to that architecture are still necessary.

World Cup 1998 was held in France from June 10 through July 12, 1998. It was commonly called, France '98, and it is considered the most widely covered media event in history. The estimated cumulative television audience is about 40 billion who watched the 64 matches, more than twice the cumulative television audience of the 1996 Summer Olympic Games in Atlanta, Georgia. The URL of France '98 was as follows: www.france98.com. It received more than 1 billion client requests during the tournament [10].

The World Cup tournament is held once every 4 years to determine the best soccer (called football outside the United States) team in the world. This tournament is open to all countries worldwide. Because of the number of participating teams, a qualifying round is usually used to select the teams that will play in the World Cup tournament. The qualifying round for France '98 was held between March 1996 at November 1997 and out of the 172 countries that participated only 30 were selected to compete in France '98, along with the host country, France, and the reigning champions, Brazil. Each match lasted for 90 minutes in length and was played in two 45-minute halves. The website of the 1998 World Cup provided current scores of the matches in real time. Moreover, fans were able to access previous match results; player statistics, player info such as there biographies, ages, and so on; team backgrounds;

information in English and French about stadiums; and local attractions; a wide range of photos and sound clips from the game; and some interviews with players and coaches. Fans were able to download free software, such as World Cup screensavers and wallpapers from the France '98 website [22]. Several companies cooperated to establish the website, which includes: France Telecom, EDS, Hewlett-Packard, and Sybase. Thirty servers were used, and were distributed across four locations: 4 servers in Paris, 10 servers in Herndon, Virginia; 10 servers in Plano, Texas; and 6 servers in Santa Clara, California. The creation and updating of all web pages were done in France. A Cisco Distributed Director was used to distributed client requests across the four locations where various load balancers were used to distribute the incoming requests among the available servers.

Arlitt and Williamson in [10] observed the following main characteristics in the web of the World Cup workload and the performance implications of these characteristics, which include the following:

1. HTTP/1.1 clients that have become more common, accounting for 21% of all requests. Widespread deployment of HTTP/1.1 compliant clients and servers is necessary for the functionality of HTTP/1.1 to be fully exploited.
2. About 88% of all requests were for image files; an additional 10% were for HTML files, signifying that most users interests were in cacheable files.
3. About 19% of all responses were "Not Modified," signifying that cache consistency traffic had a greater impact in the World Cup workload than in previous Web server workloads [11].
4. The workload was rather bursty.
5. For timeouts of 100 seconds or less, many users' sessions contained only a single request and a single response. Arlitt and Wiliamson [10] believed that this is due to improved Web caching architecture that now exists, which has potential implications on both server and protocol design.
6. During periods of peak user interest in World Cup site, the volume of cache consistency traffic increased noticeably.

1.8 SUMMARY

Performance evaluation can be considered both an art and science. This discipline has become more and more important because of the complexity and widespread applications of both computer and telecommunication systems. This chapter aimed to provides an introduction and background information to performance evaluation. We discussed the viewpoint and chief concepts as well as the objectives of performance evaluation. Then we reviewed the main application areas and techniques. Workload characterization and benchmarking were addressed along with examples.

REFERENCES

[1] M. S. Obaidat, and G. I. Papadimitriou (Eds.), "Applied System Simulation: Methodologies and Applications," Springer, New York; 2003.

[2] R. Jain, "The Art of Computer Systems Performance Analysis," Wiley, New York, 1991.

[3] K. Kant, "Introduction to Computer System Performance Evaluation," McGraw-Hill, New York, 1992.

[4] D. J. Lilja, "Measuring Computer Performance," Cambridge University Press, Cambridge, UK, 2000.

[5] M. S. Obaidat, "Advances in Performance Evaluation of Computer and Telecommunications Networking," Computer Communication Journal, Vol. 25, Nos. 11–12, pp. 993–996, 2002.

[6] M. S. Obaidat, "ATM Systems and Networks: Basics Issues, and Performance Modeling and Simulation," Simulation: Transactions of the Society for Modeling and Simulation International, Vol. 78, No. 3, pp. 127–138, 2003.

[7] M. S. Obaidat, "Performance Evaluation of Telecommunication Systems: Models Issues and Applications," Computer Communications Journal, Vol. 34, No. 9, pp. 753–756, 2003.

[8] M. C. Ghanbari, J. Hughes, M. C. Sinclair, and J. P. Eade, "A Principles of Performance Engineering for Telecommunication and Information Systems," IEE, Herts, UK, 1997.

[9] K. Hwang, and Z. Xu, "Scalable Parallel Computing," McGraw-Hill, New York, 1998.

[10] M. Arlitt, and T. Jin, "Workload Characterization of the 1998 World Cup Website," HP Technical Report 1999–35R1, Hewlett-Packard, 1999.

[11] M. Arlitt, and C. Williamson, "Internet Web Servers: Workload Characterization and Performance Implications," Transactions on Networking, Vol. 5, No. 5, pp. 631–645, 1997.

[12] L. John, and A. Maynard, (Eds.), "Workload Characterization of Emerging Applications," Springer, New York, 2003.

[13] J. L. Hennessy, and D. A. Patterson, "Computer Architecture: A Quantitative Approach," Morgan, Kaufmann, 3rd edition, 2003.

[14] J. Banks, J. S. Crason II, B. L. Nelson, and D. Nicol, "Discrete-Event System Simulation," 3rd edition, Prentice Hall, Upper Saddle River, NJ, 2001.

[15] S. M. Ross, "Simulation," 2nd edition, Harcourt Academic Press, San Diego, 1997.

[16] M. S. Obaidat, "Performance Evaluation of the IMPS Multiprocessor System," Journal of Computers and Electric Engineering, Vol. 15, No. 4, pp. 121–130, 1989.

[17] The Standard Performance Evaluation Corporation: http://www.spec.org

[18] http://imls.lib.utexas.edu/redesign/slideshow/tsld009.html

[19] NAS Parallel Benchmarks: http://science.nas.nasa.gov/software/npb/

[20] Parkbench Parallel Benchmarks: http://www.netlib.org/parbench/

[21] Transaction Processing Council (TPC) Benchmarks: http://www.tpc.org/

[22] www.france98.com

EXERCISES

1. Compare and contrast the possible techniques to evaluate a computer or a network system.
2. Visit the website of SPEC and write a report on the new benchmark programs that have been released recently and their applications.
3. For each of the following computer and telecommunications systems, give two performance metrics that can used to assess its performance:
   a. A web sever
   b. WiMax network
   c. A Wi-Fi wireless LAN
   d. A cross-bar–based multiprocessor computer system
   e. An airline reservation system
4. Describe what you think would be the most effective way to evaluate each of the following systems:
   a. A 1000-processor massively parallel computer system
   b. The performance of an ATM-based LAN system
   c. A battlefield-communication system
   d. A cellular network in a large city
5. Explain the role of empirical experimental studies and trace-driven simulation analysis in the performance evaluation of computer and telecommunication systems.
6. To estimate the performance of a multiplexer, the packet arrival should be modeled accurately. Recent empirical studies have shown that the Poisson process is an inaccurate model for the packet arrival process. The statistical structure of the packet arrival process is more complex than assuming it to follow a Poisson process or a finite source models that are often used for modeling call arrivals.

   Explain why this statement is correct. What is the process that is used nowadays to accurately model such an arrival process? Give examples from published literature.