Preface

Performance evaluation in its most general sense is closely related to operations research – both are concerned with decision-making for situations encountered in a wide spectrum of applications. The pressing need to allocate limited resources invokes the use of performance evaluation techniques in engineering, business, economics, and social and environmental sciences, as well as in other aspects of life.

Performance evaluation is a process consisting of a sequence of states, such as problem formulation, parameter selection, model construction, model tuning, result validation and result interpretation. As stated previously, this process is invoked for decision-making purposes; the objective being the identification of a set of variables and the conditions they must satisfy for obtaining the most desirable measure of effectiveness.

Models in general, and analytical models in particular, represent an idealization of the real problem, and approximations and simplifying assumptions usually are required to make the model tractable. Computer system technology is a rapidly evolving and changing field, and performance evaluation in this area has many facets requiring the application of many different techniques.

In general terms we can say that performance evaluation of computer systems deals with the aspects of *performance prediction*, *performance optimization*, and *configuration selection*. In other words, we can say that for certain cost/performance specifications the optimal configuration has to be selected (i.e., hardware, software, and firmware) to support the forecasted workload profiles by providing the required performance (i.e., throughput and response time). If we remember that the operating system is an integral part of the configuration to be selected, it is easy to imagine the complexity involved in the evaluation of such systems. For the sake of illustration we list only a few of the factors that have to be considered; the complexity of the problem becomes self-evident. Factors related to such areas as *I/O Programming* and *Interrupt Programming*, *Processor Management*, *Device Management*, *Information Management*, and *Memory Management* have to be carefully considered and reflected in the models being developed.

The area of *Memory Management* has attracted the widest attention in recent years, triggering a great deal of activity in performance evaluation. Factors to be represented in the models include single contigous allocation vs. partitioned allocation; the whole spectrum of management strategies such as relocatable partitioned memory management, paged memory management, segmented memory management, demand-page memory management, and demand-paged and segmented memory management, as well as swapping and overlays; the impact of replacement algorithms, page sizes, fragmentation and thrashing; storage hierarchies and so on. All those factors have provided the means to implement the virtual memory concept that is found in the large contemporary computer systems.

The importance of memory management in general, and of virtual memory in particular, was the major reason for the selection of the first papers in this topical issue. Although all aspects of memory management are currently of great interest, there is general agreement today that *program behavior* is among the least understood aspects of computer system design and analysis. The subjects treated in the first group of papers in this issue are evidence of the efforts made by computer scientists in the search for better models and better understanding of this aspect.

The papers in this topical issue present the application of existing analytical models – stochastic in nature, in which the underlying ideas are based on queuing theory – as well as the development of new techniques aimed at generalizing some queuing concepts and simplifying others.

The existence of the digital computer and its computational capability have caused us to seek entirely new approaches to formulating mathematical models. The choice of a suitable model, embodying all the properties of a physical system which are critical to its performance, is a difficult task. Complexity *per se* is no longer the great obstacle that it once was. We are not limited anymore to simple mathematical models, easily amenable to paper and pencil study; we can now study problems with hundreds of time-variant functions, along with the secondary effects caused by their dynamic interdependencies, and perform calculations to precisions that exceed our measuring abilities.

Generally speaking, no unifying concept, general theory, or common systematic approach is as yet evident in the field of performance evaluation of computer systems. Yet, one can probably place past and present work in this field into three major groups as a function of the method used in evaluating performance. The three groups consist of 1) *simulation models*, 2) *analytical models*, and 3) *hybrid models* that are a combination of simulation and analytical methods

In the first category, phenomenological simulation, the digital computer has enormously increased the number of parameters that can be represented, as well as the ease with which complex models can be constructed through the use of simulation languages. One should be aware, however, that in addition to the lengthy running time and the difficulty of determining whether the simulation model has reached its steady state before collecting the results, accurate interpretation of the causes and effects embodied in the results may be an extremely difficult task.

Of the second group, analytical models, we have witnessed an increased use in recent years. With the continuous growth in the complexity and sophistication of large computer systems, the need to gain better insight and understanding of the intricate relationships among the parameters has become acute. The best that one can do is to devise analytical models of component subsystems in which the relationships among parameters can be mathematically described within acceptable limits of accuracy and within strict considerations of the strong interdependencies among these subsystems. In other words, the output parameters of certain subsystem models are the input parameters of other subsystem models, and vice versa; a subsystem model cannot be constructed in isolation, and "images" of the attached subsystems must be reflected in the individual mathematical models.

In the development of analytical models we have also seen an increasing acceptance and use of probabilistic concepts, such as the characterization of load profiles by parametric functions, the evaluation of response times and throughputs using queuing models, the determination of system states by applying Markovian models, and the validation of simulation models by experimental design and regression methods. Geometric Programming, Linear Programming, Integer Programming, and other analytical methods are being used in the optimization of computer performance.

In the third group of evaluation techniques are the hybrid models. Generally, closed-form analytical solutions are invariably faster and usually yield a much better insight into the process or system of interest. Situations do exist, however, where simulation is unavoidable. Many questions cannot be answered using a closed analytical form, yet, by using an analytical model, the pertinent relationships and interactions among parameters can be sufficiently well understood to make simulation rather straightforward. On the other hand, there are situations where a closed analytical form does exist but where the range of values of certain parameters cannot be determined *a priori* and a simulation model should be used for determining those ranges. The

accrued advantages of these two are provided by the hybrid models.

We can further differentiate the papers of this topical issue along a different line: The first seven papers describe the *application* of various techniques to the performance evaluation and optimization of computer systems. The subsequent four papers report on the *development* of innovative techniques that make the application of some existing models more amenable to a wider range of problems.

Bard is concerned with the performance of multiprogrammed virtual storage computer systems. He shows in his paper how a system-wide Page Survival Index can be calculated "on the fly" by the operating system and how the value of this index can be used to estimate users' memory requirements and to control system performance by maintaining the proper multiprogramming level.

Bryant discusses algorithms for predicting working-set sizes which provide operating-system schedulers and dispatchers with the capability of estimating the resources required for running programs. The data for his investigation were derived from traces of programs running under CMS on a CP-67 system.

Freiberger et al. use stochastic models to determine the behavior of programs from their reference strings. They indicate that the choice of a correct probabilistic model is far from obvious and that assumptions made in analytical models may prove to be invalid when model predictions are compared with empirical results. The authors show that the notion of regime processes plays a useful role in describing the observed phenomena mathematically.

Ferrari suggests in his paper that, in addition to devising efficient memory management strategies for virtual memory systems in multiprogramming environments, efforts be made to tailor the behavior of programs to the model underlying the storage management strategy, in order to improve system performance. He proposes the use of dynamic off-line restructuring methods to increase the locality of the program.

Schatzoff and Tillman describe the design of experiments in simulator validation as applied to the dispatching algorithm of a time-sharing system. Their basic approach is to view both the real system and the simulator as "black boxes" and to run indentical experiments on the real system and on the simulator to determine whether they have produced statistically comparable effects.

Chiu et al. use a combination of analytical modeling and measurement for the performance analysis of an IBM System/360 Model 75 with OS/MVT and HASP. They use a cyclic queuing model to analyze the system and then they compare the results, for validation purposes, with those obtained from a central-server model. Reconfiguration of the system along the lines suggested by the models has improved the system's performance significantly.

Chang provides stochastic models for the analysis of communication systems. Using queuing models, he obtains an estimate of the total terminal response time, though not its detailed distribution, without which the actual percentile of response time cannot be obtained. The author claims, however, that a reasonable approximation of the percentile value can be obtained by assuming a normal or a gamma distribution for the total response time.

Reiser and Kobayashi describe the use of generating functions to derive closed-form solutions for stability, normalization constant, and marginal distribution for a generalized case of queuing networks in which customer transitions are characterized by more than one closed Markov chain. In their paper the authors show how open and closed subchains interact with each other in such systems, and they derive computational algorithms, applicable to the general class of queuing networks, from the generating function.

Herzog et al. present a recursive method for efficient computational analysis of a wide class of queuing problem. Interarrival and service times are described by multidimensional Markovian processes while arrival and service rates are allowed to be state-dependent. A detailed numerical example is given in the Appendix.

Sauer and Chandy present an approximate analysis of central-server models, providing the means to analyze first-in, first-out queuing models with preemptive or nonpreemptive priorities at a reduced cost and effort. Their techniques are based on Norton's theorem and are used to study and compare a variety of problems.

Chow investigates the stationary behavior of a single-server queue with different classes of jobs. The author assumes that the input process has state-dependent exponential interarrival times and that the service process is nonpreemptive. Like Sauer and Chandy, the author uses a central-server model, but he obtains an exact solution.

The implicit conclusion of the papers in this issue appears to be that, in addition to the application of sophisticated techniques from the fields of mathematics, statistics, operations research, information theory, optimization, and other related disciplines, a good understanding of the interactions among hardware, software, firmware, and load profiles in general, and of the operating system in particular, is an absolute must for the creation of realistic and efficient models.

Ron Ashany
Associate Editor