H. F. Silverman
P. C. Yue

Response Time Characterization of an Information Retrieval System

Abstract: A methodology for computer performance evaluation based on the statistical characterization of response time is described. The results of its application to an information retrieval system are presented. The first part of the paper gives a general discussion of measurement techniques, data reduction procedures and the structure of the system being examined. A set of "system environment" parameters and a set of "job" parameters are then defined and appraised in terms of actual measurements collected over two different weekly periods. Various ways of using the statistical characterization for improving performance are then considered.

1. Introduction

The statistical evaluation of computer performance is of much current interest (see, e.g. [1,2]). This paper presents the methodology as well as the results of a recent study based on measurements made on an information retrieval system running in a teleprocessing environment under a multiprogramming operating system. The discussion is aimed at showing how a limited set of raw measurements can be reduced into a form suitable for statistical analysis by the identification of significant performance factors. The performance criterion chosen is response time, because it is the quantity of prime importance in a query system. Response time is characterized at the transaction level in terms of functional relationships between the response variable and a set of selected parameters. In particular, the concepts of job requirement parameters and system environment parameters are introduced. This provides a framework in which performance before and after system changes can be compared statistically.

Succeeding sections provide a description of the types of measurements available and the ease with which data are gathered, and a functional description of the system structure. Next, the precise definitions and the physical motivations for all the parameters are introduced, followed by a validation of their effectiveness. Finally, an indication of some methods for using the response characterization as an aid for performance enhancement is considered.

2. Data acquisition

A set of software measurement routines record a block of information for each transaction (or inquiry in the case of an information retrieval system). The total of all such blocks is referred to as a *data log*. Since software measurements are event-oriented, they are more appropriate than hardware measurements for studying the interactions among concurrent users because current hardware monitors usually yield utilization factors that cannot be related to individual jobs. Data log routines are reentrant and are built into the system. The degradation of system performance due to logging is kept to a minimum.

A data log generally consists of four types of data:

- Classifications,
- Event counts,
- Time stamps, and
- Cumulative times.

Classification measurements are the simplest. Entries of this type require calculation only once during the lifetime of a transaction. Examples are date, inquiry type, system features requested, user identification, etc. Event counts are also relatively simple to derive. A typical count is of I/O events. Time stamps require a running timer. If a software timer is used, then the cost of measurement is a function of resolution because of required periodic updates. Inquiry start time and finish time are typical time stamps. Cumulative times are the most ex-

pensive to obtain. The data are gathered by taking several time stamps and doing some additional processing. Total CPU time is a typical cumulative time. One should be aware that large overestimates (e.g., of CPU time) may result due to roundoff errors.

3. Data reduction

• *Analysis log*

Raw measured data are first converted into a form suitable for interpretation, called here the *analysis log*. It consists of important items that are directly measured, as well as entries that are calculated from the observed data based on a knowledge of the system. The analysis log discussed below contains one data block per transaction.

• *Directly measured parameters*

The data log is essentially a list of resource demands and resource usage for each transaction. A well-conceived data log, therefore, contains sufficient summary statistics to capture the system load. The more basic direct measurements include

- Inquiry class,
- Time received,
- Time completed,
- Total CPU time used by inquiry,
- Number of I/O events, and
- Memory size used.

In most systems, the load may not be adequately described by these measurements. In such cases, any of the basic measurements may be subdivided so as to provide an improved description. For example, the item called inquiry class might include data about external scheduling priorities, files to be accessed, etc. Certainly, counts of I/O events may be divided into counts of events of different types. For systems in which all inquiries proceed through fixed program code in some sequential manner, such as in an information retrieval system, the life span of each inquiry in the system might be divided into subintervals as determined by segments of the fixed program code. By taking a set of measurements for each subinterval, rather than for one large interval, the time estimates for the occurrence of individual events become more accurate, and, therefore, a more precise characterization results.

• *Derived parameters of system environment*

It is the main purpose of this paper to introduce a characterization of the operating condition of a computing system at the level of detail of a transaction. Therefore, a set of parameters is defined for each transaction in order to account for contention with other users for the same resources. These parameters are referred to collectively as *system environment parameters*.

The set of environment parameters is intended to be an indicator of effects on response time due to

1. Contention among jobs for the CPU,
2. Competition among jobs for I/O channels and devices, and
3. Overhead incurred in being supervised by a software operating system.

It is necessary to find such a set of parameters that are at the same time obtainable from the information in the data log.

• *Derivation of an environmental characterization*

The system used for the present case study is a teleprocessing system built upon Operating System/360 (multiprogrammed with a variable number of tasks) [3] and the Queued Telecommunications Access Method (QTAM) [4]. At a terminal, a user can request information from a single file of the large data base made up of many indexed sequential files stored in disks. Each user request is called an *inquiry*. All program code is reentrant and is used concurrently by all inquiries to the system. No updating of the data by a user is allowed. An error-free inquiry goes sequentially through the stages of translation, index searching, data reading and sorting and/or report building. A feature exists that allows many inquiries from a single user to be handled as a single task. A group of inquiries of this nature is called a *message*.

In this system, both the maximum level of multiprogramming M and the maximum main storage allocated to each inquiry are controlled by the operator. If, therefore, M users are attached when a message arrives, the new arrival must wait. Also, if an inquiry requires more than the maximum allowed storage space, it is placed into a low-priority queue for processing in the background, a message telling the user of this event is issued, and a report is mailed to the user. Neither of these conditions is central to the problem of the evaluation of the system but they are mentioned for the sake of completeness. Only tasks that can fit into storage and can be processed on-line are considered.

The lifetime of a message containing K inquiries is illustrated in Fig. 1. It is necessary to introduce some notation at this point. Suppose the jth message arriving at time $x(j)$ consists of K_j inquiries and is attached at time $A(j)$. If processing of the nth inquiry starts at time $S(n,j)$ and is completed at time $C(n,j)$, then $S(n+1,j) = C(n,j)$ for $n = 1,2, \ldots, K_j - 1$, since the inquiries are processed sequentially without intermediate delay.

The *user response time* for the jth message is defined as

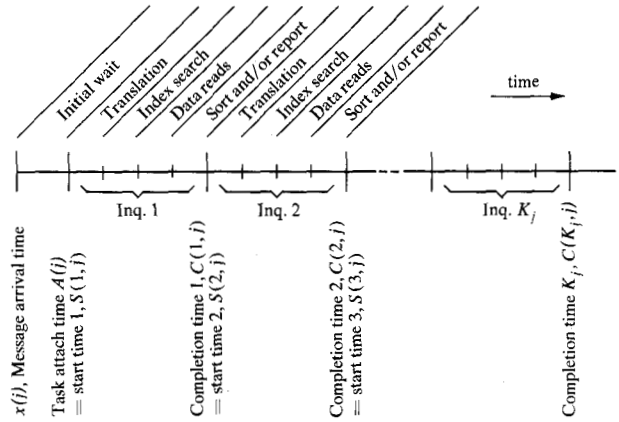

Figure 1 Decomposition of the lifetime of the jth message.

$$URT(j) \equiv C(K_j,j) - x(j). \tag{1}$$

One can decompose (1) into

$$URT(j) = [A(j) - x(j)] + \sum_{n=1}^{K_j} IRT_{n,j}, \tag{2}$$

where each term $IRT_{n,j}$ is called the *inquiry response time* defined as

$$IRT_{n,j} = C(n,j) - S(n,j),\ n = 1,2,\ldots,K_j \tag{3}$$

Alse define a *lifetime function* for each inquiry as follows: For the nth inquiry of the jth message, let

$$LI_{n,j}(t) = \begin{matrix} 1\ S(n,j) \le t \le C(n,j), \\ 0 \text{ otherwise.} \end{matrix} \tag{4}$$

The *average number of attached users* over the lifetime of inquiry (n,j), denoted $\overline{U}_{n,j}$, is given by

$$\overline{U}_{n,j} = \frac{1}{IRT_{n,j}} \sum_{j'=1}^{J} \sum_{n'=1}^{K_{j'}} \left[\int_{S(n,j)}^{C(n,j)} LI_{n'j'}(t)dt \right], \tag{5}$$

where J = total number of messages during one day or one run period for the system.

This is the first environmental parameter. It is an indicator of the overall level of congestion of the inquiry system. This parameter should also indicate the magnitude of processing delays resulting from task switching for CPU service and from channel contention, as each of these delays should be proportional to the degree of multiprogramming.

The second parameter is $\overline{Q}_{n,j}$. Let $q_{n,j}(t)$ denote the *number of attached users having higher dispatching priority than inquiry* (n,j) at time t. Thus

$$q_{n,j}(t) = \sum_{j'=1}^{J} \left\{ 1[A(j) - A(j')] \sum_{n'=1}^{K_{j'}} LI_{n'j'}(t) \right\} \tag{6}$$

where $1(x-y) \equiv \begin{matrix} 1\ x > y \\ 0\ x \le y \end{matrix}$ (the unit step function).

Then $\overline{Q}_{n,j}$ is simply the average value of $q_{n,j}(t)$ over its lifetime, i.e.,

$$\overline{Q}_{n,j} = \frac{1}{IRT_{n,j}} \int_{S(n,j)}^{C(n,j)} q_{n,j}(t)dt. \tag{7}$$

Parameter $\overline{Q}_{n,j}$ is indicative of the time spent waiting for CPU service. The existing scheduler of this system is of the seniority-preemption type [5], and, therefore, a job will preempt the CPU from any other inquiry with lower priority. Priority is assigned in the order of task creation time.

In this particular system, each inquiry (n,j) is allowed to query only a single file, and thus has a file designator, $\phi_{n,j}$, associated with it. The third parameter for characterization of the environment is $\overline{F}_{n,j}$ such that

$$\overline{F}_{n,j} = \frac{1}{IRT_{n,j}} \sum_{\substack{j'=1 \\ j \ne j'}}^{J} \sum_{n'=1}^{K_{j'}} \delta(\phi_{n,j}, \phi_{n'j'}) \int_{S(n,j)}^{C(n,j)} LI_{n'j'}(t)dt, \tag{8}$$

where

$$\delta(x,y) = \begin{matrix} 1 & x = y \\ 0 & x \ne y. \end{matrix}$$

If each file were resident on a single device, then $\overline{F}_{n,j}$, the average number of concurrent users competing with inquiry (n,j) for file $\phi_{n,j}$, would be an accurate indicator of the wait time associated with a secondary storage device.

In the system being studied, however, $\overline{F}_{n,j}$ is a somewhat degraded indicator of device wait time. This is due in part to the fact that one file might be spread over many disks (although almost no disk contains data from more than one file). A second degradating factor results from a lack of adequate time measurements. Only a fraction of the lifetime of an inquiry is spent in the file read mode. It is only the overlap between these periods that should be considered, and not the entire IRT. Thus, $\overline{F}_{n,j}$ derived in Eq. (8) only represents the potential for contention for a file, because only over a fraction of time is there real contention for a device in the physical system. However, as will be shown in the next section, $\overline{F}_{n,j}$ does have some significance in spite of these problems.

4 Data analysis

In this section, actual data are presented to show the effects of job characteristics and system environment parameters on inquiry response time. The data examined were collected over two different weekly periods in January 1971 and October 1971. Between these two periods, the following changes were incorporated into the system: 1) the disk packs were redistributed with the intention of equalizing channel utilization and 2) re-

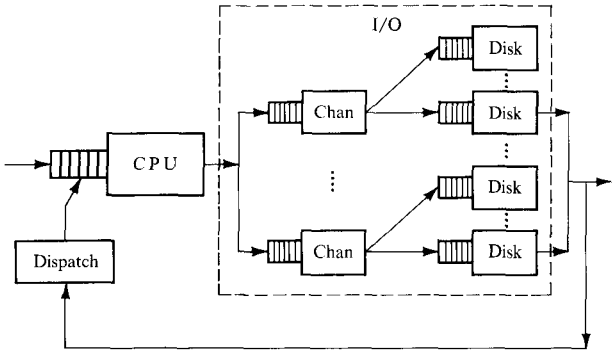

Figure 2 Multiprogramming queue model.

dundant data reads were eliminated whenever the records required had already been brought into main storage by preceding inquiries. The primary concern is whether these changes have resulted in response time improvements, or more appropriately, whether the parameterization introduced in this paper provides a suitable framework for such comparisons.

• *Inquiry response time versus job requirement*

One parameter for characterizing the job requirements of an inquiry is the total number of I/O events, N. This parameter is chosen for two reasons. First of all, N represents the number of times that an inquiry goes through the CPU-I/O cycle in the multiprogramming queuing model for the system under study, as illustrated in Fig. 2. Thus N provides a convenient way of comparing empirical data with queuing theoretic results. Secondly, it is found empirically that the total CPU execution time of each inquiry is, on the average, a monotonic increasing function of N. This fact is illustrated in Fig. 3 for the October data. Therefore, in spite of the variability due to I/O-bound or compute-bound jobs, N is a good indicator of the inquiry's processing requirement.

The average inquiry response time, IRT, is displayed versus N in Fig. 4 (solid curve); as expected, it increases with N. All curves in Figs. 4–6 represent the result of fitting a smooth curve through sample averages. Labelled on each curve are the sample size and the value of CV, where CV = sample standard deviation/sample mean. Higher moments are not within the scope of the present discussion. Two immediate observations can be made: 1) the variance of IRT is quite large, 2) IRT is not proportional to N as one would expect from a first-come, first-served (FCFS) scheduling policy with random access of data; rather, the response curve exhibits a concave shape implying that large jobs get a more favorable response time per I/O event than small jobs. These observations motivate a more careful study of the interactions among inquiries as reflected by the three system environment parameters defined earlier.

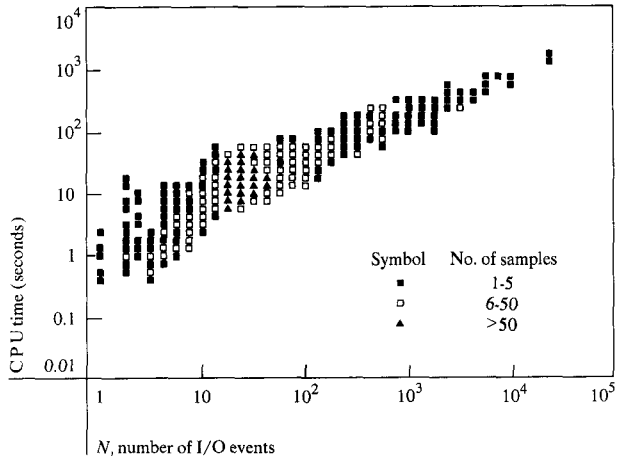

Figure 3 Problem state CPU time vs number of I/O events.

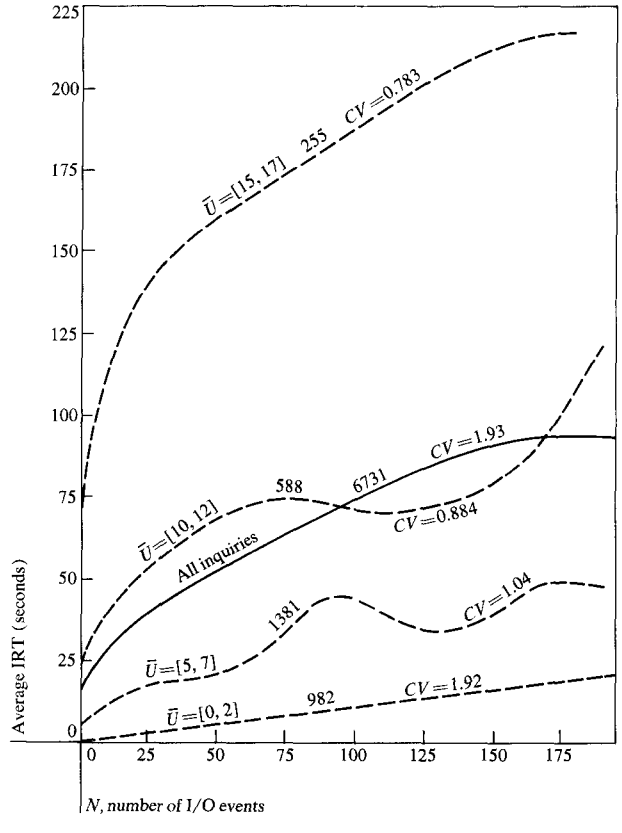

Figure 4 Average IRT vs N and the effects of $\overline{U}$.

• *Inquiry response time versus system congestion, $\overline{U}$*

The environmental parameter first considered is $\overline{U}$. Recall that $\overline{U}_{n,j}$ is the average number of concurrent users in the system during the lifetime of inquiry (n,j). Since the statistical properties of $\overline{U}$ as a parameter are being

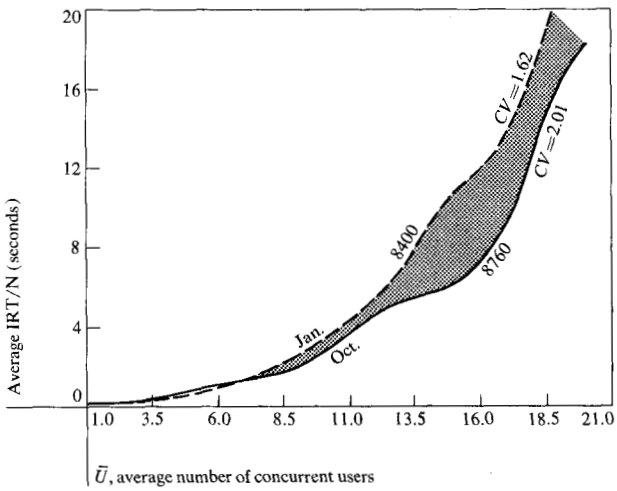

Figure 5 Average value of IRT/N versus $\overline{U}$.

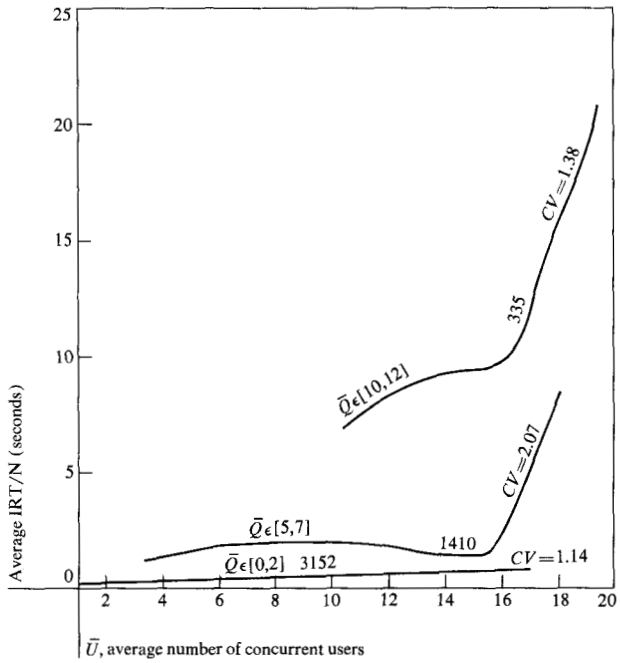

Figure 6 Effects of $\overline{Q}$ on the average values of IRT/N vs $\overline{U}$.

studied, the subscripts (n,j) for individual samples will be suppressed for brevity in the notation $\overline{U}_{n,j}$ (and also in $\overline{Q}_{n,j}$, $\overline{F}_{n,j}$, etc.).

As the system congestion level increases, the average response time per I/O event should increase, since the system resources are limited. This fact is confirmed by the family of curves in Fig. 4 and justifies $\overline{U}$ as an effective measure of congestion. Therefore, the relative performance of the two data collection periods can be compared on the basis of the average IRT per I/O event as a function of $\overline{U}$. This is shown in Fig. 5, where some reduction in response time for October relative to January is apparent in the range between 10 and 17 concurrent users. The data also indicate that the variance in inquiry response time increases as the mean value increases. One might note, as a consequence of the phenomenon displayed in Fig. 4 that jobs with small values of N tend to have longer response times per I/O event than the overall average, and jobs with large N have shorter response times per I/O event. Consequently, one must resist the temptation of interpreting $N.\text{Avg}(IRT/N)$ as an estimate of the inquiry response time for a particular job.

• *Inquiry response time versus dispatching priority, $\overline{Q}$*

The concave relationship between IRT and N suggests that the CPU dispatching priority received by an inquiry might be a significant performance factor (channel and I/O device dispatching are both under the FCFS discipline). Particularly, among the various parameters defined in Section 3, the value of $\overline{Q}$ is the most likely to affect the waiting periods in the CPU service queue and also the service delays due to preemption by jobs with higher priority. This observation is confirmed by Fig. 6 where, for example, when $\overline{U} = 12$, the average response time per I/O event ranges anywhere from 0.8 to 8, corresponding to $\overline{Q} = 1$ and $\overline{Q} = 11$, respectively. The variance is also found to be much smaller for jobs with higher priority, i.e., jobs with smaller values for $\overline{Q}$. Thus, $\overline{Q}$ is useful as a job interaction parameter. We will indicate later how different response characteristics can be enhanced by tailoring the scheduling algorithm to alter the $\overline{Q}$ values for different job classes.

The significance of $\overline{Q}$ on IRT can also be examined with respect to job classes as follows. Suppose $R(N,\overline{Q})$ is the average response time per I/O event as a function of N and $\overline{Q}$; then, for any two values (or ranges) $Q_1 < Q_2$, the quantity

$$\frac{R(N,Q_2) - R(N,Q_1)}{R(N,Q_1)} \times 100\%$$

represents the percentage increase in response time per I/O event as a result of increasing $\overline{Q}$ from Q_1 to Q_2. A typical case is shown in Fig. 7 where Q_1 and Q_2 denote the ranges $0 \leq \overline{Q} \leq 4$ and $4 \leq \overline{Q} \leq 8$, respectively. It may be seen that the value of $\overline{Q}$ affects the small jobs (i.e., jobs with small N) much more significantly than large jobs. For the system under study a more detailed job classification is possible. The I/O events consist mainly of index-search data reads and actual data reads for processing. Denote the number of data reads for index search as N_{IS} and for processing as N_p. A three-dimensional plot analogous to Fig. 7 is given in Fig. 8,

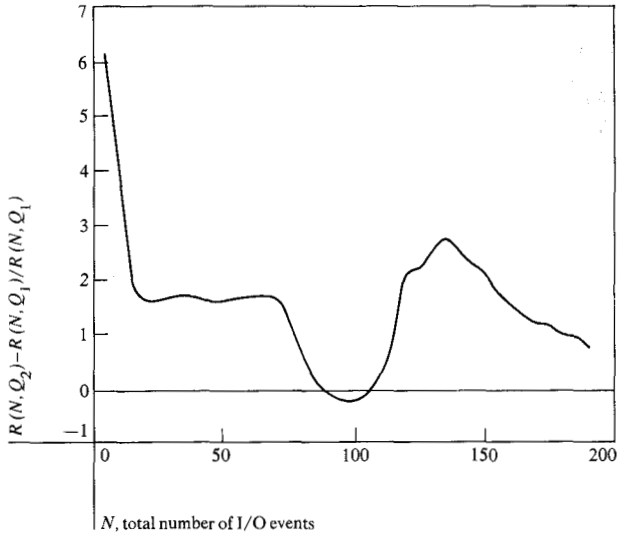

Figure 7 Fractional increase of IRT due to increase in $\overline{Q}$ from [0,4] to [4,8].

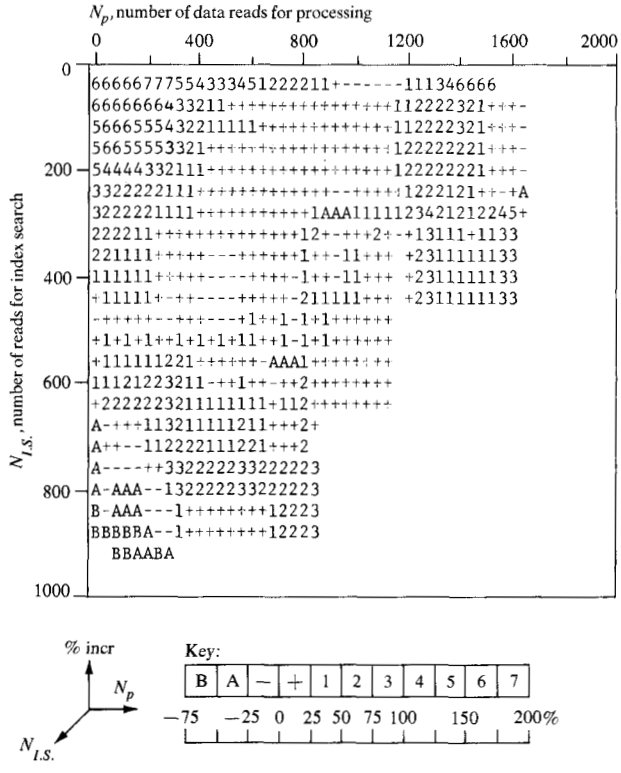

Figure 8 Percentage increase of IRT due to increase in $\overline{Q}$ from [0,4] to [4,8].

and shows the percentage increase in $R(N_{IS},N_P,\overline{Q})$ due to an increase in $\overline{Q}$. The important fact to note is that the directional derivatives with respect to N_{IS} and N_P are approximately equal. Therefore, the aggregate variable $N \approx N_{IS} + N_P$ would suffice to indicate the job requirement for studying the effects of $\overline{Q}$. Close inspection is necessary, however, because in other systems this may not be the case.

• *Inquiry response time versus file contention, $\overline{F}$*

Thus far, two parameters have been introduced to describe the interactions between inquiries. It is fair to say that $\overline{Q}$ mainly accounts for CPU contention as a result of preemptive (priority) scheduling, while $\overline{U}$ accounts for both CPU contention (as a result of interrupts) and channel contention. Neither parameter, however, accounts for I/O device contention. Since the system being studied is a large data base system, it is pertinent to build into the performance evaluation scheme a way of assessing how well the files are organized and how well the subfiles are mapped into I/O devices. Such questions can be answered in relation to inquiry response time. Conceivably, if the accesses to subfiles are traced for each inquiry and the mapping is known, then meaningful device contention factors can be obtained to account for response time variations. The parameter $\overline{F}$, while a compromise due to limitations on available measurements, is nonetheless found to be a meaningful measure. Even without detailed classification with respect to $\overline{U}$ or $\overline{Q}$, the effects of file contention alone on average response time per I/O event are drastically varied for different files, as shown in Fig. 9. By comparing the slopes of these curves, one can judge the sensitivity of particular files to contention. Therefore, the parameter $\overline{F}$ is used most effectively in association with the parameter ϕ, which is the file designator.

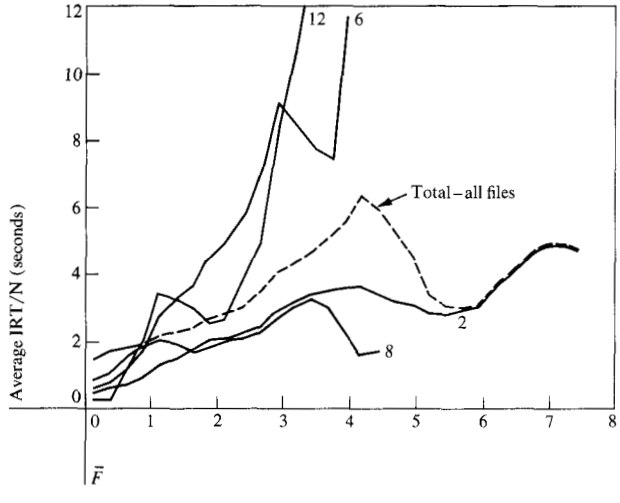

Figure 9 Average value of IRT/N vs File Contention $\overline{F}$ for individual files (January data).

Table 1 Characterization of average inquiry response times (a) October data, (b) January data.

(a)		$\overline{U}$	1–5			5–9			9–13			13–20		
ϕ	$\overline{F}$	N \ $\overline{Q}$	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3
I	< 0.5	S	6	–	–	15	18	–	18	46	44	24	88	100
		M	25	–	–	50	54	–	70	108	106	81	175	200
		L	97	–	–	181	132	–	248	221	146	197	325	317
	≥ 0.5	S	6	–	–	14	21	–	28	60	57	39	95	111
		M	34	–	–	51	74	–	53	117	109	49	175	234
		L	98	–	–	177	140	–	230	236	188	310	383	363
II	< 0.5	S	6	–	–	16	22	–	17	49	50	67	73	89
		M	19	–	–	29	45	–	48	93	73	44	207	195
		L	65	–	–	92	–	–	211	195	–	87	332	–
	≥ 0.5	S	–	–	–	10	26	–	–	11	26	–	–	118
		M	–	–	–	–	96	–	–	–	–	–	110	205
		L	–	–	–	–	–	–	–	–	–	–	–	–

(b)		$\overline{U}$	1–5			5–9			9–13			13–20		
ϕ	$\overline{F}$	N \ $\overline{Q}$	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3
I	< 0.5	S	5	–	–	9	15	–	19	48	49	26	112	118
		M	20	–	–	37	42	–	62	114	123	104	225	189
		L	92	–	–	141	128	–	202	251	–	273	374	206
	≥ 0.5	S	7	–	–	13	21	–	21	60	67	33	118	139
		M	27	–	–	44	56	–	54	126	84	68	212	226
		L	130	–	–	214	149	–	260	263	152	330	407	495
II	< 0.5	S	4	–	–	7	13	–	13	33	39	11	28	99
		M	18	–	–	34	38	–	58	90	73	54	233	186
		L	62	–	–	84	96	–	91	259	–	141	365	–
	≥ 0.5	S	–	–	–	17	10	–	–	101	23	8	173	–
		M	26	–	–	35	–	–	–	126	127	77	180	136
		L	–	–	–	132	–	–	298	–	–	82	218	–

S: $0 \leq N < 40$ Q_1: $0 \leq \overline{Q} \leq 5$ I: $\phi = 2,3,\ldots,14$

M: $40 \leq N < 400$ Q_2: $5 < \overline{Q} \leq 9$ II: $\phi = 1,15,16,17,18$

L: $400 \leq N < 4000$ Q_3: $9 < \overline{Q} \leq 20$

5. Results

• *Methodology for relative performance evaluation*

The above discussions have, in essence, led to a characterization of the inquiry response time as a function of $(N,\overline{U},\overline{Q},\overline{F},\phi)$. This quintuple is a set of summary statistics, which can be easily derived from the inquiry log. It is minimal because a smaller set would omit pertinent information about distinct aspects of performance. This set is, of course, not necessarily complete because completeness depends on the desired emphasis. An example will be given of how one can extend the set for specific needs.

Employing this characterization, one can now evaluate the relative performance for the two data periods. The results are presented in the form of a 5-way classification table. No attempt is made here to fit specific models of response time. In Table 1, the mean values of *IRT* for October and January, 1971, are computed for each class (or cell). Those cells underscored by a solid line indicate significant improvements in October relative to January, i.e., reductions in response time at the 90% confidence level (assuming a normal distribution for sample averages, which is reasonable when the sample size is large). On the other hand, degradation is underscored by a dotted line.

Since every operational system goes through an evolutionary process of numerous software and hardware changes (or tuning), relative performance becomes important. The approach presented here not only evalu-

ates whether improvements have been made, but more importantly, it indicates for what types of jobs and under what conditions improvements have been made. In the present case study, Table 1 suggests that the changes implemented between the two periods have not really altered the system's behavior dramatically. It appears, however, that in October the system performs better under heavy congestion, relative to $\overline{U}$, and performance is somewhat degraded when the load is small. No other obvious patterns are detected. Note that Table 1 only gives results on the means, but the approach is also valid for studying variance and higher moments.

The effectiveness of classification in Table 1a and b is determined as follows. Suppose the parameter set used for classification is ξ and $R(\xi) = [IRT|\xi]$. Then, for each inquiry,

$$R(\xi) = IRT(1 + \varepsilon_\xi), \tag{9}$$

and the mean-squared value of the classification error ε_ξ is readily computed. The error is defined in this manner because it is observed that the standard deviation of $(R(\xi) - IRT)$ is not constant, but rather, is proportional to IRT. The percentage reduction of classification error due to the addition of a particular parameter ξ_i to the set ξ is given by

$$r(\xi_i|\xi) = \frac{[\varepsilon_\xi^2] - [\varepsilon^2_{\xi_i \cup \xi}]}{[\varepsilon_\xi^2]} \times 100\% \tag{10}$$

Numerical values of these reductions, given in Table 2, again validate the usefulness of the selected parameter set.

• *Calibration of analytic and simulation models*

Analytic and simulation models are often used as tools for system design and tuning. These models usually require knowledge of service time distributions. In the case of queuing models, if all the distributions are parameterized and have a simple form, estimators such as maximum likelihood can be sought. In doing so, there are two basic problems: 1) closed-form expressions for the likelihood function can be obtained only for special cases (see, e.g. [6]) and are usually difficult to maximize; 2) functional blocks in the models may well be conceptual idealizations and are not always physically identifiable for direct measurements (even ignoring the questions of measurement availability and economy). However, the calibration of both types of models depends heavily on the successful choice of an optimality criterion for "goodness-of-fit." The characterization in this paper can be used for that purpose, e.g., by choosing as a criterion $J(\theta)$, the mean-squared error between the inquiry response times of the model and the real system being measured, both as functions of the parameters $(N,\overline{U},\overline{Q},\overline{F},\phi)$,

Table 2 Percentage reduction of classification variance due to each additional parameter.

Additional parameter	*% of variance reduction*
$(N\|\text{nil})$	82.0
$(\overline{U}\|N)$	56.5
$(\overline{Q}\|N,\overline{U})$	40.0
$(\overline{F},\phi\|N,\overline{U},\overline{Q})$	20.0

Table 3 Sensitivity of inquiry response time to file contention.

File no.	α	β	*Relative access freq.*
1	0.608*	2404	180
2	0.179	2457	342
3	0.166	2538	563
4	0.127	2375	323
6	0.151	2367	876
7	0.189	2440	408
8	0.099	2183	513
12	0.190	2417	342
13	−0.103	2614	106
14	0.019	2598	32
15	1.999*	2512	78
16	0.944*	2719	77
17	0.450*	2382	95

$$J(\theta) = \int [R_{\text{MOD}}(N,\overline{U},\overline{Q},\overline{F},\phi;\theta) - R(N,\overline{U},\overline{Q},\overline{F},\phi)]^2 dP(N,\overline{U},\overline{Q},\overline{F},\phi), \tag{11}$$

where θ is the unknown parameter vector for the model. A simple way of computing this function is to discretize the variables as in Table 1 and then use the empirical joint distribution for P in (11), thus reducing the integral to a finite multiple sum.

• *Performance enhancement*

There are two ways to use the proposed characterization for design purposes. The first one exploits the dependence of IRT on $(\overline{F},\phi)$. For example, a regression model can be fitted to the data for each file, ϕ, in the form:

$$IRT = a(\phi)\overline{F} + b(\phi) + \delta' \tag{12}$$

or alternatively,

$$\log IRT = \alpha(\phi)\overline{F} + \beta(\phi) + \delta \tag{13}$$

if the curves in Fig. 9 are interpreted as exponential rather than linear. The magnitude of the regression coefficient $a(\phi)$ or $\alpha(\phi)$ enables us to pinpoint the files for which contention affects response times most significantly. Once these sensitive files are detected, there are

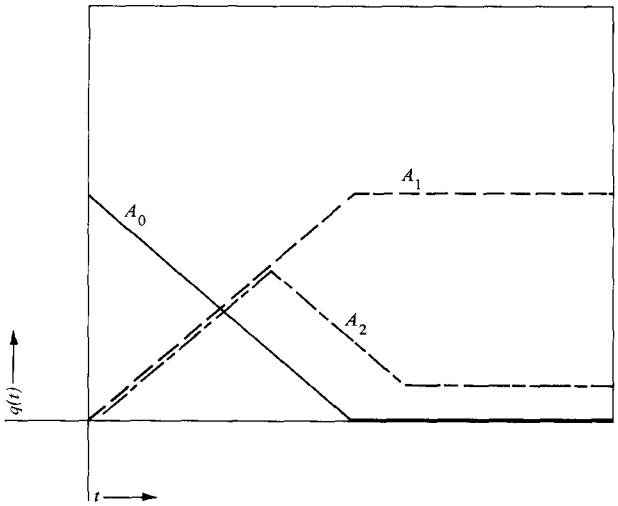

Figure 10 Dynamic priority assigned as a function of time.

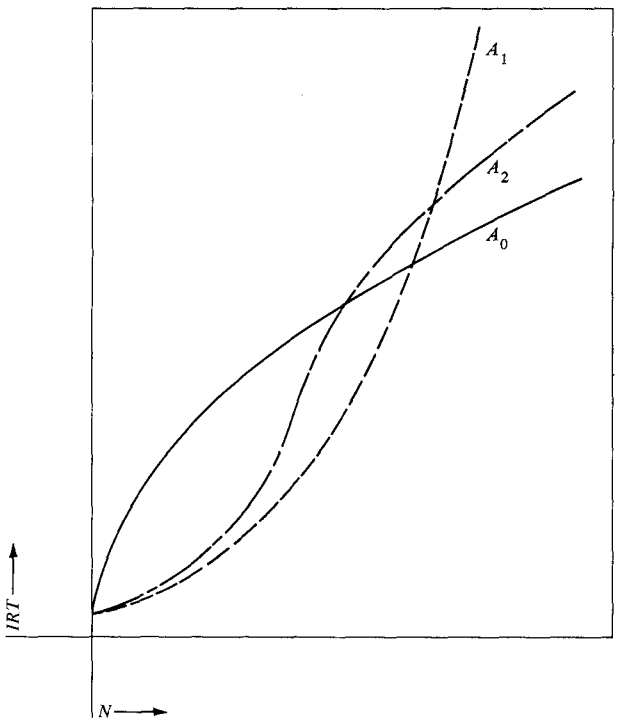

Figure 11 Response curves under various CPU priority assignment policies.

many ways to redistribute disk packs, relocate records, duplicate records or restructure the file itself. Table 3 lists the coefficients $\alpha(\phi)$ and $\beta(\phi)$ for the October data using Eq. (13), where the sensitive files are marked by an asterisk. Similar results using (12) have also been obtained but are not presented.

Another application is the empirical derivation of alternative CPU dispatching priority assignment policies based upon observations of system behavior under the existing scheme. This will be demonstrated by three policies denoted A_0, A_1 and A_2.

Policy A_0 is the existing policy. Suppose the average number of users in the system is u_0 and the message completion rate is γ. Then the mean value of dynamic priority index assigned to the average job as a function of time is approximately given by $q(t) = \max(u_0 - \gamma.t, 0)$, where t is the time from job start. Using $\overline{Q}$ as an intermediate parameter, one can exhibit how different response characteristics can result as follows. First, the relation

$$\overline{Q} = \frac{1}{IRT} \int_0^{IRT} q(t)dt \tag{14}$$

gives an estimate of $\overline{Q}$ for any given value of IRT. Then the measurement data yields an estimate of the number of I/O operations that can be serviced for these values of IRT and $\overline{Q}$. Figure 10 shows $q(t)$ versus t and Fig. 11 shows IRT versus N. The response curve for A_0, IRT vs N, has already been explained in Section 4.

Response curves of a different shape can be synthesized by imposing a new $q(t)$. Suppose the policy is A_1, which assigns the highest priority to newly arrived tasks (one such algorithm is suggested in [5]). Then the priority of a particular job is lowered by one at each new arrival, i.e., the value of $q(t)$ is increased by one. At the same time, $q(t)$ is decreased by one at every completion of these newcomers. The mean value of the dynamic priority index becomes approximately $q(t) = \min(\delta.t, u_0)$ where δ is the message arrival rate. Thus, the response curve for A_1 can be obtained in the same way and is shown in Fig. 11.

There is yet another possibility, A_2, which employs first A_1 and then switches to A_0 at some threshold point. Policy A_2 can speed up the processing of small jobs and at the same time prevent the pitfall of holding large jobs too long. This is illustrated in Fig. 11 by the dotted line. Therefore, the characterization provides a heuristic procedure for choosing among design alternatives by a comparison of their expected response curves.

• *Extension*

An example is now given to show how the parameter set can be extended for specific needs. Suppose the overhead due to supplying characters to terminals is of particular interest (say, when a change of output scheduling is being considered). Output processing interacts with inquiry execution in the form of I/O interrupts. (In the system being considered, all outputs to the user's terminal for a message are transmitted only after processing has been completed of all inquiries of

that message, i.e., in a period following $C(K_k j)$, and there is no delay because I/O interrupts have top priority for CPU service.) Therefore, in addition to using $\overline{U}$, a new parameter $\overline{L}$—the average number of users transmitting output over the lifetime of an inquiry—is introduced. Specifically if m_j is the number of characters sent by message j, and λ is the terminal output rate in characters per second, then the new parameter is defined by

$$\overline{L}_{n,j} = \frac{1}{IRT_{n,j}} \sum_{j'=1}^{J} \int_{S(n,j)}^{C(n,j)} LO_{j'}(t)\,dt,$$

where

$$LO_j(t) = \begin{cases} 1 & 0 \le t - C(K_j j) \le m_j/\lambda, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, $\overline{L}$ may be used to obtain information about that portion of CPU service delay that is a consequence of outputs.

Conclusions

In this paper, we have introduced a technique for the analysis and evaluation of computer systems based on measurements and applied it to a case study. A minimal set of parameters for describing job requirements and system environment is proposed and examined in terms of the effects on inquiry response time. The choice of this set is consciously influenced by both queuing models and the structural aspects of operating systems. A specific way of extending this parameter set is also indicated. The statistical characterization yields 1) a unified methodology for assessing relative performance, 2) a criterion for model calibration, 3) a sensitivity measure of file contention, and 4) a synthesis procedure for the CPU dispatching policy.

Acknowledgment

The authors are indebted to C. Cooper, R. Williams, and V. Wino for their cooperation in providing data and information. Many fruitful discussions with H. Kobayashi are gratefully acknowledged.

References

1. *Proceedings of Workshop on System Performance Evaluation*, ACM/SIGOPS, Association for Computing Machinery, New York, 1972.
2. *Proceedings of Conference on Statistical Methods for the Evaluation of Computer System Performance*, W. Frieberger, Ed., Academic Press, New York, 1972.
3. *IBM System/360 Operating System MVT Guide*, Form GC28-6720-2, IBM Data Processing Division, White Plains, N.Y.
4. *IBM Operating System/360 QTAM User's Guide*, Form C20-1640, IBM Data Processing Division, White Plains, N.Y.
5. H. Kobayashi and H. F. Silverman, *Some Dispatching Priority Schemes and Their Effects on Response Time Distribution—Part 1, IBM Research Report RC-3584*, Yorktown Heights, New York.
6. D. R. Cox, "Some Problems of Statistical Analysis Connected With Congestion," *Proc. Symposium on Congestion Theory*, University of North Carolina Press, Chapel Hill, North Carolina, 1965, pp. 289–316.

Received January 5, 1973
Revised May 9, 1973

The authors are located at the IBM T. J. Watson Research Center at Yorktown Heights, New York 10598.