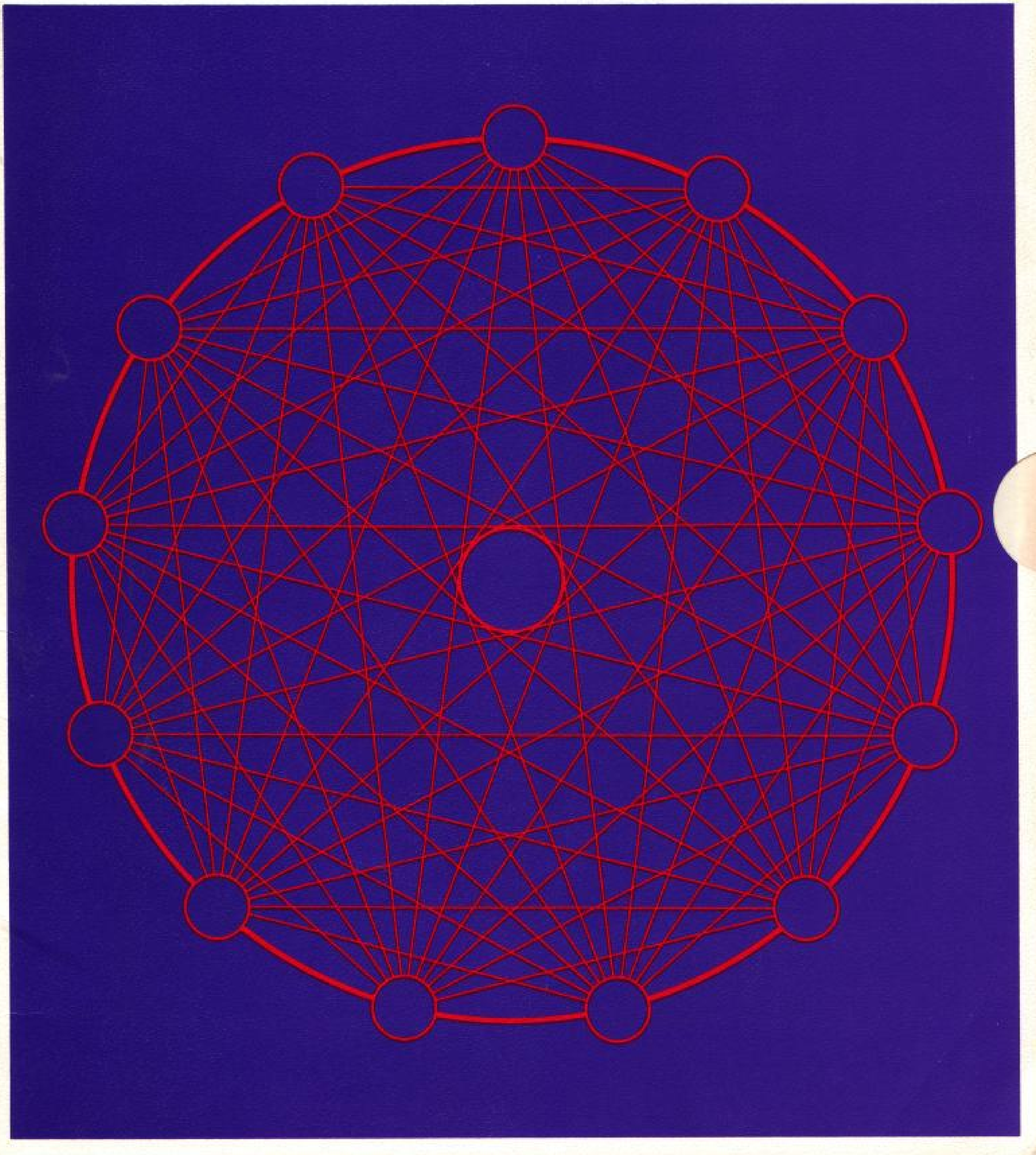
technical review

western union

autumn 1969 vol. 23 no. 5

INDEX



technical review western union

The purpose of the Technical Review is to present technological advances and their applications to communications. This technical publication is published by Public Relations for management, supervisory and technical personnel in Western Union. It is issued quarterly in Spring, Summer, Autumn and Winter. Occasionally, special issues are added to the four regular issues.

committee on technical publications

MARY C. KILLILEA	Chairman
M. BORSELLA	J. R. SKELTON
J. V. BURKE	C. G. SMITH
E. H. GIBBONS	C. R. TRENDOWSKI
J. A. PENCE	R. B. ZINGRE

editor	MARY C. KILLILEA
photography	WILLARD RALSTON
artist	JOHN J. WITT

cover

A fully connected network of communication centers representing a bounding condition on communication network geometry.

subscriptions

New Subscriptions—$10.00 per year including postage—(4 issues)

Back issues—$5.00 per copy (minimum order 2 copies)

address

EDITOR
TECHNICAL REVIEW
WESTERN UNION
60 HUDSON STREET
N.Y., N.Y. 10013

autumn 1969 **vol. 23 no. 5**

technical review western union

60 HUDSON STREET, NEW YORK, N.Y. 10013

NEW YORK NEW YORK

FIRST CLASS MAIL
U. S. POSTAGE
PAID 36 CENTS
PERMIT NO. 697

James H. Haynes
Computer Center
Univ. of California
Santa Cruz, Calif. 95060

western union
TECHNICAL REVIEW
AUTUMN 1969 VOL. 23 NO. 5

CONTENTS

An Interview

with

Mr. Finney

Editor's note:
Mr. R. G. Finney, Assistant Vice President of Shared-Information Systems, in the Planning and Engineering Operation is responsible for the design and implementation of message switching systems including ISCS and SICOM. The Editor asked Mr. Finney to update the status of the ISCS for the readers of the TECHNICAL REVIEW. The following questions were answered by Mr. Finney to indicate the forward look of Western Union in computerized communications technology.

What is the current status of ISCS?

ISCS has just recently achieved a major milestone, the netting of all five computer sites on September 20, 1969.

The system is built around Univac 418 processing and communication centers in San Francisco, Chicago, Atlanta, plus two sites in New York. The first New York site went into operation offering the service, TCCS in January, 1968. INFO-COM was introduced in Chicago in September, 1968. The first netting was accomplished in May, 1969, with the cutover of San Francisco and the second New York site, and their netting with the first New York site. The final site, Atlanta, was cutover and netted with New York and San Francisco on September 15, 1969. The final step was the netting of Chicago with the other sites on September 20th. I might add that all these steps were accomplished without disturbance to those subscribers of INFO-COM and TCCS already using the system, and we are particularly proud that this five-site network was achieved one month ahead of the schedule promulgated in the Fall of 1968.

You may be interested to know one more major package of hardware and software performance enhancements will be placed in operation next March 1970. At that point, design and development will be essentially complete.

What do you mean by the netting of the ISCS?

"Netting" means that all computers are directly connected to each other through high-speed 2400 bits per second lines. This allows simultaneous transmission of large numbers of messages with rapid access to and from terminals anywhere in the country, all under direct and automatic computer control.

This computer network allows us to offer higher capacity and improved service while minimizing our costs. It also gives us the technical and operational base for offering new and expanded services to our customers in the future. We envision in the future that a subscriber with one simple connection to one Western Union computer can have access to a variety of services, automatically supplied and routed through this computer network.

The articles in this issue of the Technical Review are concerned with various aspects of system design and analysis. What is the significance of this type of activity at this relatively mature stage of the system?

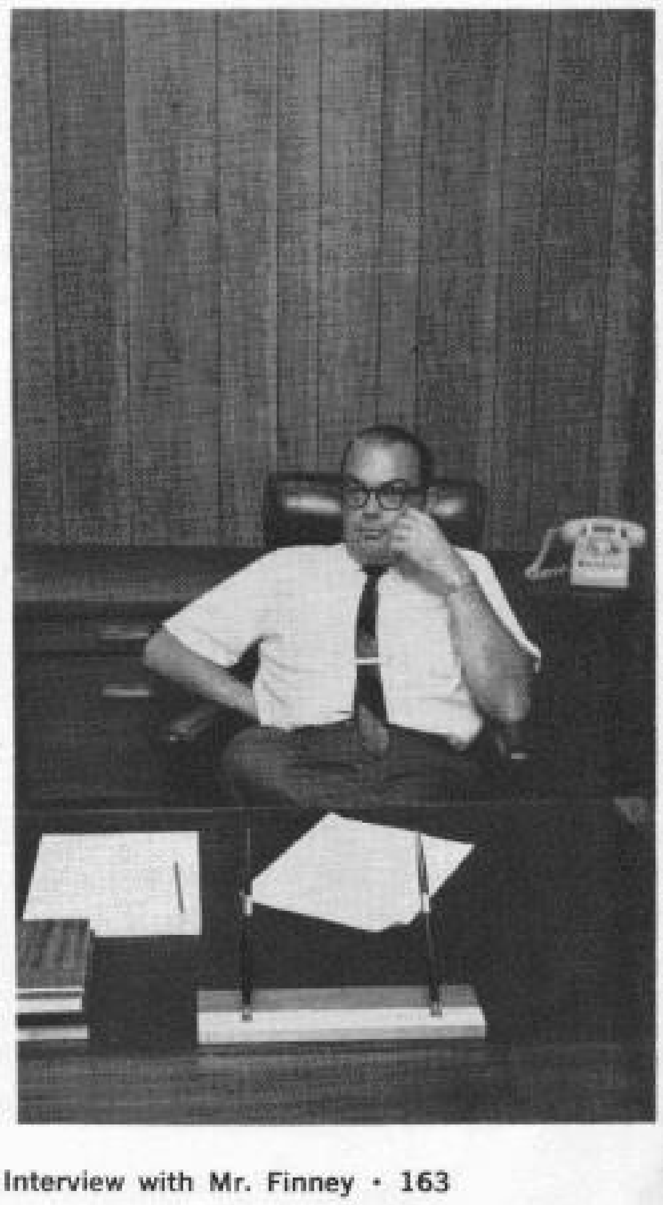
There is, of course, a continuing need in any system to evaluate actual operational performance and to compare this against the predicted performance of the initial design stage. In ISCS, this need is particularly pronounced. ISCS is a "shared" system—i.e., it is used or shared by many different Telex and INFO-COM subscribers. These subscribers come on the system one at a time as the service is sold to them individually, resulting in a steady traffic build-up over a fairly long period of time. At present, no site is handling more than 40% of capacity, with full load expected to be reached in late 1970. In contrast, a "dedicated" (single customer) system is normally loaded at initial cutover with its full population of terminals, and the initial traffic load can be close to design capacity.

The result is that there is no straightforward way to test a shared system at its full capacity in early stages of its life. It would be prohibitively expensive to implement a full terminal capacity for testing purposes. Therefore, extensive and creative means of analysis and simulation must be developed to obtain experimental results from which valid conclusions can be drawn of system behavior at capacity operation.

Another important aspect of a shared system with its many individual users is the unpredictability of many of the fundamental usage parameters upon which the system design must be based. This means that in the initial design, some basic characteristics could be predicted only statistically, without much precision. Now with operational experience, we must determine the actual usage and compensate for it, if necessary.

As a simple example, a basic design parameter in a message switching system such as ISCS is the expected distribution of message lengths. Software structure, storage requirements and communication interfacing are all partially dependent upon this parameter, as are throughput and capacity performance. Now, subscribers in actuality might choose to send much longer messages on the average than were predicted in initial design. If so, either design changes must be made to regain predicted performance, or the performance resulting from actual usage characteristics must be accepted. In either case, detailed analysis must be done to obtain the required knowledge.

Since many of the parameters and interrelationships are quite sophisticated, the analyses must be extensive and sophisticated.



What is involved in doing such analysis?

Generally, the process moves through theoretical analysis, verification by experimental data and analysis, and extrapolation of the results to general conclusions which will be valid at capacity operation. The principal problem now is obtaining valid data for this verification and extrapolation. There are two sources of data, the operating system itself and the development laboratory.

The actual operating system obviously yields true, real-world data, and this is extensively used for analysis. Numerous statistics are automatically generated and are available at periodic intervals or upon request. This allows quick construction of time profiles of total traffic, intercept load, queue sizes and the like. We have computer programs available to obtain more sophisticated data—the daily system journals (records) are run through these analysis programs to yield information on message types, speed of service, message lengths, and the like. Finally, the old-fashioned method of poring over pieces of paper, such as message copy, is still the best and/or only way for some problems, such as obtaining a detailed understanding of customer input errors.

The operating system yields data only at current traffic levels. The real challenge is to obtain data valid for prediction of capacity performance, and this is where the development laboratory comes in. This laboratory can be configured to simulate a single site or a partial network, plus a reasonable sample of all types of subscriber terminals. It also provides computer processing power adequate for any of our engineering analyses or simulations.

We have developed and programmed powerful simulators which allow us, totally within the lab, to simulate a capacity traffic environment. Real input from numerous actual terminals can be combined with this simulation to give a high degree of flexibility in analysis of various types of traffic, messages, errors, etc. Computers next to each other in the laboratory can be connected by high speed lines which loop to San Francisco and back, if desired for analysis of coast-to-coast network operation. In addition to performance analysis, these simulators are principal tools in the debugging and testing of new software in a high traffic environment, and for the re-creation and solution of operational problems. They are also valuable operational training aids.

The articles in this issue discuss examples of the various analyses and simulations by which, using the development laboratory and operational data, capacity performance of the system is predicted.

And what about SICOM?

The SICOM (Securities Industry COMmunication) system is the company's other operational shared system. As you know, it offers a service similar to Info-COM, but specialized for the needs of the stock brokerage industry—transmittal of securities orders, executions, and, related traffic between brokers' branch offices and the floor of the stock exchanges.

SICOM was cutover into operation in June, 1968, and has been very successful. This computer system has consistently exhibited good performance, speed and reliability. Design and development work are complete, and SICOM is already a mature system. With its good reception, capacity additions will be required next year.

What is the significance to Western Union of the activity you have described?

First, technology and economics dictate that the computer must be an essential element in the transfer, control and processing of information as we move into the future. Information transfer is of course Western Union's main business, so expertise in computer technology and operation—or more precisely, computerized communication technology and operation—will grow increasingly more important to the company.

Second, this large-scale shared system approach in a commercial environment is breaking new ground, as did AUTODIN and the Advanced Record System in the government sector, and the various private commercial systems implemented by the Company. Western Union need take a back seat to no one in this computerized communication technology.

The combination of this importance with corresponding leadership/competence is encouraging for the future. The Company has a good start toward the systems and services of the 1970's.

MEASUREMENT OF ISCS THRUPUT

by T. M. DERLINGA

Thruput, broadly stated, is a measure of the primary performance of a message switching system; it is a measure of the number of transactions which can be handled per unit time by the system. This number is dependent upon the processing time required per transaction, upon the in-process storage requirements, and upon the input/output channel utilization and the message (bulk) storage requirements. Over much of the system's normal operating region, thruput is roughly independent of message delay. An optimized system will attain its maximum thruput when, under conditions which cause its limits to be invoked, all limits are reached simultaneously at some pre-specified message delay time. System specifications usually include allowable delay and thereby set the maximum level of system activity or thruput.

Recently, ISCS was tested for performance and thruput margins were established. The process involves testing to as high a level as possible, observing the limit and then establishing whether this limit is structural or allotmental in nature. A structural limit is time related and, as such, requires recoding, new coding, or faster hardware. Such limits would consist of the available processing time per message resulting from I/O and CPU utilizations, or on instruction sensitive table sizes. An allotmental limit is storage capacity related and is defined as one where only expansion is needed. Examples of the latter category include: the number of active lines that are specified to be possible at any one point in time, the amount of storage set aside for in-transit message processing, etc. Usually, the allotmental limits are first observed and when they are isolated, they are expanded and the process repeated until the structural limit is reached. In the process of establishing the thruput margin many system sensitivities are recovered and the formulation of the system model accomplished.

The demonstration of thruput margin provides in a sense a somewhat optimistic view of the system's expected behavior under volume. During this demonstration, the multi-varied service requests and error conditions which normally occur in an operational environment are held at a minimum in order to permit system time to be used most efficiently for message processing. The additional requirements of message assurance, message liability, service and functional requirements, and system stability are necessary ingredients for establishing system acceptance, but are not of interest herein except insofar as these attributes concern the demonstration of thruput margin.

For the ISCS system, the performance measure of thruput was taken as its optimized or maximum capability defined in terms of a long time average traffic handling capacity without significant message backup within the system. Mathematically, the ISCS measure of thruput may be presented as:

$$\lambda_{\text{thru}} = \frac{\lambda_{\text{in}} + \lambda_{\text{out}}}{2} \qquad (1)$$

where $0.9\lambda_{\text{in}} \leq \lambda_{\text{out}} \leq 1.1\lambda_{\text{in}}$

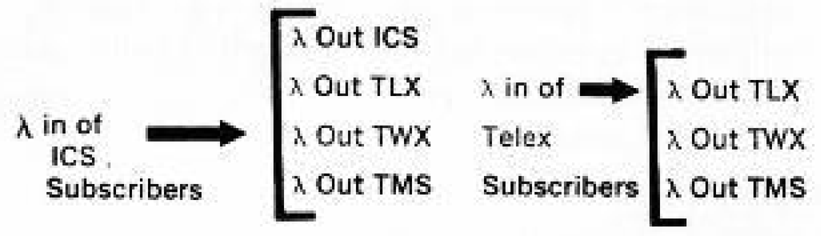
This definition was chosen in order to satisfy the peak hour traffic requirements of the ISCS system. In a system as complicated as ISCS, thruput margin is a function of many influencing factors including traffic characteristics and the time history of traffic characteristics. The three main limitations to system thruput include the maximum possible Central Processing Unit (CPU) utilization, limits of in-process and in-transit storage area, and the input/output (I/O) channel utilizations. All of these are effected differently by the different classes of service incident upon this system and by the traffic characteristics of these services. In this design of demonstration tests it was adequate to linearize the system's complexity, since for the accuracies desired a linear approximation was sufficient when considered locally; and any further refinements because of the complexity of the system would be uneconomical. It is therefore necessary to be quite careful to apply these linear approximations only over dynamic ranges where they can be considered to hold.

ISCS Functional Overview

In the design of a good thruput test, the elements which would restrict thruput first have to be identified. For this reason, a test model was developed which separated these system sensitivities into nearly orthogonal components and then the tests designed to measure these sensitivities. This in effect created a surface of thruput margin of many dimensions as a function of the many components of the environment and system limitations.

Test model development requires an understanding of the ISCS system functions and their implementations. The ISCS system is a message switch—concerned with accepting and delivering messages. It consists of two functional and physical entities; a Communications Center (CC) and a Processing Center (PC). As the name implies, the CC acts as a line processor and concentrating element where data entering on low speed lines, character by character, is analyzed for format, blocked, and shipped to the PC via an intercomputer coupler for processing. A CC element in a network also does primitive routing of the message in order to present it to the proper PC. The PC functionally consists of queue facilities to process the messages. It interrogates the various aspects of the message address for errors or destination, performs privacy and validation checks, routes and queues for output, and outputs to the destination CC once a line to the address has been seized.

ISCS currently services, on input, two types of customers; Telex subscribers, who may dial up the CC, and directly-connected INFO-COM subscribers. An INFOCOM subscriber has a set of terminals with coded answerback drums called an INFO-COM network. Thus, for an INFO-COM subscriber, ISCS is a private message switch utilizing a shared store-and-forward system. In addition, both INFO-COM subscribers and Telex subscribers can send messages via ISCS to Telex, TWX, and TMS (Telegram Message Service). Here, is the first part of definitizing the traffic environment.



Where ICS = INFOCOM Service
TLX = Telex Service
TWX = TWX Service
TMS = Telegram Service
→ = allowable delivery path.
λ in = input message rate
λ out = output message rate

Another aspect of the traffic environment is the multiple address (MA) input message (many addresses for one text message) and for INFOCOM the Group Code message (one input message which keys to many prestored addresses). These two services result mainly in generating more deliveries than input messages; and in addition MA messages have a tendency to increase the mean message length on input, because of the additional addresses.

Another message environment parameter which will effect thruput is the message length. Both CC and PC require the moving of data either within or through core. This will require both an in-process storage level and code to be executed for this function and on output, queues need to be scanned to find a deliverable message.

Hardware for the system consists of the terminals and lines, two UNIVAC 418 computers, one containing the CC functions, the other containing the PC functions, an Intercomputer Coupler (ICC) connecting the PC and the CC and peripheral equipment on the PC I/O channels for maintaining routing files and for storage. The peripheral set consists of a Fastrand Drum, 330 Drum and Tape units.

The Fastrand is a mass storage drum which contains primarily the message data while waiting for delivery. It also contains the necessary journals, some recovery information, overflows of 330 storage areas and routing tables. The 330 drum is much less massive but faster and is used primarily for storage of data which is more frequently used in the processing of a message such as message control packet and message queues. The tapes are used for permanent records and contain all system journals.

Thus, we find that the PC behavior is governed by the input and output traffic rate, the type of service, the message length and the queue statistics; while the CC in effect is dependent on class of service and character rate.

Armed with this understanding of the functions of both the PC and the CC and investigating the implementation of these functions, we can proceed to the development of the test models.

Test Model

The CC functions, as implemented, basically consist of low speed line servicing, high-speed line processing, and task queue processing. Low-speed line servicing is initiated periodically, roughly a little faster than the input or output character rate. As the characters are received, they are blocked and placed on the task queue for processing; on output the blocks are emptied one character at a time. When the low-speed processor is allowed to run it removes an item off the task queue, processes it according to its contents and if it is an input block, schedules it for high speed out to the PC; if it is an output block, it is placed on an output queue for low speed line processing. The CC is mainly character rate dependent and its capacity measured by the contents of the task queue or the wait time statistics of the task queue elements. Thus, a characterization of the CC as to its thruput would be the relationship between task queue wait time and character rate. The character rate taken within the expected environment will then define the line handling and terminating capability of a CC.

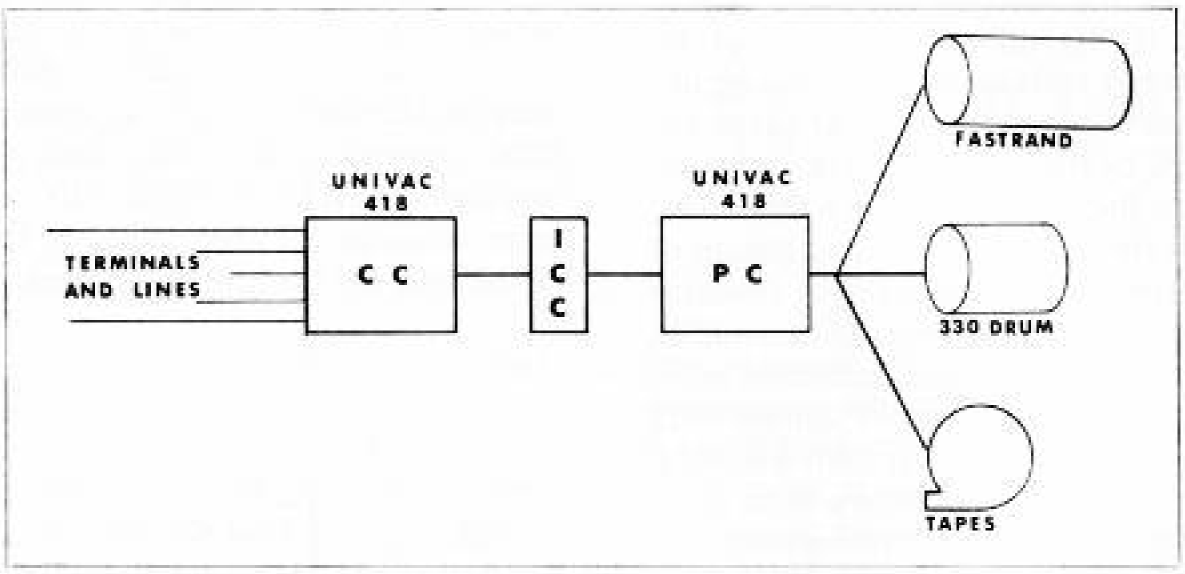


Figure 1—A Typical ISCS Site

As a result of the character sequence detecting for TCCS message format and the implementation of this as opposed to single character detection for formats of INFOCOM messages, the task queue processing for these two services is significantly different. INFOCOM processing time per message in the CC is considerably less than that for TCCS and thus in considering CC performance under volume averages, task queue wait times need to be observed under each type traffic separately and then at various traffic mixes. The functional relationship between the task queue wait time and traffic can then be determined and used to predict performance of any specified traffic and mix. An upper limit to CC thruput is defined by the CC structure which is such that, on input, characters transferred to a circular buffer have to be emptied faster than it is filled. These buffers are segmented into five processing blocks. When one processing block is filled it is scheduled for processing via the task queue where it waits for service. In the mean time the other segments of the buffer are being filled at line speed.

The buffer is entered via a Low Speed Line Service (LSLS) pointer for insertion of characters. When this pointer encounters the last word of this buffer, it initializes to the top of the buffer. The characters are analyzed by low speed processing (LSP), keying into the buffer by the LSP pointer. Overlap is observed when the LSLS pointer catches the LSP pointer. The upper limit thruput margin of the CC is therefore determined by observing, under a given environment (message length and traffic by class of service), the traffic when the task queue wait time consistently indicates overlap in the circular buffer. This overlap has the effect of potentially dropping characters. When the overlap is detected by the CC, the connection on which it occurs is dropped, thus degrading service to the customer.

The PC test model is more application oriented; by this it is meant that as different aspects of the messages are analyzed and as new functions are required, they are scheduled to be performed. These scheduled functions are then performed on a particular message within a PC priority structure established by the Executive system. When the PC has no work to do, it sits in an idle loop which will accumulate idle time.

Functionally we represent the PC CPU utilization (ρ) test model as;

$$\rho = f(\lambda, \psi, Q) \qquad (2)$$

That is, we establish that ρ is dependent in some way on three parameters: the work required for input and output processing of messages (λ), on the moving of data characters for input, output, and journalling functions (ψ), and on the need to scan and find available queue entries to initiate output (Q). From an understanding of the PC functions previously described and their implementation, the functional expression takes the following linear form:

$$\begin{aligned}\rho = {} & A_0 + A_1 \cdot \lambda_{\text{TLX in}} + A_2 \cdot \lambda_{\text{ICS in}} \\ & + A_3 \cdot \lambda_{\text{total in to TMS}} + A_4 \cdot \lambda_{\text{in}} \cdot \psi_{\text{in}} \\ & + B_1 Q_{\text{TLX}} + B_2 Q_{\text{ICS}} + B_3 Q_{\text{TMS}} \\ & + C_1 \cdot \lambda_{\text{TLX out}} + C_2 \cdot \lambda_{\text{ICS out}} + C_3 \cdot \lambda_{\text{TMS out}} \\ & + C_4 \cdot \lambda_{\text{out}} \cdot \psi_{\text{out}} \\ & + D \cdot \lambda_{\text{in errors}} + E \cdot \lambda_{\text{out errors}} + \cdots \end{aligned} \qquad (3)$$

The λs, Qs are respectively the traffic and message queues established by the environment and indexed by the class of service. ψ is the mean message length and is required to obtain character sensitive performance. Here TLX and TWX processing is so similar that we equate TLX to TLX plus TWX. These parameters act as dependent variables in PC utilization models. The As, Bs, Cs, etc., are the various sensitivity coefficients identified in the model whose recovery is required for the test objective. A_0 is a bias coefficient and is obtained under conditions where traffic and queues are zero. It is the overhead work accomplished when there is no message processing to do. A_1 and A_2 are the sensitivities to the different types of input; TLX and INFO-COM. Since TMS destined message routing requires state and city searching in addition to privacy checks and validation implied in A_1 or A_2, these messages have additional processing on the input side. This sensitivity is defined as A_3. A_4 is the character rate sensitivity on the input side. B_1, B_2, and B_3 are the queue size sensitivity coefficient for the TLX, INFO-COM and TMS services respectively; C_1, C_2, and C_3 are the output work for the same services. C_4 is another character sensitivity but on the output side. D, E, etc., are undefined coefficients related to types of input and output error conditions as well as infrequently used services available such as alternate routing terminals and group codes in INFO-COM and amended header processing at an intercept position for TLX.

Test Objectives and Results

These test models make it possible to translate economic objectives of revenue producing traffic into performance test objectives. These test objectives and the results of testing fall into four main categories: Thruput Margin, In-Process Storage Limit, Recovery of PC Sensitivity Coefficients and the CC Task Queue Wait-Time Function.

Thruput Margin

The first and probably the most significant, from a thruput margin sense, is the measuring of the upper-most bound of PC CPU utilization; i.e., the maximum depth of penetration into the idle time at which the system remains operationally stable. In actuality, since there is always some amount of I/O channel functions that have to be serial with CPU there will always be an upper bound of CPU utilization that is less than 100 percent. This objective will, of course, be attainable only if all other system allotmental limits are avoided. This structural limit will be established if the system runs out of channel time or CPU time. The maximum CPU utilization so far attainable for any sustained period of time is approximately 70 percent. Figure 2 is a parametric plot from data obtained during testing of CPU utilization as a function of thruput for the test environment of 100 percent INFOCOM, an exponentially distributed message length variate at various mean message lengths, and the mean output rate within 10 percent of the mean input rate. The gathering of data samples for establishing this relationship consisted of running tests at various levels of thruput under the standard test environment and taking long term averages. The time histories of traffic and CPU utilizations is the first level of reduction. From these graphs, sample points are determined by segmenting the curve into intervals of at least 5 minutes long and usually not longer than 20 minutes at times where transient conditions or changes in traffic level were not observed. The CPU utilization over this interval is then plotted against the thruput for the same interval to obtain the parametric relationship. Once sufficient sample points are obtained which span the range of thruput they are fitted to a linear curve in the sense least squares (i.e., the sum of the squares of the sample points from some curve is minimized by adjusting the curve).

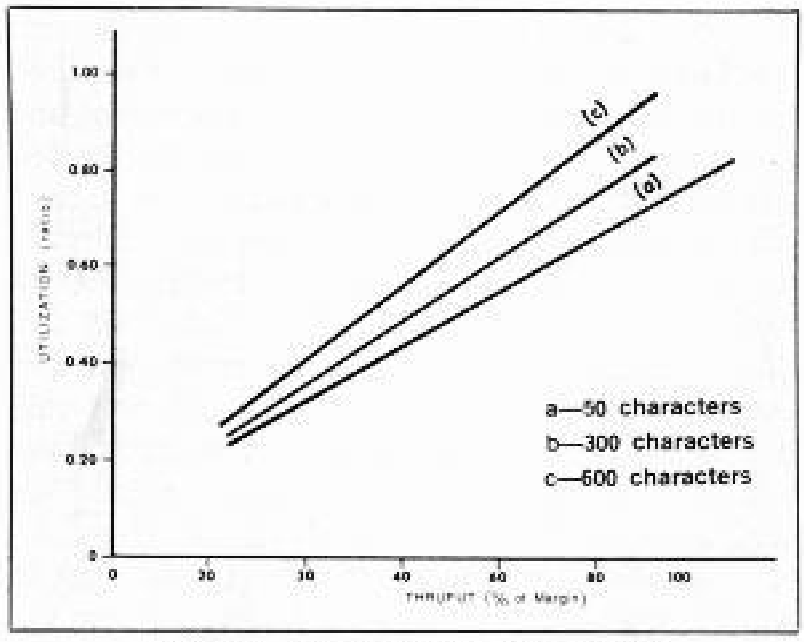


Figure 2—PC CPU Utilization vs Traffic at Various Message Lengths

Figure 3 depicts the input and output traffic rates and CPU utilization time histories taken over 1 minute samples for both 300 and 600 character messages. During both test runs the long term averages of input and output rates are nearly equal, thus yielding sample points for parametric relationships. Two events are noted on the graph; a) the IOSERR table limit is an example of an allotmental limit which was overcome by expanding this table, whereas b) the slot table limit being instruction sensitive it exhibited a structural limit. The reaction of the system after these limits were encountered is peculiar to the individual limit and not necessarily to the type of limit.

At the maximum CPU utilization (the thruput margin point) any attempt to force more traffic through the system will cause a departure of the output traffic from the input traffic. Generally, input has priority and as long as no limits are invoked, the system will accept input even when it does not have any processing time left to handle output. Thus, as the input increases above saturation, the output will decrease. This will have the effect of building queues at a rapid rate and as such remove even more processing time from output. If we were to project this effect and allow the definition of thruput to become nonlinearized, the curve, of Figure 2 would tend to turn around on itself (in effect decreasing thruput).

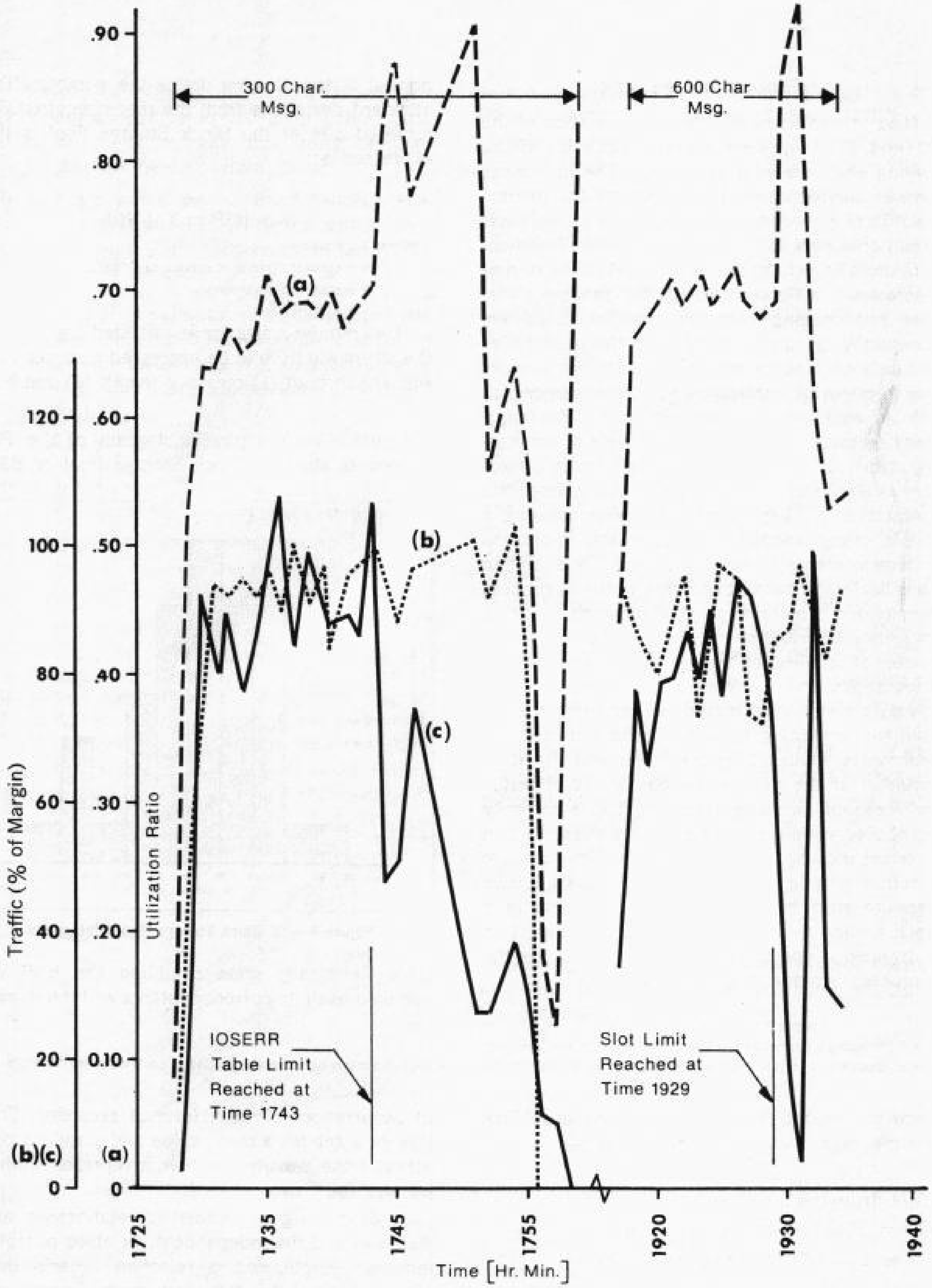


Figure 3—Time Histories of PC CPU Utilization (a), Input Rates (b) and Output Rates (c)

In-Process Storage Limit

Next in significance is the determination of the CC and PC in-process storage statistics under varying environmental conditions. The in-process storage current contents will exhibit the characteristics of a random variable because of two basic conditions. First is the fact that the environment is random in nature; i.e., the arrival times of new messages is a Poisson distributed random variable and message lengths are distributed exponentially. Secondly, because of the use of files and data on random access mass storage devices and because of contention for these devices as well as contention of programs for CPU (since many messages may be in some state of processing at the same time), the service time of a message is also random with a not so easily described characteristic. These two facts (when viewing a simple queue model) will cause the in-process storage contents to be distributed. By sampling periodically the current contents of the in-process storage and maintaining constant mean traffic, a frequency distribution of the number of occurrences of a level of usage of the storage will characterize this performance parameter. From this data a statistical model evolves which will be used for predicting behavior under untested environments. Calculating the mean and standard deviation of the data at various levels of traffic and message size and assuming it is a normally distributed variate we can size this storage given a certain message rejection criteria. The message rejection criteria can be translated (with known message size) into a probability statement of a block finding the storage full since a connection is dropped if one or more of its blocks finds the in-process storage full.

Letting P ( ) — probability of ( )

P (dropping a connection — 1 — P not dropping a connection)

P (not dropping a connection) — P (not dropping the 1st blk P (not dropping the 2nd blk . . . . P (not dropping the nth blk)

Since the probability of dropping any one block of a message is assumed equally likely—

$$\begin{aligned} &P(\text{not dropping a connection}) \\ &\quad = P(\text{not dropping one block})^n \\ &\quad \cong 1 - nP(\text{dropping one block}) \qquad (4) \end{aligned}$$

Where n = average number blocks in a message

With this, we now can specify in a statistical sense a P (dropping a block) and by applying this to a normal distribution we define the number (N) of standard deviations from the mean required. The required size of the Block Storage Pool is then expressed as:

$$\text{Size BSP} = X + N\,S$$

Where X = mean

N = constant (grade of service variable)

S = standard deviation

If we require a size for an untested environment the sizing will have to be expressed parametrically with the thruput (λ) message length (ψ) and N.

Figure 4 is a typical histogram of the PC's in-process storage (Block Storage Pool or BSP).

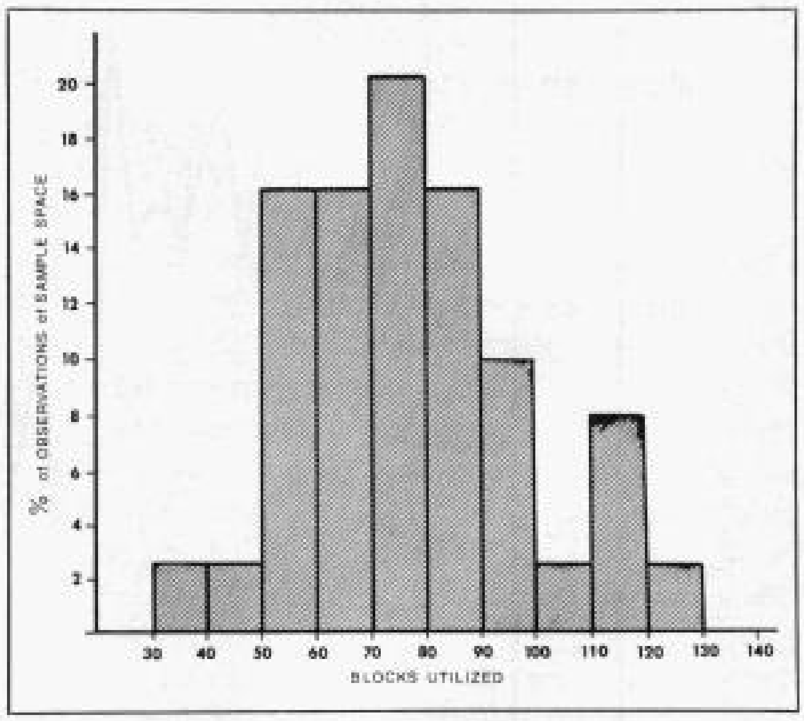


Figure 4—FC Block Storage Pool Histogram

Under a steady state condition the BSP was sampled as to its current contents and the number of occurrences of each interval recorded. From this we establish a mean value and standard deviation of the density function it represents which we use for sizing.

In developing a parametric relationship of a BSP size and the independent variables of traffic, message length and a rejection criteria three observations of the data were made. These were:

1) The mean BSP value was linear with respect to variation in message length while traffic

was held constant

2) The mean BSP value was linear with respect to variations in traffic rate while message length was held constant

3) And finally that the standard deviation was consistently close to twice the square root of the mean BSP independent of the traffic or message length.

With these observations and by taking partials of the mean BSP function first with respect to traffic and second with respect to message length, the following BSP sizing model was developed:

$$Y(\lambda, \psi, N) = k\psi\lambda + 2N\sqrt{k \cdot \psi \cdot \lambda} \qquad (5)$$

where $Y(\lambda, \psi, N)$	=	BSP Size
k	=	1/3, a constant of proportionality
ψ	=	mean message length
λ	=	mean thruput
N	=	number of standard deviations required to satisfy the message rejection criteria

Applying these sizing equations with N equal to 3 and 4, the curves of Figure 5 are generated. Fig. 5 illustrates the size required for the BSP, as a function of traffic. At $N = 3$, we would expect to drop 3 messages out of every 100; where at

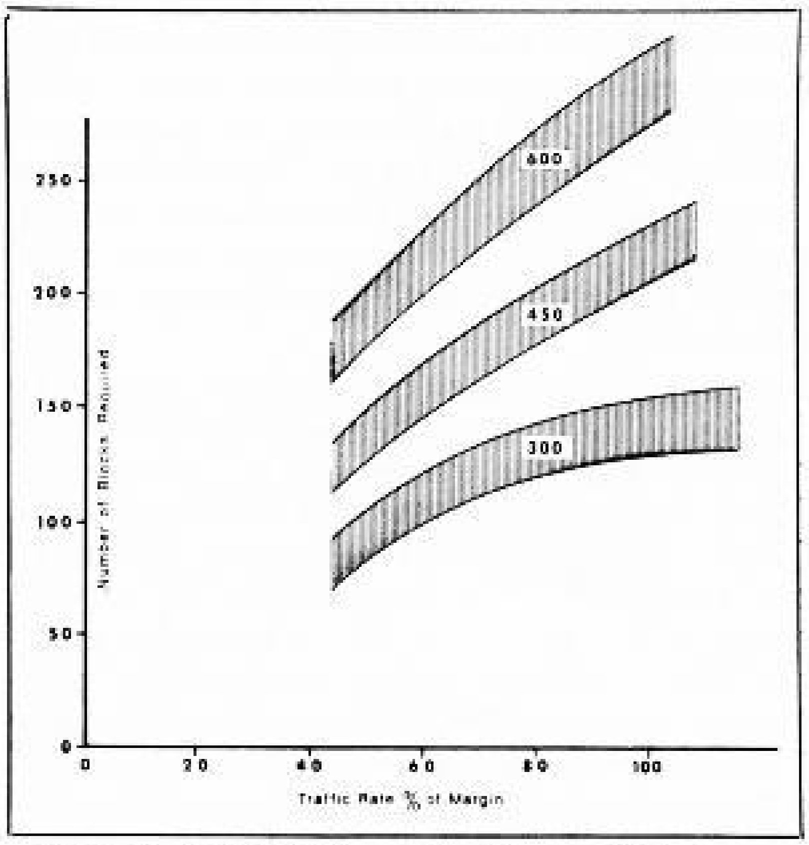


Figure 5—PC Block Storage Pool Average Contents vs Mean Message Length at Given Traffic Levels

$N = 4$ we would expect to drop less than one 300 character message out of every 1000.

This analysis concentrated on the PC BSP, since this was one of the first allotmental limits reached. Although the CC in-process storage statistics indicate a usage greater then the PC BSP there are many more blocks available in the CC. The same analysis of the CC in-process storage would apply except that observations and eventual assumptions made for the PC BSP may take on different forms.

RECOVERY OF SENSITIVITY COEFFICIENTS

The third objective is the recovery of the coefficients of the PC test model. This is accomplished by establishing a baseline level of traffic near the thruput margin and varying one dependent parameter at a time. Since these parameters are essentially orthogonal the variation of one should not significantly effect any other. The relationships are not always linear over the system's total operating range. However, they do exhibit a local linearity and because we are interested in system capacity, these sensitivities coefficients are measured near the thruput margin.

Since the system only requires that output traffic follow closely to input traffic, which also will result in low queues, the PC CPU model was reduced by combining input and output traffic of like service. Thus—

$$\rho = A_0 + A_1\lambda_{\text{TLX thru}} + A_2\lambda_{\text{ICS thru}} + A_3\lambda_{\text{thru to TMS}} + A_4(\lambda_{\text{TLX thru}} + \lambda_{\text{ICS thru}})\psi \qquad (6)$$

Solving for the coefficients we identify partials and then take the related differences to obtain the coefficients.

$$\frac{\delta\rho}{\delta\lambda_{\text{TLX thru}}} = A_1 + A_4\psi \qquad (7)$$

$$\frac{\delta\rho}{\delta\lambda_{\text{ICS thru}}} = A_2 + A_4\psi \qquad (8)$$

$$\frac{\delta\rho}{\delta\lambda_{\text{thru to TMS}}} = A_3 \qquad (9)$$

$$\frac{\delta}{\delta\psi}\left(\frac{\delta\rho}{\delta\lambda_{\text{TLX thru}}}\right) = \frac{\delta}{\delta\psi}\left(\frac{\delta\rho}{\delta\lambda_{\text{ICS thru}}}\right) = A_4 \qquad (10)$$

A_0 is obtained by noting the value of at the intersection of the curve of Figure 2 with the ordinate at zero traffic. As observed A_0 equals .085.

In order to obtain A_1 and A_2 it is easier first of all to obtain A_4, the message length sensitivity. The uncovering of A_4 is done by averaging the series of partials obtained at each traffic level investigated. Thus, by observing the average normalized slope of the curves of Figure 6, we obtain A_4 in terms of utilization in percent per (msg/sec X char/msg).

Therefore,

$$A_1 = \frac{\delta\rho}{\delta\lambda_{\text{TLX thru}}} - A_4\psi \tag{11}$$

and

$$A_2 = \frac{\delta\rho}{\delta\lambda_{\text{ICS thru}}} - A_4\psi \tag{12}$$

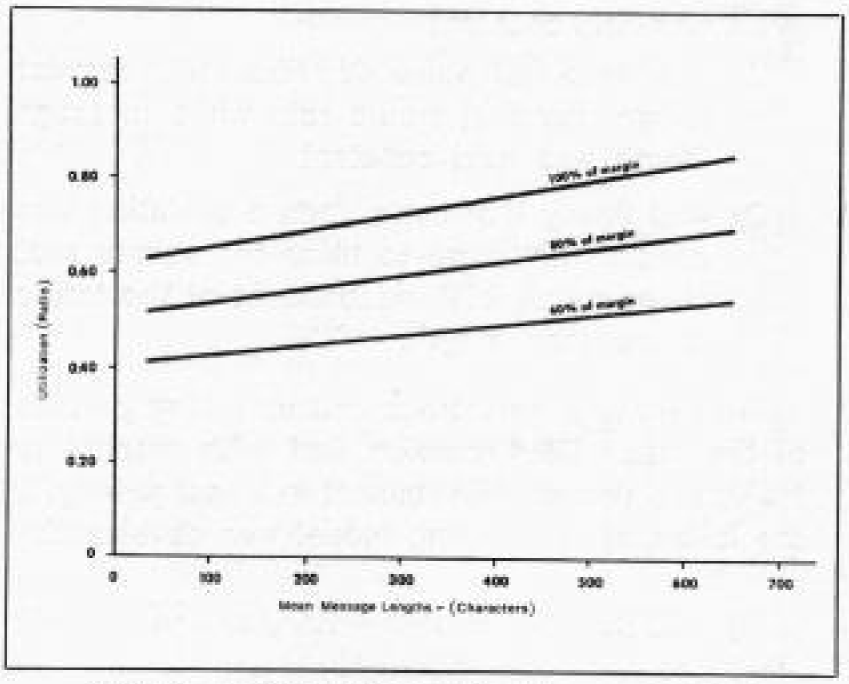


Figure 6—PC Block Storage Pool Average Contents vs Traffic at Given Mean Message Sizes

A_3 is solved for in the same manner as A_4.

For a finer analysis of PC performance more tests can be run where the other identified sensitivities such as those related to queue size, error traffic, and class of service can be separated and the model expanded to incorporate these sensitivities.

CC Task Queue Wait-Time Function

The last objective is the establishing of the task queue wait time function. This in effect would establish the CC thruput margin related to CPU work load as opposed to in-process storage limits. We obtain this function by monitoring the current number of active connections both input and output under a specified environment or a particular character rate and observing the task queue waiting times. By obtaining many such samples at a fixed environment we plot the averages against the thruput rate of a particular message size. The CC thruput, in this sense, is that thruput rate where we just exceed the maximum allowable task queue wait time, which is about 5 seconds. This function, though seemingly straight forward, is complicated by the fact that a collocated CC, which is terminating high speed lines from remote CCs in a network, places these blocks on its task queue for processing before shipping it on to the PC. Figures 7 and 8 on pgs. 174 and 175 are time histories of task queue and connection data for a remote and collocated CC in a network respectively. Both CCs are processing low speed traffic that is equivalent. The collocated CC is also handling the remote CC's traffic for shipment to the PC. We note that the averages differ slightly, but the variations and the means of the maximum contents is many times greater in the collocated CC as opposed to the remote CC.

The task queue wait time is a highly non-linear function of traffic which follows the characteristic queueing/utilization shape. This is observed in Figure 9 which defines the task queue wait time function for both classes of service; INFO-COM and Telex. The critical thruput rate for TCCS is shown in the data for a 300 character message and was easily obtainable, but PC limits were invoked in the all INFO-COM environment before CC task queue wait time reached criticality. This difference is due to the different processing approach taken in TCCS from INFO-COM as mentioned earlier.

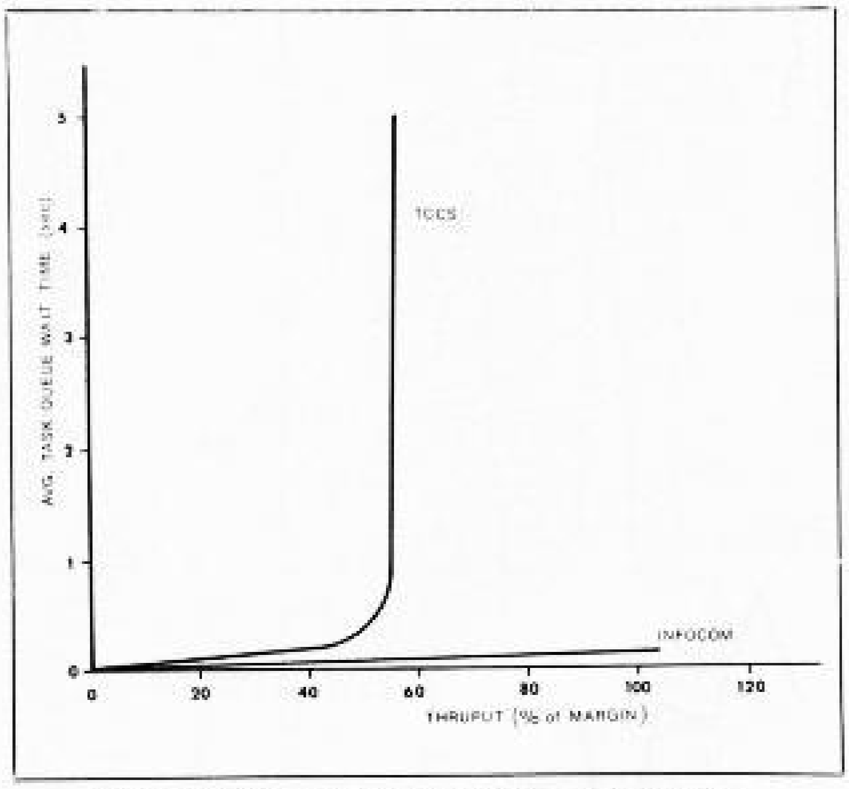


Figure 9—CC Task Queue Wait Time vs Thruput

Test Support

Throughout this article the terms "maintaining the environment", "fixed traffic level", and "long term averages in steady state conditions" have been mentioned. Indeed, in order to establish some mathematical relationship between two parameters, environments have to be maintained, so that the statistical parameters may be time invariant. This controlled environment for performance testing is quite necessary. If a test to be performed requires a sequence of events, to be acted upon at prescribed times, these events have to be coordinated. If a test is run, for instance, where message length sensitivity is being measured, at some time in the test it would be desirable to either change the message length while maintaining all other stimuli constant or to rerun the previous test with only this change in the environment. A controlled traffic environment is rather difficult to maintain where manned terminals are supplying this environment, especially if the input terminals are physically separate and are geographically located some distance from the computer sites.

To avoid this coordination problem and to gain an easily adjustable controlled environment, a software diagnostic called TPUT, was used for performing tests. TPUT is a CC machine resident routine, which obtains control from the line scanning functions and services the line input and output buffers simulating terminals. It maintains its own tables and as such has as little interference in regular CC operation as conceived possible. It does effect CC running time and consumes buffers, but it maintains timing statistics on itself which can be used to update CC data. The TPUT user has complete control of the environment when he specifies the type of input service, the destination mix, the message length function and traffic by the number of active lines. In these tests, the environment was specified first to match expected environment and then modified to separate the various message sensitivity coefficients and gather statistics on the in-process storage limits.

TPUT, which requires ultimately one operator to perform a specified volume test, was particularly efficient in testing the performance of ISCS. The efficiency of testing (useful data per unit time of testing) was very high.

Another important facet of test support is the recovery and reduction of system data. In order to know the task queue wait time or the PC utilization it first has to be measured and then presented externally. The ISCS system maintains and gathers its own status and periodically (over minute intervals) snaps the data on a high speed printer. In the CC, the data consists of items such as: the number of current lines active on input and output, minimum and current available in-process storage buffers, and the maximum over the interval and average over the interval of the task queue wait times. In the PC the accumulated number of input and output messages, peripheral accesses, idle time, blocks transmitted to and from the CC, current status of the contents in the systems queues, and in-process storage availability are monitored.

In addition, the PC provides an option of also gathering the data on tape. Thus, for the PC data off-line data reduction routines were designed and implemented to present the system raw data in engineering terms ready for analysis.

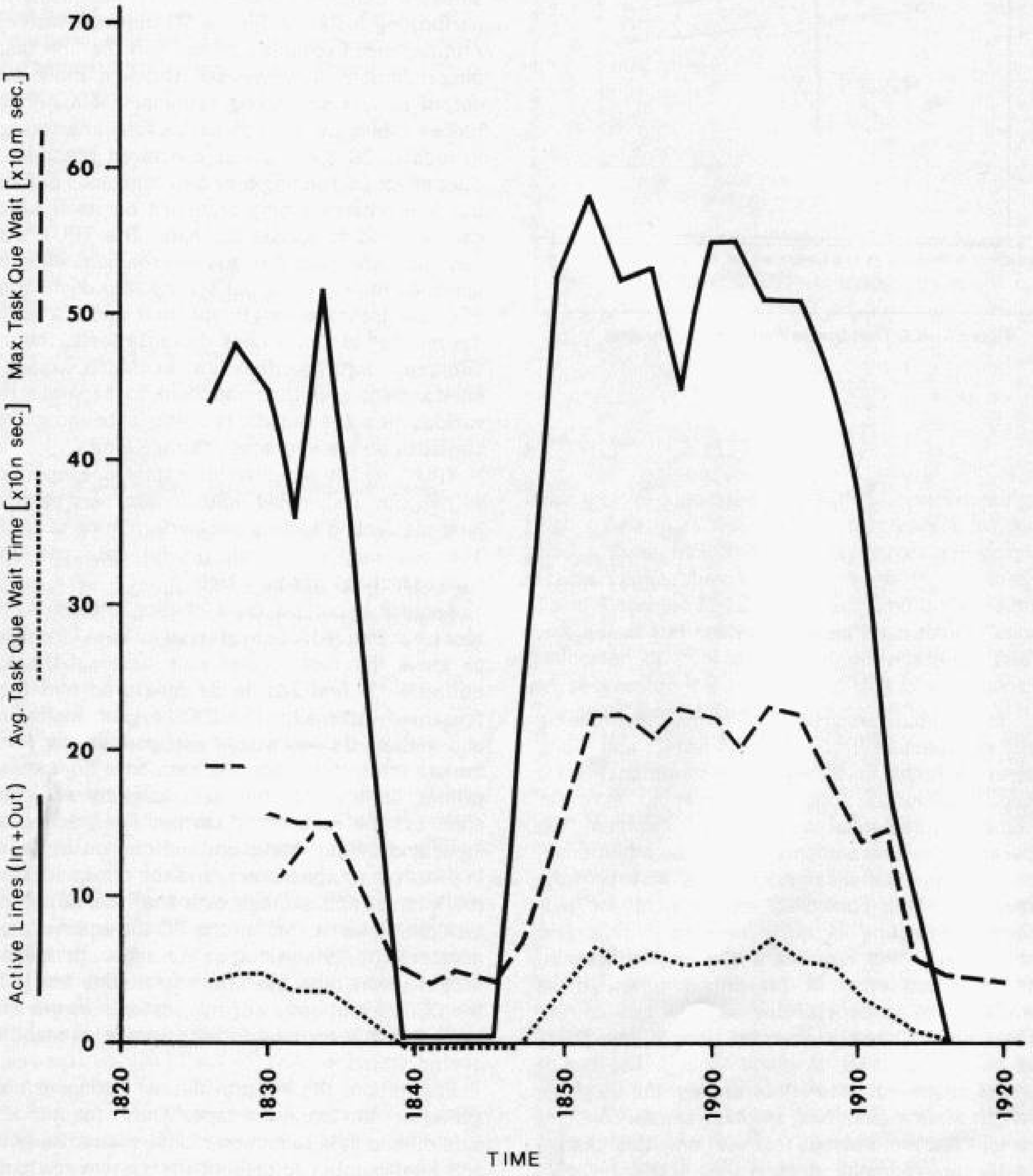


Figure 7—Remote CC Time Histories

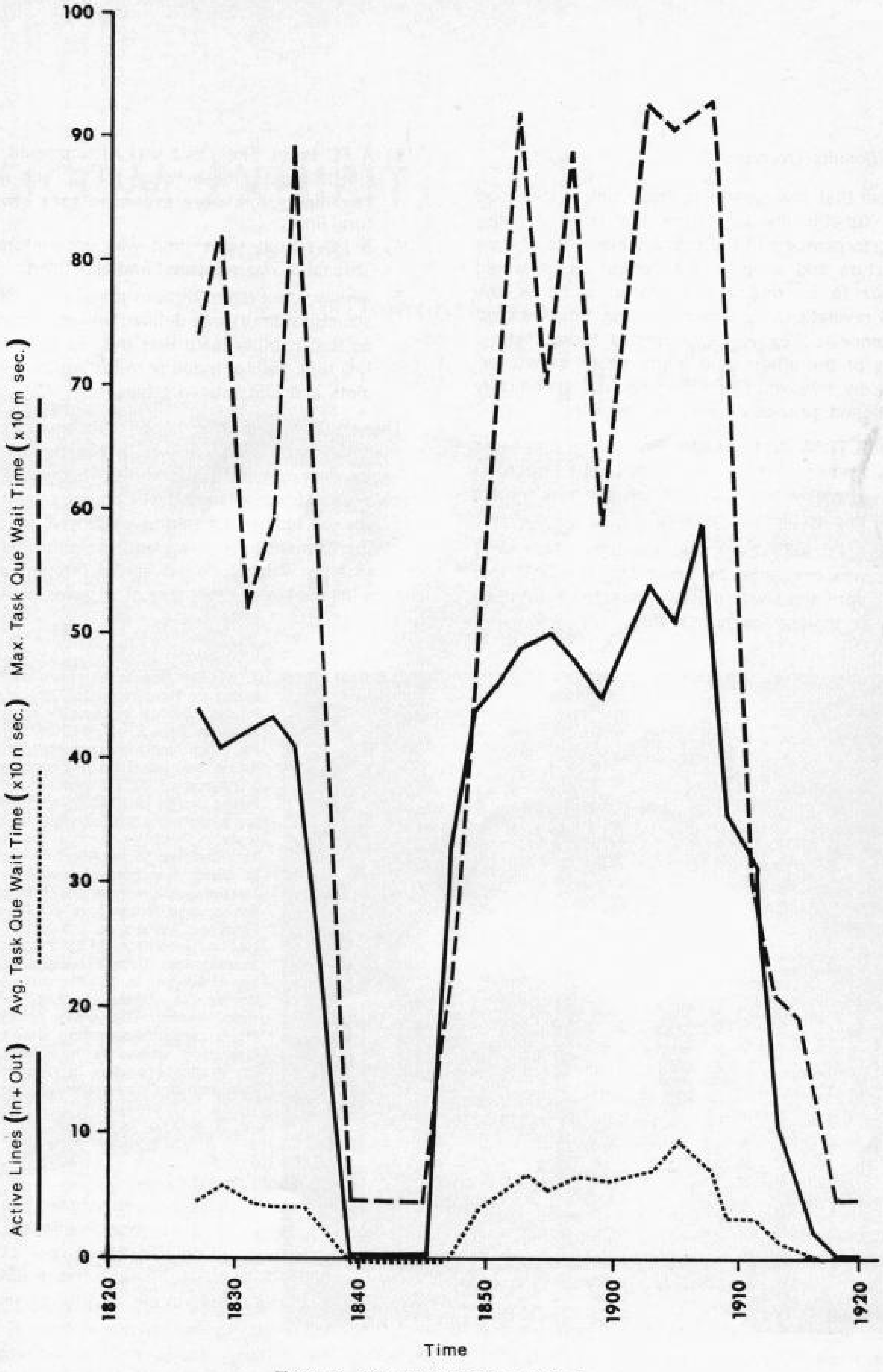


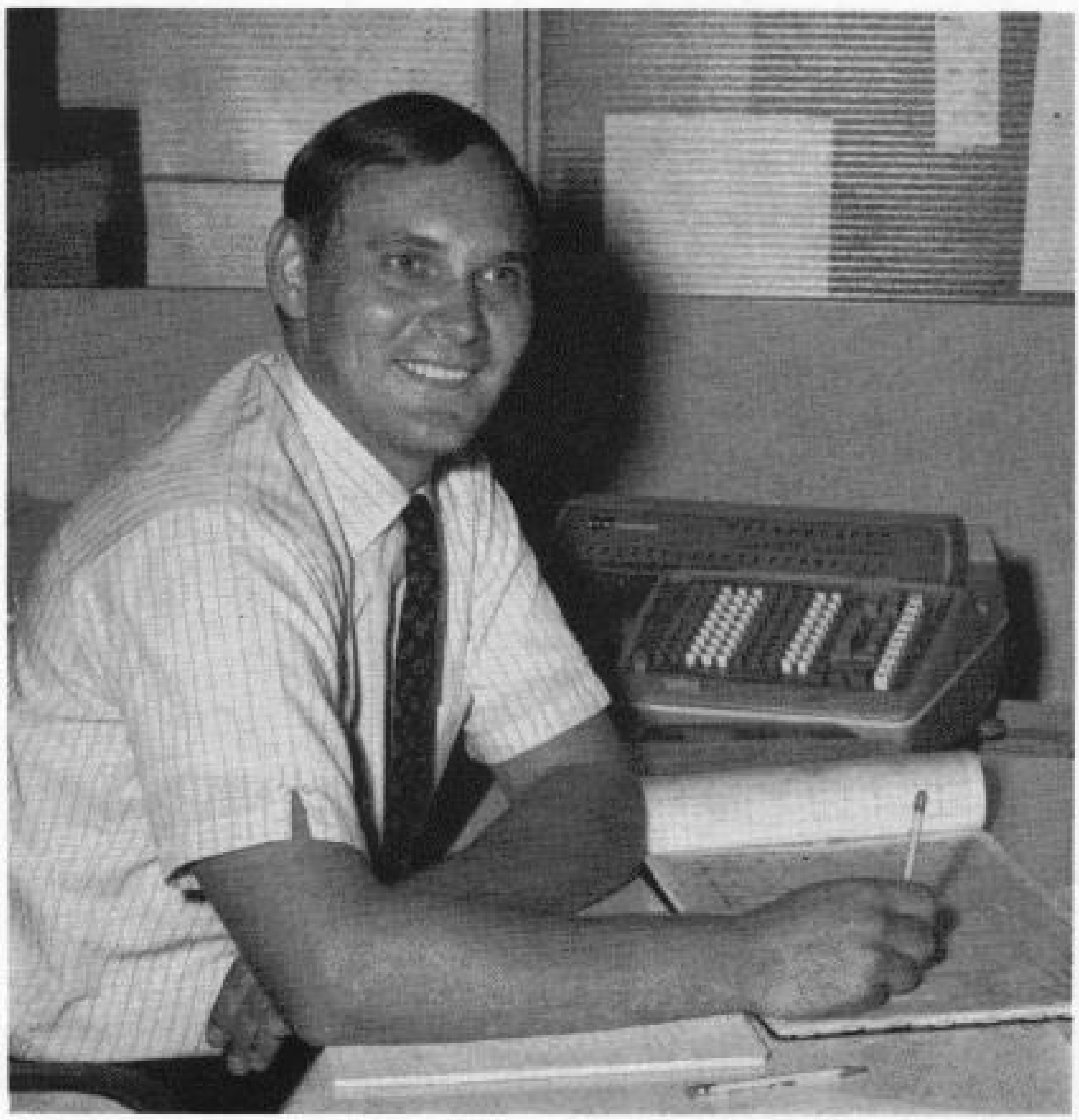
Figure 8—Collocated CC Time Histories

Test Results Overview

Now that the system is fairly well understood both functionally and from the point of view of performance and the primary elements of data reduction and analysis are behind us, it would appear to be desirable to recap some of the ISCS revelations in a more or less time ordered sequence as they were experienced during testing. Some of the allotmental limits which were overcome by relocation or expansion not specifically mentioned previously are the following:

1) A TCCS CC Character Processing Limit was first encountered and found to be structural in nature. A structural redesign would solve this problem if required.
2) A PC BSP Limit was identified. This limit was overcome by expanding the BSP into core areas where service functions not used in testing normally reside.
3) A PC Input Slot Limit was encountered. A slot is a data link between the CC and PC. The input slots were expanded to a structural limit.
4) A PC output table limit was encountered. This table was relocated and expanded.
5) Finally, the system thruput margin (PC CPU structural limit) was defined and recognized as the inability to further overlap CPU and I/O time and/or inability to further increase slots and slot related tables.

These "discoveries" and their subsequent circumvention were of some aid in the design of software enhancements. Since ISCS is typical of a major real-time software development program, this type of testing technique employed to upgrade performance efficiency and the subsequent data analysis which followed are of interest and can be invaluable to designers of systems similar to ISCS.



T. M. Derlinga is Manager of Performance Requirements, Design and Analysis in the Systems Design and Analysis Group, of P&EO.
He is currently responsible for defining and monitoring system performance of ISCS I and for aiding in the design of ISCS II by developing system models, simulations and tests.
Mr. Derlinga joined Western Union in 1966. Previously he worked on inertial guidance navigation system design and evaluation at General Precision Aerospace.
He received his M.S. degree in mathematics from Fairleigh Dickinson University in 1967 and his B.S. degree in Electrical Engineering from Villanova University in 1959. He is currently attending New York University where he is continuing his graduate studies in Operations Research and Computer Sciences.

SICOM EXHIBIT has live Demonstration at ASEF SHOW

The Securities Industry Communications system was demonstrated live for over 1500 members of the Association of Stock Exchange Firms, at the New York Hilton, N.Y.C. on September 3 thru September 5, 1969.

The exhibit was designed to show the flow of information through SICOM's three work areas: the stock exchange, the brokerage firm's wire room and the local brokerage office. Viewing the exhibit were senior management partners and operations personnel as well as communications managers who sought further information and literature on the system. Printed brochures and reprints of the article which appeared in the Summer 1969 issue of the TECHNICAL REVIEW, entitled, "Shields & Company—The First Customer on Wall Street to Cut-Over to SICOM" were picked up quickly.

The major application of the simulator has been in the analysis and evaluation of Western Union's SICOM system. It will serve as a valuable tool in the performance analysis of future applications such as Order Matching, which is to be integrated into the SICOM system.

A SIMULATOR FOR EVALUATION OF A SHARED COMMUNICATIONS SYSTEM

by C. FREY

A simulator was designed by Western Union for use with a shared communications system for multiple subscribers. A shared system has an inherent volume testing problem in that traffic buildup is on an incremental basis as new subscribers are added. Consequently the full capacity of the system is not realized for a long period of time. Therefore, unless a very large network is available for volume testing, system performance at full capacity cannot be evaluated until full capacity is actually achieved. To overcome this problem a simulation program, which can provide maximum loading under various test conditions, has been developed.

The basic hardware configuration to which the simulator is applied consists of two Univac 418 II computers connected by an Intercomputer Synchronizer (ICS). One 418 computer, designated as the Front End (FE), acts as an interface between the message switch and the communication lines. The other 418, designated as the Message Processor (MP), acts as the basic message switch—receiving messages from the FE, performing analysis on them, and routing them through the FE to their proper destination. The simulator is part of the FE; it actually has two functions, 1) it completely replaces the Front End programs, and, 2) it is integrated with the Message Processor to perform data capturing functions for volume testing.

Front End Simulator

The FE simulator, located in the FE computer, completely replaces the normal on-line programs. Its basic function is to generate messages at a controlled rate and according to a predetermined Active Test Pattern. These messages are sent across the ICS to the MP, which processes them as it would during normal on-line operation and sends them back to the FE. The FE throws the returned message away and sends an End of Message (EOM) to the MP after a ten second delay.

This ten second wait simulates a good transmission to an ASR set of an eighty (80) character message. This is the average length of message handled during on-line operation.

Message Types

The FE simulator is designed to generate two basic message types. The first is a Formatted Text Message which is very strictly validated for line length and specific field content by the Message Processor. The second type is an Administrative Message which is validated only for proper header format and message length by the Message Processor. Both message types are thirty (30) characters long. Either message type may be routed to single or multiple destinations or addresses.

Test Patterns

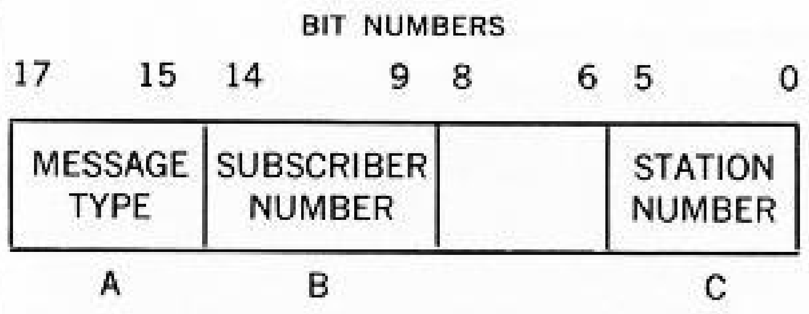
Two types of test patterns have been developed: active and non-active.

a) Active

The messages are generated according to an Active Test Pattern, created from the computer console at the initiation of the simulator run. Once this test pattern is created, it remains constant and is repeated in cycles throughout the duration of the test. The messages are generated at a controlled rate, but this rate may be changed at any time during the test by a console entry. This rate may vary from one message per second to a maximum of fifteen messages per second.

b) Non-Active

The simulator also contains eleven core Non-Active Test Patterns which are inserted at assembly time. These Non-Active Test Patterns are combined by console entries at the beginning of the simulator run to produce the *Active* Test Pattern. These Non-Active Test Patterns contain one word for each message to be generated by the pattern. The words in memory have the format shown in Fig 1.



FIELD A

Indicates any one of the following four message types:
Multi-Address Administrative
Multi-Address Formatted Text
Single-Address Administrative
Single-Address Formatted Text

FIELD B

Contains a Subscriber Number. This field is not specified until the Active Test Pattern is generated.

FIELD C

Contains a Station Number.

Figure 1—World Format for a Non-Active Test Pattern

The following Non-Active Test Patterns exist in the simulator:

P001	0400001	SA Formatted Text	Station 1
	0100002	MA Administrative	Station 2
	0300003	SA Administrative	Station 3
	0200004	MA Formatted Text	Station 4
	0000000	End of Pattern	
P002	0300005	SA Administrative	Station 5
	0300006	SA Administrative	Station 6
	0300007	SA Administrative	Station 7
	0300010	SA Administrative	Station 8
	0000000	End of Pattern	
P003	0400011	SA Formatted Text	Station 9
	0200012	MA Formatted Text	Station 10
	0000000	End of Pattern	

MA—Multiple Address
SA—Single Address

It can be seen from the above illustration that Non-Active Pattern one, P001, has the capacity for producing four messages, as does Pattern two, P002. Pattern three, P003 however, can produce only two messages. It must be remembered that they can produce *no* messages unless they are chosen as part of the Active Test Pattern.

How an Active Test Pattern is Generated

At the start of a simulator run, the Non-Active Test Patterns are combined to form an Active Test Pattern via a series of console entries. This generation takes place in the following 8 steps:

1. Program Request	ENTER SUBSCRIBER
2. User Response	Number from 1-20 indicating which subscriber is to be associated with the following pattern(s), or ALL—if all subscribers. If ALL, program proceeds to step 5.
3. Program Request	NEXT
4. User Response	The next subscriber number to be associated with the following pattern(s), or END if no more subscriber numbers are to be entered. If a number is entered, the program will return to step 3. If END is entered, the program will proceed to step 5.
5. Program Request	ENTER DESTINATIONS
6. User Response	The three digit number of a Non-Active Test Pattern to be associated with the previously entered subscriber numbers. This number must be proceeded by a P. For example: P001.
7. Program Request	NEXT
8. User Response	The next test pattern to be associated with the previously entered subscriber numbers, or END if no more pattern numbers are to be entered. If a pattern number is entered, program will return to step 7. If END is entered, the program will return to step 1 for a new group of subscriber numbers. This process continues until an answer of STP is received to step 1. When STP is entered, the Active Test Pattern is complete.

The process that actually occurs during this operation is that the Non-Active Test Patterns specified are moved to another section of core storage and the subscriber numbers specified are inserted in field B, as shown in Fig. 1.

Example A

As an example of an Active Test Pattern, let us use the previously illustrated Non-Active Test Patterns and associate pattern one, P001, with subscribers 2, 4, and 10, pattern two, P002, with subscribers 5 and 6, and pattern three, P003, with subscriber 11. The console entries are as follows:

```
TYPEOUT:     ENTER SUBSCRIBER
RESPONSE:    2
TYPEOUT:     NEXT
RESPONSE:    4
TYPEOUT:     NEXT
RESPONSE:    10
TYPEOUT:     NEXT
RESPONSE:    END
TYPEOUT:     ENTER DESTINATIONS
RESPONSE:    P001
TYPEOUT:     NEXT
RESPONSE:    END
TYPEOUT:     ENTER SUBSCRIBER
RESPONSE:    5
TYPEOUT:     NEXT
RESPONSE:    6
TYPEOUT:     NEXT
RESPONSE:    END
TYPEOUT:     ENTER DESTINATIONS
RESPONSE:    P002
TYPEOUT:     NEXT
RESPONSE:    END
TYPEOUT:     ENTER SUBSCRIBER
RESPONSE:    11
TYPEOUT:     NEXT
RESPONSE:    END
TYPEOUT:     ENTER DESTINATIONS
RESPONSE:    P003
TYPEOUT:     NEXT
RESPONSE:    END
TYPEOUT:     ENTER SUBSCRIBER
RESPONSE:    STP
```

The Active Test Pattern generated in the memory in Example A, by the communication, would look like the following:

P001	0402001	Subscriber 2
	0102002	
	0302003	
	0202004	
P001	0404001	Subscriber 4
	0104002	
	0304003	
	0204004	
P001	0412001	Subscriber 10
	0112002	
	0312003	
	0212004	
P002	0305005	Subscriber 5
	0305006	
	0305007	
	0305010	
P002	0306005	Subscriber 6
	0306006	
	0306007	
	0306010	
P003	0413011	Subscriber 11
	0213012	
	0000000	End of Pattern

The simulator program would then, beginning at the top of the Active Test Pattern, proceed sequentially down the pattern, generating the messages specified. When reaching the bottom, the scan is begun at the top again. This process is repeated until the end of the test run.

MP Simulator Routine

The function of the Message Processor simulator routine is to record information under simulated load conditions, and use it to evaluate system performance during the test.

As soon as a message enters the Message Processor, it is assigned a slot in a table. It will retain possession of this slot until it has been transmitted through the Front End to an ASR set and an End-of-Message is received from the Front End. When this slot is assigned to the message, the data capturing routines are activated and continue to record until the slot is returned to the available pool. The information gathered is then written on magnetic tape. Thus, the system performance, during the handling of the message selected is recorded. The selection of a message to be traced is completely random. The first message to enter the system activates the data gathering routines. After the table slot corresponding to that message has been released, the next message to enter will reactivate these routines, independent of the number or type of messages entering the system, in the intervening time.

Data Recorded During Trace

The following data is recorded during the trace of a message:

1. Dayclock time of start of trace.
2. Slot number of message which actuated this trace.
3. Accumulated time from the point at which the message being traced entered the Message Processor until it is selected for output.
4. Accumulated time from the point at which the message being traced entered the Message Processor until its slot has been released to the available pool.
5. Idle time during trace. This idle time is defined as being the total time that no user programs are actively executing. They may be in a completely non-active state (not in control and not desiring control) or in a suspended state (e.g. waiting for the completion of an Input/Output operation).
6. Ratio of idle time to total trace time.
7. Input message rate during trace time.
8. Number of message slots given out during the trace.
9. Number of message slots returned to the active pool during trace.
10. Number of message slots in use at start of trace.
11. Number of system generated error messages produced during trace.
12. Various other parameters not meaningful to anyone not intimately familiar with the system design and operation.

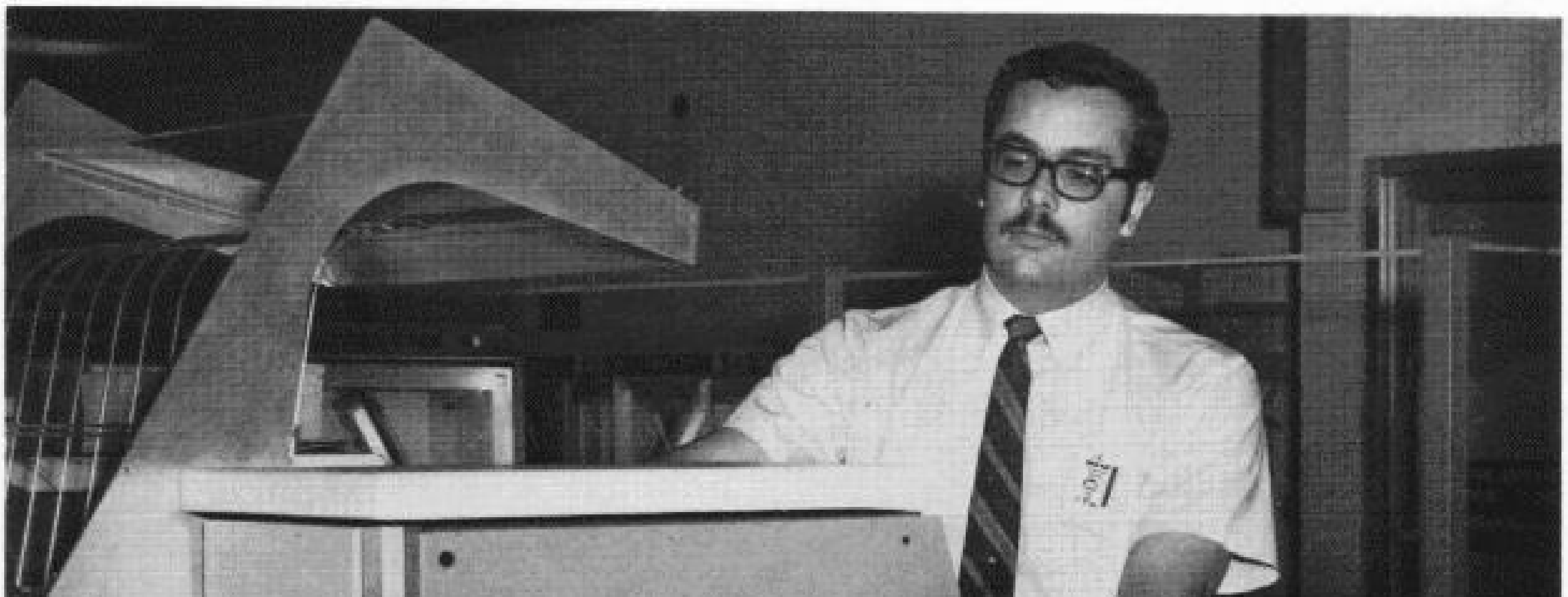
SICOM Applications

This simulator has one significant application. It was particularly useful in evaluating Western Union's SICOM, the results of this application helped gain valuable insight into future system performances.

C. Frey, Jr. is Senior Communications Analyst in P&EO, responsible for the modifications to SICOM.

He joined Western Union in April, 1969, and has been involved in the SICOM simulation. Previously, he was with the Research and Development Department of Univac, concerned with systems programming.

Mr. Frey received his B.S. degree in General Science from Penn State University in 1966.



SYSTEM SIZING and Simulation in the Design of Communication Networks with Memory

by Leonard Stier

Before implementing a communications network, the network designer must resolve a number of basic questions related to system sizing. Given a set of communication system requirements, a variety of factors must be considered and systematic studies must be carried out to determine the number of switching centers, fix their respective locations and design the intra-system trunking network. While a comprehensive description of network modeling and optimization will not be presented in this article, the nature of the problem and applicable approaches will be summarized before proceeding to the main topic of interest—network simulation.

System sizing may be approached as a trade-off between line cost and switching cost. The nature of the trade-off can be seen from the fact that as the number of switching centers increases at a corresponding increase in switching cost, the distance from any customer to the nearest switching center is likely to decrease with a corresponding decrease in line costs. Therefore, access costs together with trunking costs are traded-off against switching costs.

A communication network, N, consisting of a set off n interconnected switching centers is illustrated by the Graph G in Figure 1 having n vertices (nodes), V_1, V_2,---, V_n and m branches (links), b_1, b_2, ---, b_m. Each branch b_l is connected between a pair of distinct vertices v_k and v_j. The vertices and branches in G correspond to the switching centers and trunks in network N. Associated with each branch is a real non-negative number which represents the capacity of a corresponding trunk in N and a direction in which traffic may flow. Such a graph is then called a weighted directed graph.

Due to the similarity of communications problems with transportation problems, some attempts have been made at using the optimization techniques developed in the latter field. However, as shown in Figure 1, formulations for communication networks involve a higher order of complexity since each node does not simply serve as an origin, or a destination for commodities (messages) but is simultaneously an origin, destination or relay

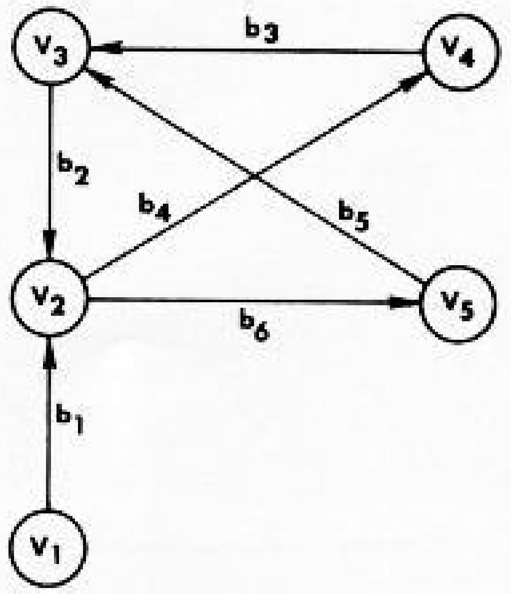


Figure 1—A Typical Communications Network—Graph G

point. Furthermore, even the simpler models must have a multi-commodity formulation corresponding to individual message flows of different priorities between origin-destination pairs. An added complication in the design of store-and-forward systems arises from the presence of memory (storage) at the nodes. Thus, at any given time the flow out of any node may be less, equal or greater than the flow into the node. A powerful model which takes into account both the time dependent nature of the traffic and message priorities has been developed by Western Union[1]. This is a linear programming formulation applicable to both circuit switching and message switching networks. While impractical to apply to large complex systems, questions related to optimal routing, node storage and trunking capacity requirements of smaller systems can be studied. Of greater applicability, however, in system sizing have been iterative procedures programmed to generate hierarchial networks with specified properties representative of those required[2].

Once an outline of the structure of the network has been obtained, the many design decisions subsequently made must be verified. As such, simulation has proven an important tool in providing additional insight into the operation of complex systems.

The General Purpose System Simulator (GPSS) has been applied to study ISCS-I subsystems. As previously reported, the architecture of the ISCS-I message switching element, the Processing Center, has been simulated[3].

In addition, characteristics of a representative part of the ISCS-I trunking network have been studied using an expanded version of GPSS2 running an a Univac 1108 processor.

ISCS-I Network

Structure

ISCS-I is a hierarchial computer network which provides message switching services and interconnection to a set of terminals at various geographical locations, shown in Fig. 2.

The system can be divided into five subsystems:

1. The Terminal Subsystem—made up of user stations.
2. The Access Subsystem—made up of lines and cost-effective equipment—sharing techniques such as concentrators, way-circuits, and multiplexors.
3. The Switching Subsystem—consisting of Univac 418 Processing Centers (PC) and Univac 418 Communication Centers (CC) which provide the following functions: terminal servicing, language and format translation, directory look-up and routing, concentration and billing. A fall-back Univac 418 CPU is present at each site.
4. The Trunking Subsystem—which provides the major arteries of communication transmission.
5. The Network Control Subsystem — which provides system status monitoring and network management, for the ISCS message switching system. The Tech Center, which monitors the network is located in Mahwah, N.J.

Message Flow

Messages from TELEX and INFO-COM terminals access the ISCS-I system via low speed lines at the CCs. Once accepted by the CC, a particular message is routed to a suitably chosen PC (sometimes using an intermediate CC) for processing and billing. Depending on the multiple address factor, one or more messages are subsequently transmitted to their corresponding destination point CCs for delivery to TELEX, TWX, INFO-COM or TMS terminals via low-speed lines, thus completing a transit of the ISCS-I system.

All information transmitted on the high-speed trunks between the CCs (and on the inter-computer synchronizer between CC and PC) is in a block format. These blocks are generated in two ways. First, as part of the process of transmitting segments of messages between a CC and a PC, blocks containing up to 50 text characters are used. Secondly, as part of the control process for inter-CC transmission, various types of blocks are used as part of the block acknowledgement procedure on full-duplex trunks. The latter type of control blocks can add a significant overhead to the limited inter-CC transmission capability, thereby affecting the operation of the system under heavy traffic loads.

The procedures used to control the flow of messages between the ISCS-I sites have been simulated, in order to obtain a measure of the rate at which blocks are transmitted between the sites and to determine the resulting CC and trunk utilization. A description of these control procedures follows.

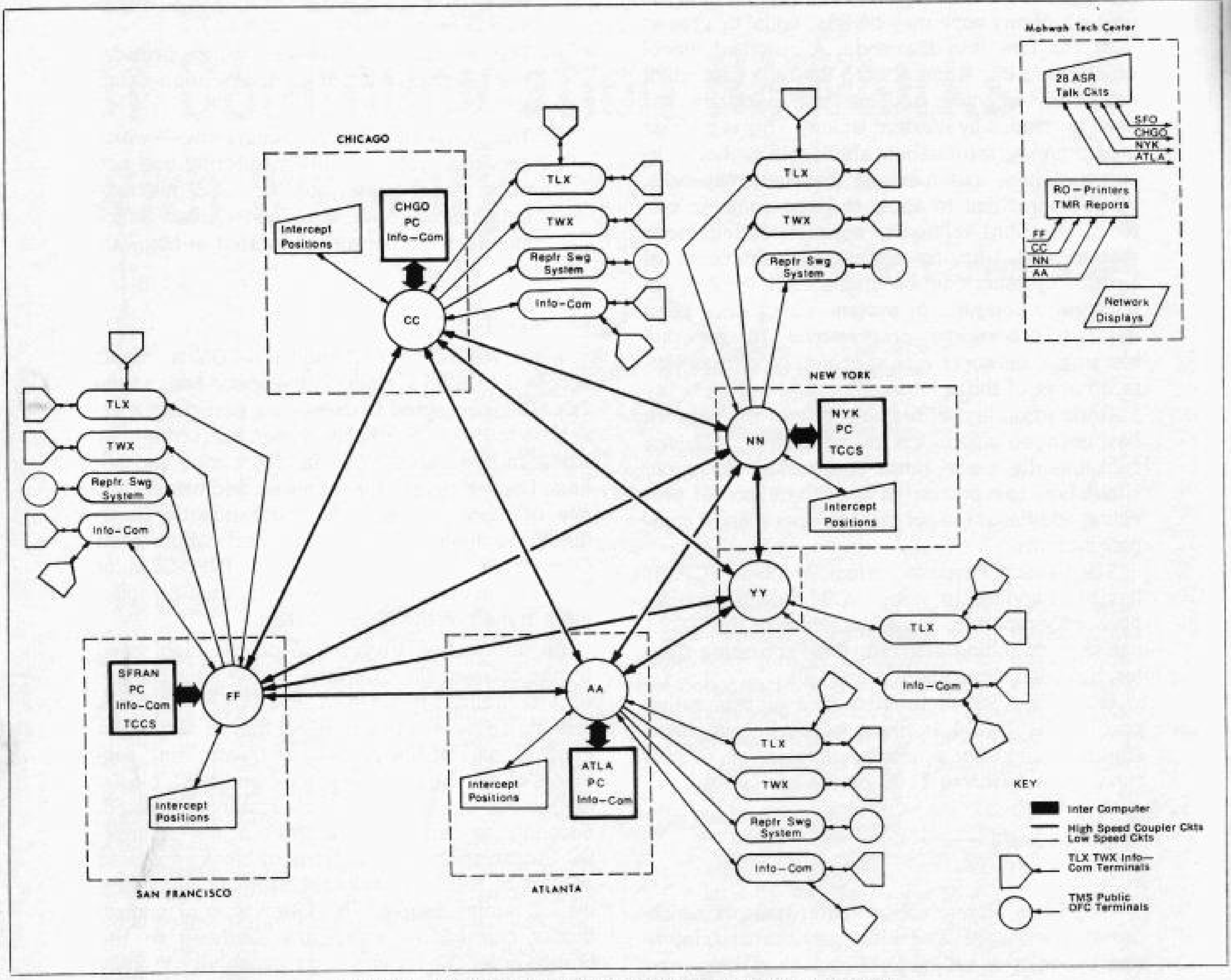


Figure 2—ISCS Phase I TCCS/INFO-COM Network with Interfaces

ISCS-I Message Transmission Control

A simplified account of the procedures used in ISCS-I to control the transmission of a message from its point of entry into the system to its point of delivery is indicated on the flow charts shown in Figs. 3, 4. Figure 3 concerns the Message Input; Figure 4 illustrates the Message Output.

Message Input

A service request generated by a terminal will be recognized at the called CC as an Input Request. After completing an interplay procedure, the CC connects the terminal and forms a BID block. This block consists of 4 SYNC characters, 12 control characters, up to 50 data characters and two characters formed from block parity check bits. The bid block is transmitted to the PC and it contains the terminal answerback for validation. Meanwhile, the CC begins accepting the incoming message from the connected terminal.

Provided the PC signals message acceptance by returning a CONNECT block within 15 seconds, the accummulated portions of the message are subsequently transmitted to the PC in block format.

The end of the transmission is signalled to the PC by means of a special EOT block. Successful receipt (in some cases requiring block retransmissions) of all segments of the transmission will be signalled to the terminal by an acknowledgement message sent from the PC to the terminal via the CC. The terminal is subsequently disconnected by the CC which signals this action to the PC by means of a DISC-ACK block.

As a result of the holding time for any connection, which is of the order of one minute, exceeding the message inter-arrival time, there will be a number of simultaneous connections active at any CC at any point in time.

The control of the transmission of interleaved message blocks on the 2400 baud inter-CC trunks is carried out by means of an Acknowledgement Waiting Table (AWT). Prior to block transmission, the CC enters the Block Sequence Number (BSN) into the AWT together with a notation of the time of transmission. Blocks received subsequently from the other site over the full-duplex trunk are checked for a corresponding Acknowledgement Block Sequence Number (ACK/BSN). The received ACK/BSN will be used to clear the corresponding entry in the AWT. However, should the distant CC fail to return the ACK/BSN this will be revealed

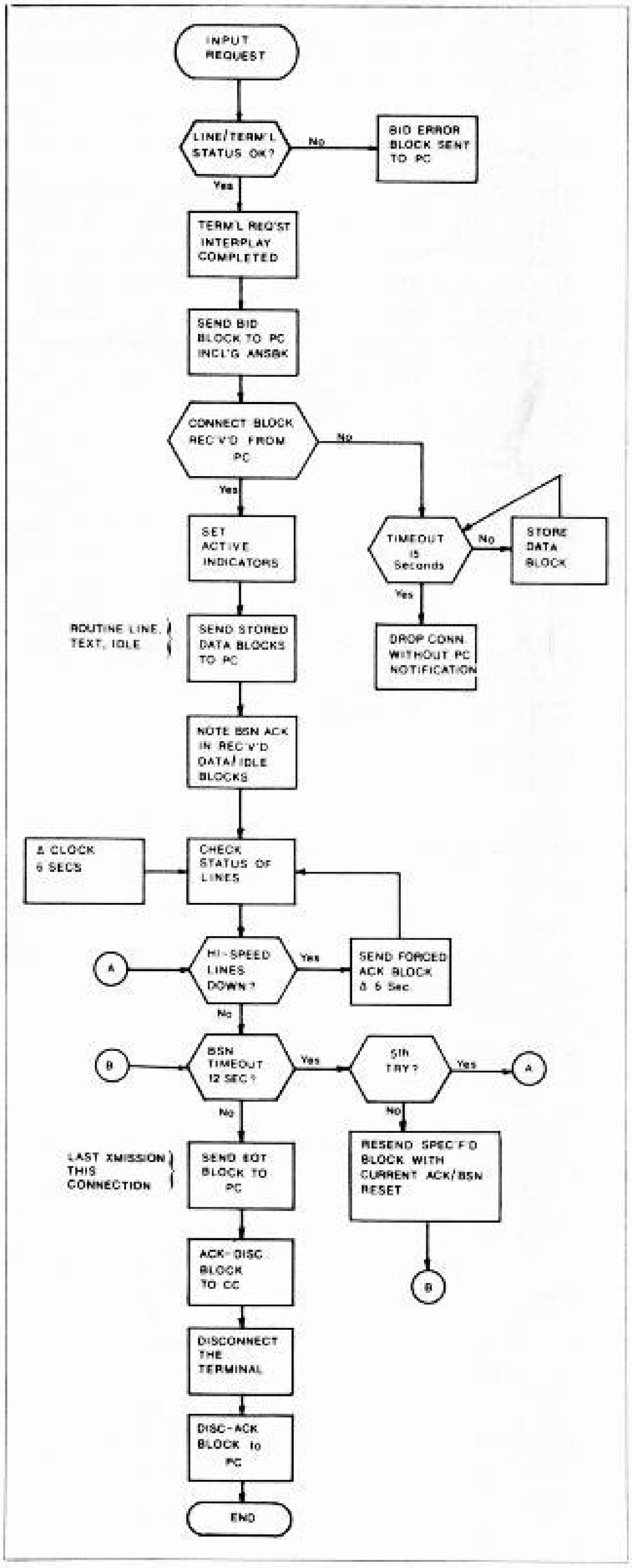

INPUT REQUEST
LINE/TERM'L STATUS OK?
No
BID ERROR BLOCK SENT TO PC
Yes
TERM'L REQ'ST INTERPLAY COMPLETED
SEND BID BLOCK TO PC INCL'G ANSBK
CONNECT BLOCK REC'V'D FROM PC
No
Yes
TIMEOUT 15 Seconds
No
STORE DATA BLOCK
Yes
DROP CONN WITHOUT PC NOTIFICATION
SET ACTIVE INDICATORS
ROUTINE LINE, TEXT, IDLE
SEND STORED DATA BLOCKS TO PC
NOTE BSN ACK IN REC'V'D DATA/IDLE BLOCKS
Δ CLOCK 6 SECS
CHECK STATUS OF LINES
A
HI-SPEED LINES DOWN?
Yes
SEND FORCED ACK BLOCK Δ 6 Sec
No
B
BSN TIMEOUT 12 SEC?
Yes
5th TRY?
Yes
A
No
No
LAST XMISSION THIS CONNECTION
SEND EOT BLOCK TO PC
RESEND SPEC'F'D BLOCK WITH CURRENT ACK/BSN RESET
B
ACK-DISC BLOCK TO CC
DISCONNECT THE TERMINAL
DISC-ACK BLOCK to PC
END

Figure 3—Flow Chart Message Input

by a check of the AWT which is made periodically. The CC will then retransmit all the blocks affected by the above transmission error.

Message Output

The PC generates an output request, as shown in Fig. 4, by transmitting a BID block to a CC. In order for answerback validation to be performed at the CC, this block contains the answerback of the terminal to which the mesage is to be delivered. Should the CC be able to complete the connection, a CONNECT block is returned to the PC at the completion of the interplay with the terminal. Should a network or terminal busy condition be encountered, the call will eventually be repeated.

The message is sent to the CC in block size segments at a rate commensurate with the low speed line rate of transmission. At the end of the transmission and prior to disconnecting the terminal, the CC will again trigger the terminal answerback. Successful validation and termination of the connection is signalled by means of a DISC-ACK block sent to the PC.

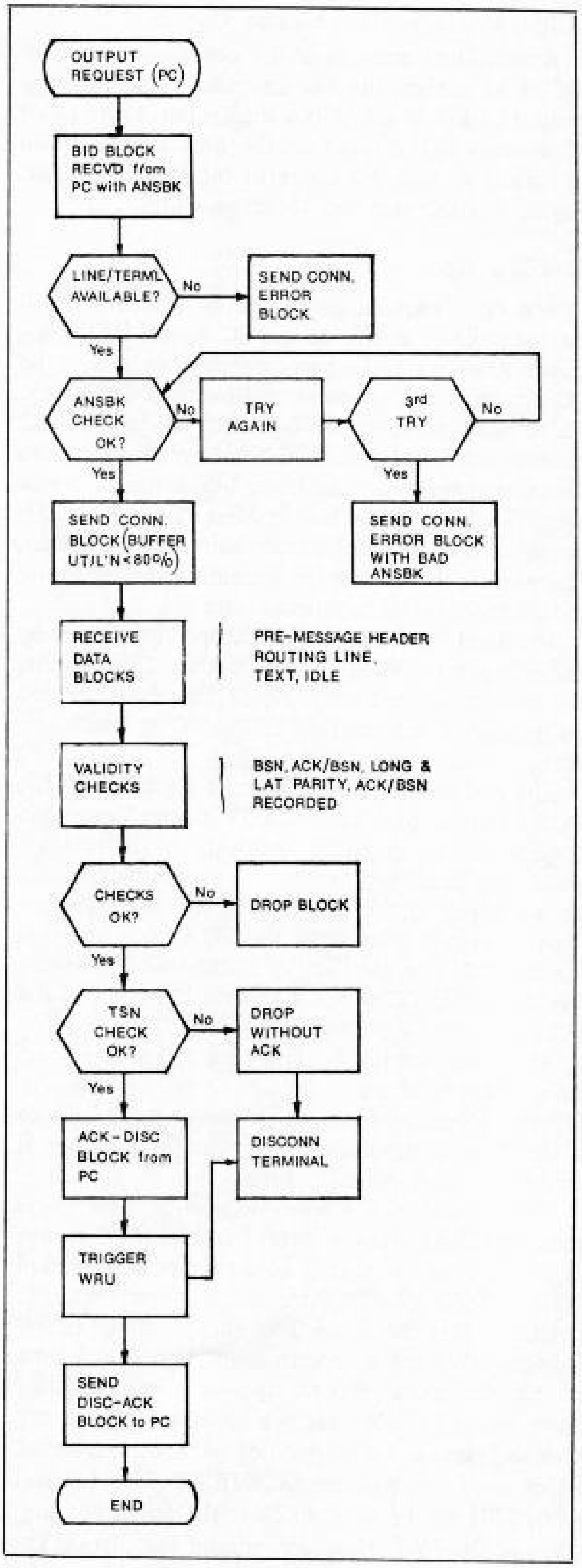


Figure 4—Flow Chart for Message Output

Block Flow Simulation Model

A typical ISCS-I site consists of a PC and a co-located CC. The latter is used to terminate local terminals, INFO-COM terminals, TWX and TELEX exchanges and TMS reperforator centers. Also, 2400 baud trunks linking the various other computer centers shown in Figure 2 are terminated at the CC. The PC is linked locally to the front-end CC through a fast inter-computer synchronizer.

A GPSS model or simulation model of an ISCS-I PC/CC site has been developed. Previously described message transmission control procedures have been incorporated such that the model may operate either as a stand-alone site or as part of a multi-node network in which the sites are linked by up to three high speed trunks.

The control mechanisms described in the flow charts in Figures 3 and 4 will generate the following data and control blocks for computer-to-computer transmission:

(i) Bid
(ii) Header
(iii) Data
(iv) EOT
(v) Connect
(vi) Ack/Disc
(vii) Disc/Ack
(viii) Idle
(ix) Transmision Error
(x) Bid Error
(xi) Data Error
(xii) Forced-Ack
(xiii) Connect Error

In GPSS, the structure of the system being simulated is described in terms of a block diagram drawn with a fixed set of predefined block types. Each block type represents a specific action that is characteristic of some basic operation that can occur in the real system. Connections between the blocks of the diagram indicate the sequence and combination of the actions that occur.

Moving through the system being modelled are certain basic units called transactions. In our system, transactions correspond to messages or blocks of data. The sequence of actions occurring in the system in real time is reflected in the movement of transactions from block to block in simulated clock time. Transactions remain at a block for an interval of time called the "action time." The action time can be specified by a mean time modified by a quantity randomly varying up to a maximum value in the range specified for the modifier or the action time may be specified through the use of a function. Any number of pairs of values (x, y) can be used in a table defining the function. The table can be interpreted in a continuous mode by assuming a linear variation between the points, thereby approximating any desired function with straight line segments. The table may also represent a step function with discrete transitions between the points.

Associated with the system being simulated are certain physical and control elements that operate on the transactions and direct their flow through the system. These are facilities, storages and logic switches.

A facility can represent any piece of equipment which can be seized by a single transaction at a time. A storage is an element which can be occupied by many transactions at a time. A logic switch is a two-level indicator that records the state of some system condition and on which certain logical decisions are based.

Statistics regarding the progress of the simulation are gathered automatically for the facilities and storages. To keep track of queues, the user specifies QUEUE blocks which may be used to measure average and maximum lengths and, if required, the distribution of time spent on the queue. Furthermore, a variety of statistics concerning various system variables may be tabulated by means of TABULATE blocks and the contents of SAVEX locations, which correspond to locations in which information of interest may be entered and saved, can be printed out during the course of the run.

GPSS System Outputs

Operation of a two site PC/CC system has been simulated using GPSS 2 in order to study ISCS-I network characteristics under various traffic loads. Statistics have been obtained for:

(1) I/O trunk utilization

(2) I/O trunk queues

(3) Occupancy level in AWT tables

(4) Average and maximum number of simultaneous call connections carried at each CC

(5) Percentage of calls which exceed a predetermined threshold for maximum simultaneous connections

(6) Percent utilization of processors for site-to-site communication

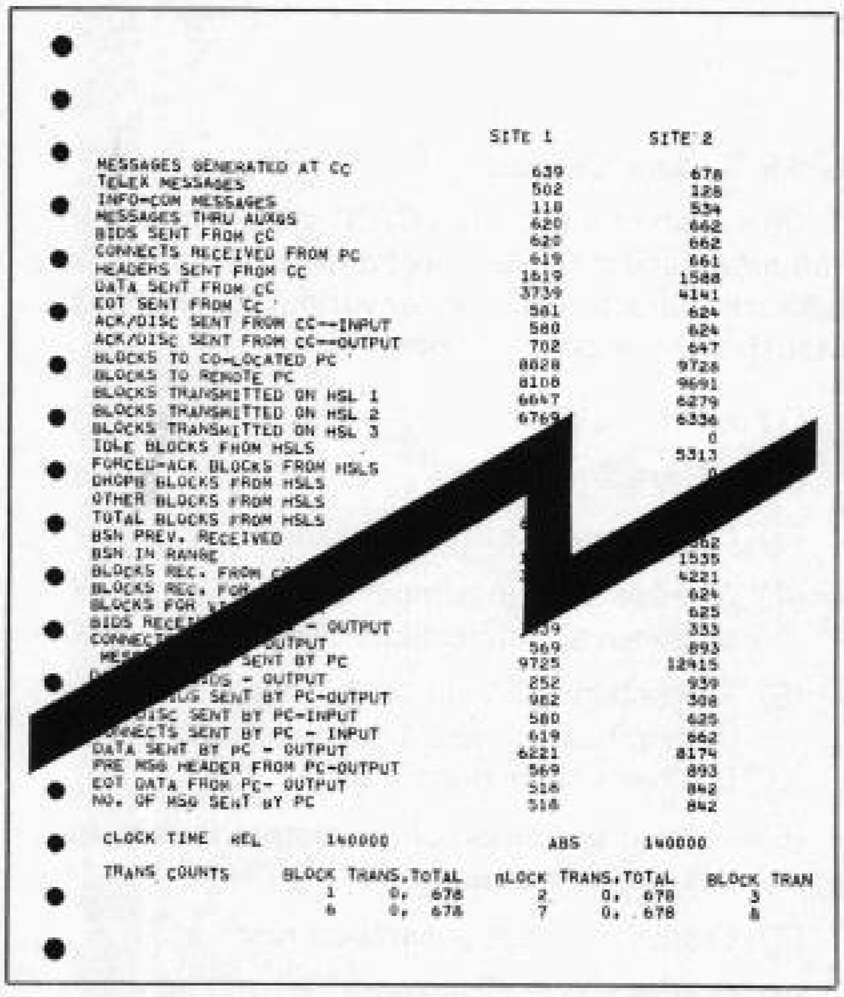
(7) Output message generation rate

(8) Output message queues

(9) Speed-of-service

A 600 block GPSS2 model for a PC/CC site has been generated by using about 450 blocks for the CC and 150 blocks for the PC. The statistics shown in Fig. 5 for a two site 1200 block system represent the state of the system after 23 minutes of simulated operation at a message input rate of 0.5 msg/sec per site. Exponential message size and appropriate message routing functions were used.

Sample formats in which the results of a simulation are presented are given in Fig. 5 and Fig. 6 for one representative run (A). As the simulation

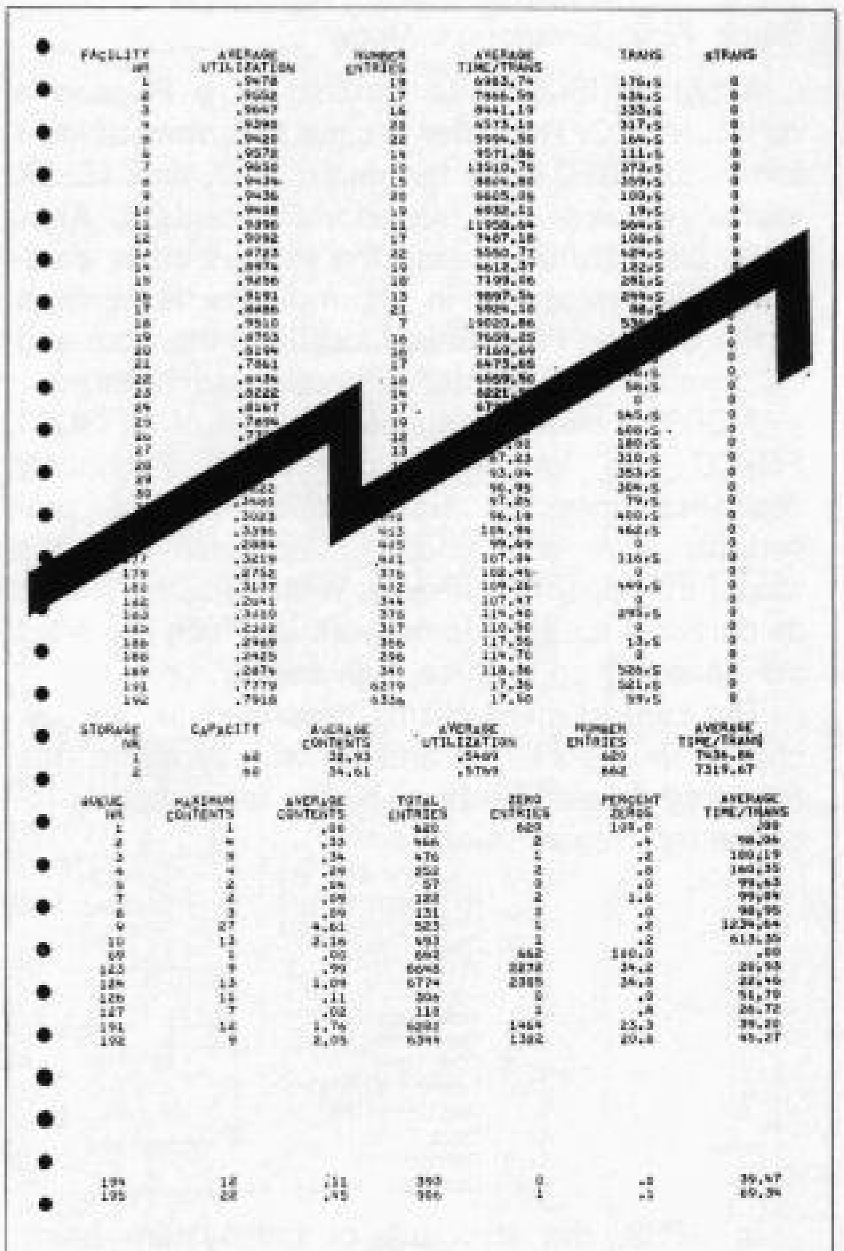


```
                                               SITE 1       SITE 2
   MESSAGES GENERATED AT CC                      639          678
   TELEX MESSAGES                                502          128
   INFO-COM MESSAGES                             118          534
   MESSAGES THRU AUXBS                           620          662
   BIDS SENT FROM CC                             620          662
   CONNECTS RECEIVED FROM PC                     619          661
   HEADERS SENT FROM CC                         1619         1588
   DATA SENT FROM CC                            3739         4141
   EOT SENT FROM CC                              581          624
   ACK/DISC SENT FROM CC--INPUT                  580          624
   ACK/DISC SENT FROM CC--OUTPUT                 702          647
   BLOCKS TO CO-LOCATED PC                      8828         9728
   BLOCKS TO REMOTE PC                          8108         9691
   BLOCKS TRANSMITTED ON HSL 1                  6647         6279
   BLOCKS TRANSMITTED ON HSL 2                  6769         6336
   BLOCKS TRANSMITTED ON HSL 3                                  0
   IDLE BLOCKS FROM HSLS                                     5313
   FORCED-ACK BLOCKS FROM HSLS                                  0
   DROPB BLOCKS FROM HSLS
   OTHER BLOCKS FROM HSLS
   TOTAL BLOCKS FROM HSLS
   BSN PREV. RECEIVED                                         662
   BSN IN RANGE                                              1535
   BLOCKS REC. FROM C                                        4221
   BLOCKS REC. FOR                                            624
   BLOCKS FOR                                                 625
   BIDS RECE               - OUTPUT                           333
   CONNEC                 OUTPUT                 569          893
    MES              SENT BY PC                 9725        12415
                  IDS - OUTPUT                   252          939
                  SG SENT BY PC-OUTPUT           982          308
               DISC SENT BY PC-INPUT             580          625
              NNECTS SENT BY PC - INPUT          619          662
   DATA SENT BY PC - OUTPUT                     6221         8174
   PRE MSG HEADER FROM PC-OUTPUT                 569          893
   EOT DATA FROM PC- OUTPUT                      518          842
   NO. OF MSG SENT BY PC                         518          842

    CLOCK TIME  REL      140000             ABS      140000

    TRANS COUNTS      BLOCK TRANS,TOTAL    BLOCK TRANS,TOTAL    BLOCK TRAN
                        1      0,  678       2      0,  678        3
                        6      0,  678       7      0,  678        8
```

Figure 5—Printout of the Record of Events for the GPSS Simulation



```
  FACILITY       AVERAGE        NUMBER        AVERAGE         TRANS     $TRANS
     NR        UTILIZATION      ENTRIES     TIME/TRANS
      1           .9478           19          6983.74         176.5       0
      2           .9082           17          7866.59         434.5       0
      3           .9647           16          8441.19         333.5       0
      4           .9398           21          4573.10         317.5       0
      5           .9420           22          5594.50         164.5       0
      6           .9578           14          9571.86         111.5       0
      7           .9650           10         13510.75         373.5       0
      8           .9436           15          8804.88         359.5       0
      9           .9436           20          6665.05         100.5       0
     10           .9488           19          4938.11          19.5       0
     11           .9896           11         11958.64         504.5       0
     12           .9092           17          7467.18         108.5       0
     13           .4783           22          5560.77         424.5       0
     14           .8974           19          6612.37         122.5
     15           .9456           18          7199.06         281.5
     16           .9191           13          9097.36         299.5
     17           .8486           21          5924.18          88.5
     18           .9510            7         19020.86         53
     19           .8753           16          7609.25                     0
     20           .8194           16          7169.69                     0
     21           .7861           17          6475.65                     0
     22           .8436           18          6989.50                     0
     23           .8222           14          8821.            56.5       0
     24           .8167           17          67                0         0
     25           .7696           19                          545.5       0
     26           .77             12                          600.5       0
     27                           13               92         180.5       0
     28                                            87.23      310.5       0
     29                                            93.04      383.5       0
     30             22                             96.95      304.5       0
                  .2485                            97.25       79.5       0
                  .3023                            96.10      400.5       0
                  .3396          413              106.96      462.5       0
                  .2884          425               99.49        0         0
                  .3219          421              107.94      316.5       0
    179           .4752          376              102.45        0         0
    181           .3137          432              109.26      449.5       0
    182           .2041          344              107.47        0         0
    183           .3489          376              114.90      295.5       0
    185           .2282          317              110.50        0         0
    186           .2969          356              117.58       13.5       0
    188           .2425          256              114.76        0         0
    189           .2874          340              118.36      520.5       0
    191           .7779         6279               17.36      321.5       0
    192           .7918         6326               17.50       59.5       0

  STORAGE     CAPACITY     AVERAGE        AVERAGE        NUMBER       AVERAGE
     NR                    CONTENTS     UTILIZATION      ENTRIES     TIME/TRANS
      1          60          32.93          .5489           620        7636.86
      2          60          34.61          .5769           662        7319.67

  QUEUE     MAXIMUM      AVERAGE      TOTAL       ZERO       PERCENT      AVERAGE
    NR      CONTENTS     CONTENTS     ENTRIES     ENTRIES     ZEROS      TIME/TRANS
     1          1           .00         620         620       100.0          .00
     2          4           .33         466           2          .4        99.06
     3          5           .34         476           1          .2       100.19
     4          4           .29         252           2          .8       100.35
     5          2           .04          57           0          .0        99.63
     7          2           .09         128           2         1.6        99.84
     8          3           .09         131           0          .0        98.95
     9         27          4.61         523           1          .2      1234.64
    10         13          2.16         493           1          .2       613.35
    69          1           .00         662         662       100.0          .00
   123          9           .99        6048        2078        34.2        20.55
   124         13          1.09        6774        2385        34.0        22.46
   126         11           .11         306           0          .0        51.79
   127          7           .02         118           1          .8        26.72
   191         14          1.76        6282        1464        23.3        39.20
   192          9          2.05        6344        1382        20.8        45.27



   194         12           .11         393           0          .0        39.47
   195         22           .45         969           1          .1        69.34
```

Facility 191—High-speed line No. 1
Facility 192—High-speed line No. 2
Storage 1—Site No. 1 input buffer
Storage 2—Site No. 2 input buffer
Queue 1—Site No. 1 input buffer overflow

Queue 2—, Queue 3—, Queue 4—, Queue 5— } Site No. 2 PC INFO-COM and TELEX Output processing

Queue 7 —, Queue 8 —, Queue 9 —, Queue 10— } Site No. 1 PC INFO-COM and TELEX Output processing

Queue 69—Site No. 2 input buffer overflow

Queue 123—, Queue 124— } Site No. 1 I/O queues

Queue 126—, Queue 127— } Site No. 1 I/O queue overflows

Queue 191—, Queue 192— } Site No. 2 I/O Queues

Queue 194—, Queue 195— } Site No. 2 I/O queue overflows

Figure 6—Printout of the Statistical Output for the GPSS Simulation

progresses, cumulative results, similar to those shown in Fig. 5 and Fig. 6 are printed out at specific time intervals to establish trends in the behavior of the system. From these data, graphs for Run A were plotted in Fig. 7 and Fig. 8 and graphs for Run B were plotted in Fig. 9 and Fig. 10. These graphs were used in further analysis to determine the duration of the simulation required to reach steady state conditions. Fig. 9 and Fig. 10 are included here to illustrate the manner in which changes in the model can be made with the simulation repeated (Run B) in order to study specific aspects of system behavior. Figure 9 and Figure 10 show the contrasting effects on the utilization of system elements resulting from the use of certain controls which limit the number of active output connections (slots) at the PC. For run (B) all other conditions present in run (A) were left unchanged.

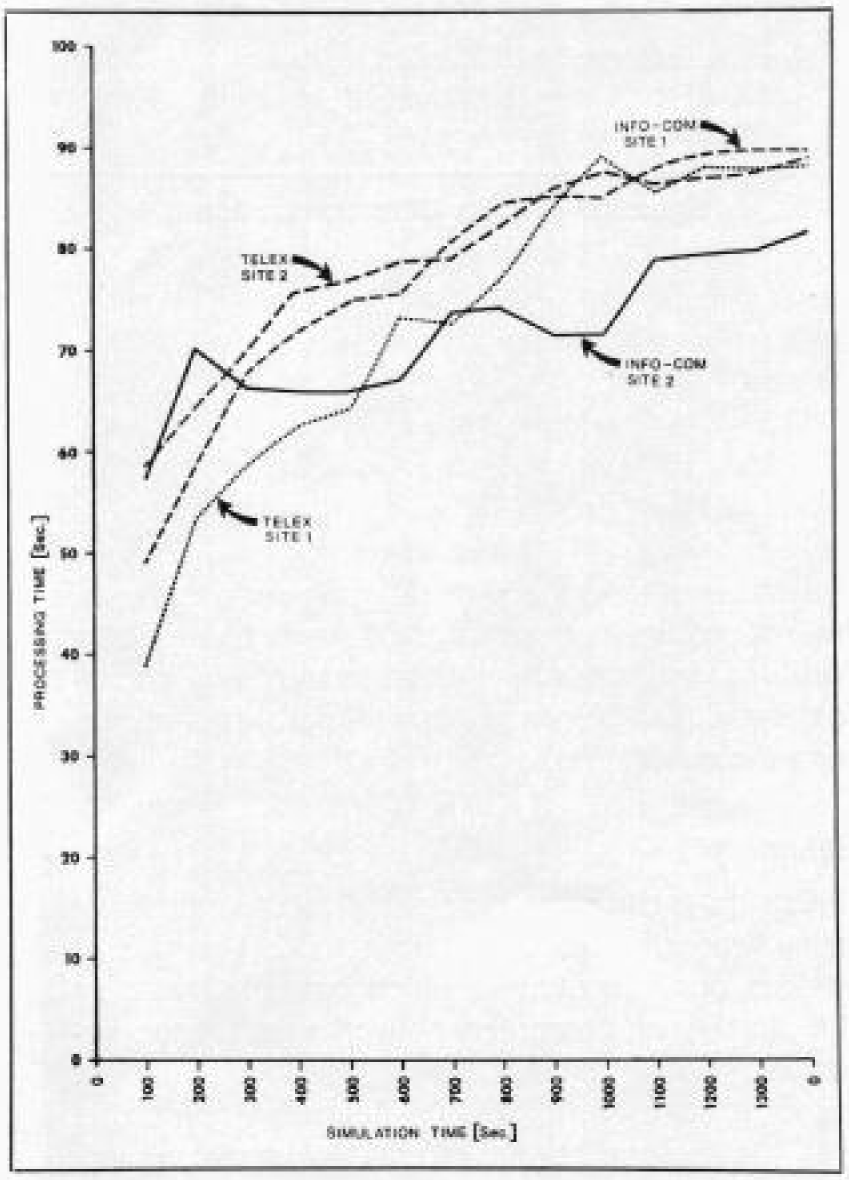


Figure 7—Input Processing Time for Run A

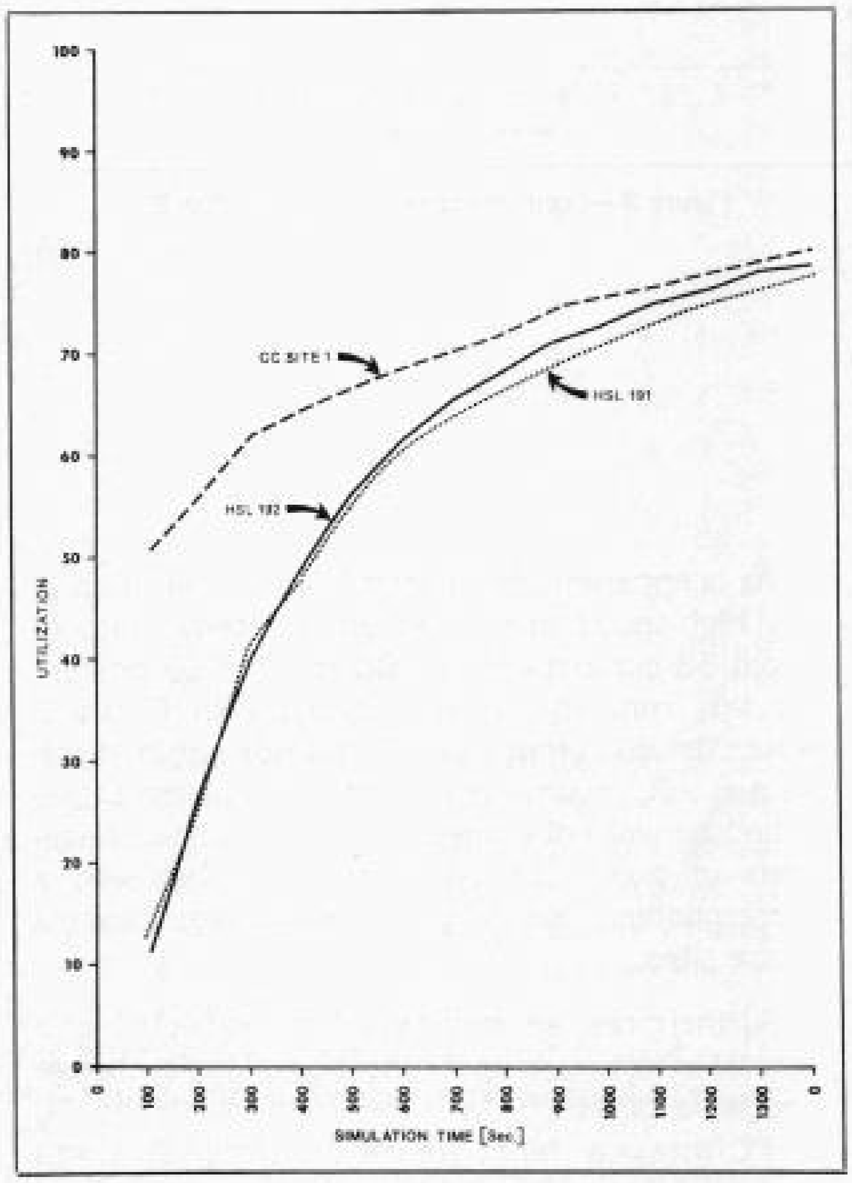


Figure 8—Facility Utilization for Run A

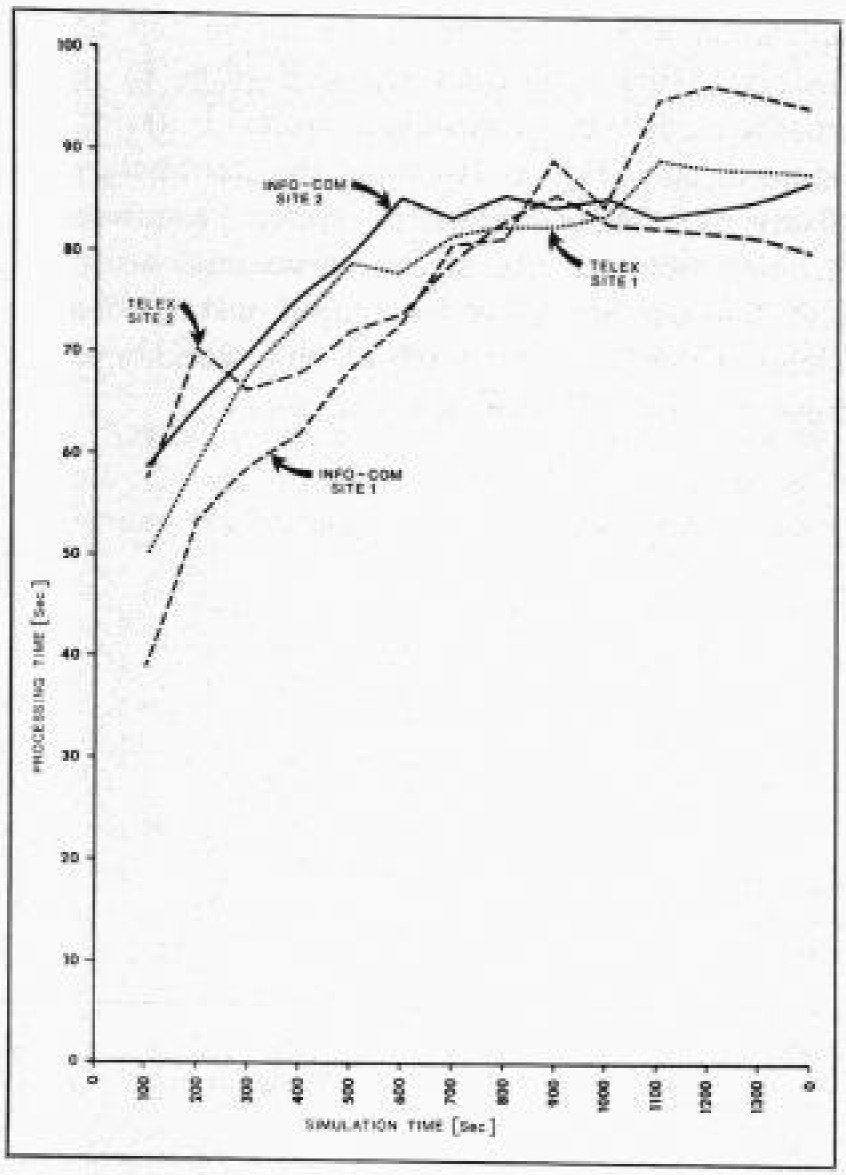


Figure 9—Input Processing Time for Run B

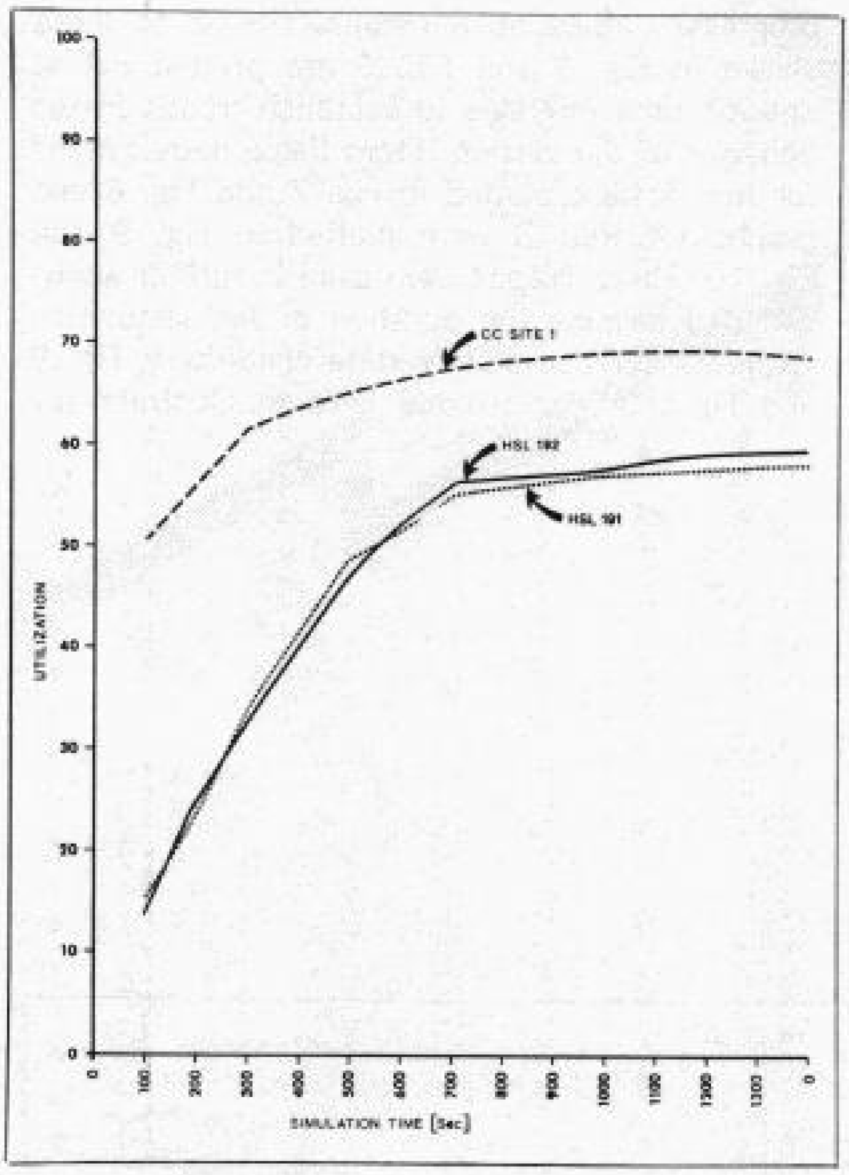


Figure 10—Facility Utilization for Run B

As is apparent from Figure 10, the utilization of the high-speed lines reached a steady state of about 58 percent and the CC reached 69 percent for this run. This is in contrast with Figure 8 where the 80 output slot PC limit was not in effect. As a result, much higher traffic flows were established between the sites resulting in line utilizations in excess of 78 percent coupled with a corresponding decrease in storage requirements at the sites.

Furthermore, as may have been expected of a control affecting output, the above logic change had little effect on the speed with which the CC to PC message input transit occurred. This was shown in Figure 7 and Figure 9.

Further studies have been made for a range of high speed line capacities and the effects resulting from loss of trunks have been analyzed by changing individual blocks as appropriate and re-running the simulation.

Summary

Design techniques applicable to system sizing have been described in this article. The use of various analytic models which are applicable to the design of computer networks has been included. Simulation has been introduced here as a tool used to verify design decisions connected with the implementation of complex systems such as ISCS-I.

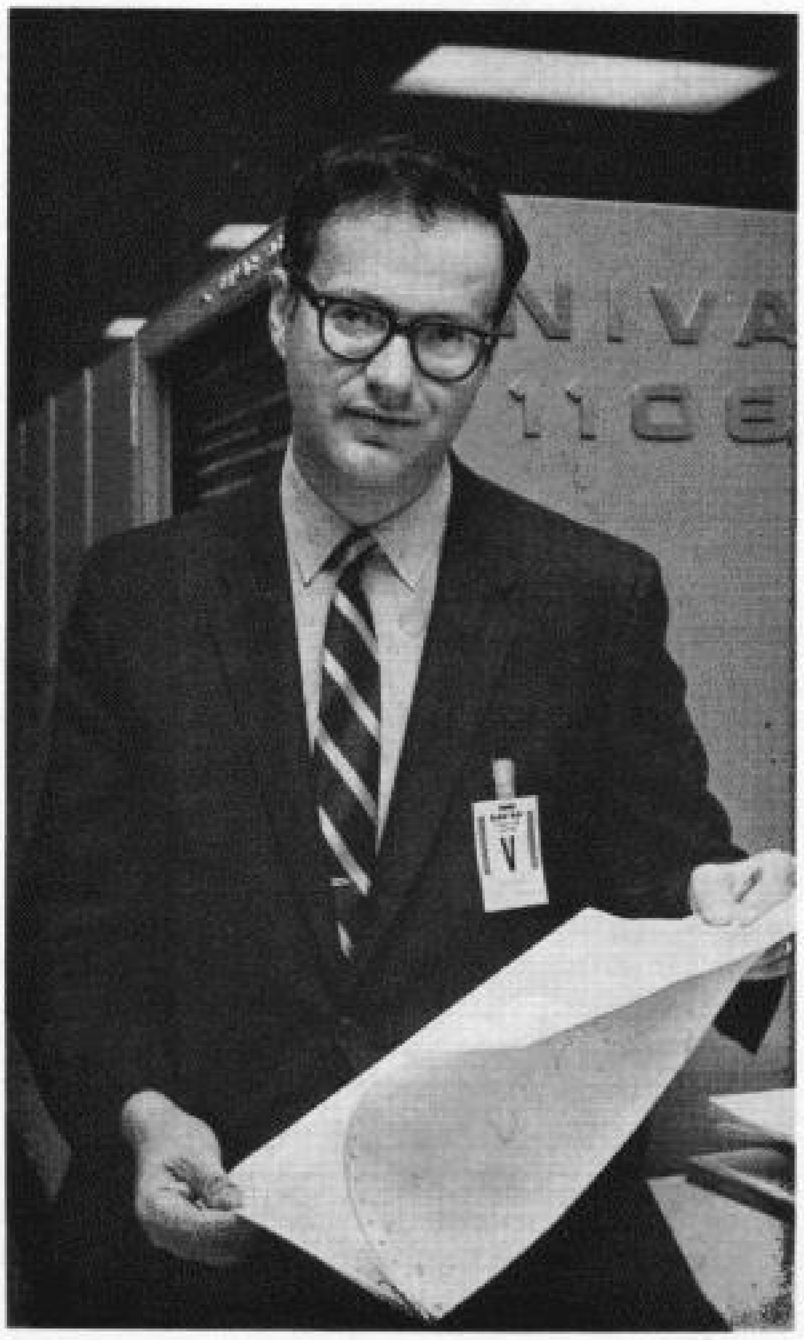
Acknowledgement

The author wishes to acknowledge the contributions made to this project of Harvey Mayerowitz shown in Fig. 10.

References

1. "Feasibility of Integrating Message Switching Networks and Circuit Switching Networks"—Western Union 66907-1, October, 1966.
2. "Network Cost of Access Program (NECAP)" SA/AS PD-02-00, January, 1968. (Western Union Internal Document)
3. "Modeling and Simulating the ISCS-I Processing Center," Western Union Technical Review—Winter, 1968.

Figure 10—Harvey Mayerowitz Discusses the Program with the Author



Leonard Stier, Manager of Special Communication Subsystem Design in P. & E.O. is responsible for network and error control subsystem and specialized interface design for computer systems.
He joined Western Union in 1966. His previous experience has been at Westinghouse working on the evaluation of radar systems.
Mr. Stier received a B.S. in 1962 from the University of Toronto. He also received an M.S. degree from Columbia University in 1966 and had done post graduate work in Operations Research at Columbia University.

Estimating Storage Requirements in a Store-and-Forward Switching System

by MILTON MORRISON and ANNE PANICCIA

Statement of the Problem

One of the problems in a real or hypothetical store-and-forward computer type message switching system accessed by means of half-duplex remote terminals, is to determine its storage requirements. Ignoring, for the moment, multiple address or group code type traffic, every message which enters the system on one terminal creates another message which leaves the system at some other terminal unless prevented from doing so because of a system failure. A half-duplex line can transmit messages travelling in both directions, but only in one direction at a time. Therefore, if a terminal is inputting a message to the system, or accepting a message from the system, any other message which must be output on that terminal must wait on queue, until the destination terminal is free. This queue is referred to as the "output message queue." It is the purpose of this article to describe the time dependent size of this output queue, so that adequate computer storage allocations can be provided.

It is also possible that messages may be waiting to be input at a terminal, similar to people standing in line waiting to use the terminal. These messages wait in what is referred to as an "input queue." This input queue is not a physical part of the computer system and makes no storage demands on it. Therefore, it needs no physical consideration in system design, but it must be considered in the design for its effect on the system's output queuing condition.

Input traffic units (messages) arrive at the terminal at a variable time rate. The hypothetical system gives input priority over output; i.e., if a message is waiting to be output at a particular terminal, the system will suppress output so that any and all messages on queue for input can be serviced. Thus, if a second message is to be input, then the output message (waiting on queue) will again have to wait until the message is through; this condition continues until that point in time when the input queue is zero. Then output can begin and will not be interrupted until that message has been delivered to the terminal.

Since all this occurs at a single terminal, storage must be provided, within the computer system, for the output messages of the entire set of terminals supported by the computer system. Naturally, during the daytime hours, the amount of occupied storage fluctuates, as the system traffic load rises and falls. It is a problem to determine the amount of computer system storage which should be supplied to accommodate the queue of output messages, with real assurance that the recommended storage will be adequate. Since the queue size, at any instant, is a random variable, the problem must be examined on a probabilistic basis. If attention is fixed on any one instant of time, the number of messages on the output queue, at that instant, is the sum of the numbers of messages on all individual output queues for the terminals associated with that computer. This sum will vary during that portion of the day during which messages enter and leave the system. The basic problem is to determine a value so that the probability of the sum exceeding this value is equal to a pre-selected small number.

Before moving to the mathematical description of the problem statement, one more *very critical assumption* must be made. It will be assumed that every terminal serviced by the system experiences the same diurnal variation in traffic. That is, consider an arbitrary time interval (t_a, t_b): the percentage of a particular terminal's total daily input traffic is the same for all terminals during this time interval (neglecting quantization errors).

These statements will be more meaningful to the reader when contemplated in connection with the traffic profile.

Typical Daily Traffic Profile

In order to solve numerically for the output queueing situations, one must first review the manner in which traffic arrives at a terminal during the day. A typical traffic profile has been selected for application in our queueing analytics.

An input traffic profile of messages actually arriving at Western Union's ISCS Phase I systems in New York and Chicago in early 1969, is shown in Figure 1. Each bar represents that portion of the total daily input traffic which has arrived in that time interval. Between 0900 and 0930, for example, Figure 1 reads 2.9 percent. This means that a terminal received 2.9 percent of its total daily input traffic in that time interval. Notice that this interpretation does not depend on the actual volume the terminal handled; whether a terminal is highly utilized or barely utilized, it still handles 2.9 percent of its daily input traffic in this half hour. This same traffic profile therefore applies to all terminals and thus to the entire system.

Summing all the bars, we observe that at time 2100, the system has received 98.1 percent of its total daily input. If the graph were extended to a 24-hour day, the sum would accrue to 100%.

The total input traffic will be divided among the individual terminals. This apportionment can be varied, but for purposes of easy accounting and description, it will be assumed that the terminals are numbered 1, 2, 3, . . . , H, and the amount of input received from each terminal is monotonically increasing with the terminal number. A simplified model would make that increase proportional to the terminal I.D. (Identification), such that, if we let the total traffic received at the jth terminal be C·j, where:

C = constant

H = number of terminals

I = total input traffic rate being received by all terminals

$$\sum_{j=1}^{H} C \cdot j = I \tag{1}$$

and

$$\sum_{j=1}^{H} j = \frac{H(H+1)}{2} \tag{2}$$

From Equations (1) and (2), it follows that

$$C = 2I/H(H+1) \tag{3}$$

Thus, input traffic rate being received at the jth terminal would be equal to $2I/H\ (H+1)$. This expression applies even if I were the traffic received up to any time t, rather than an expression of traffic rate.

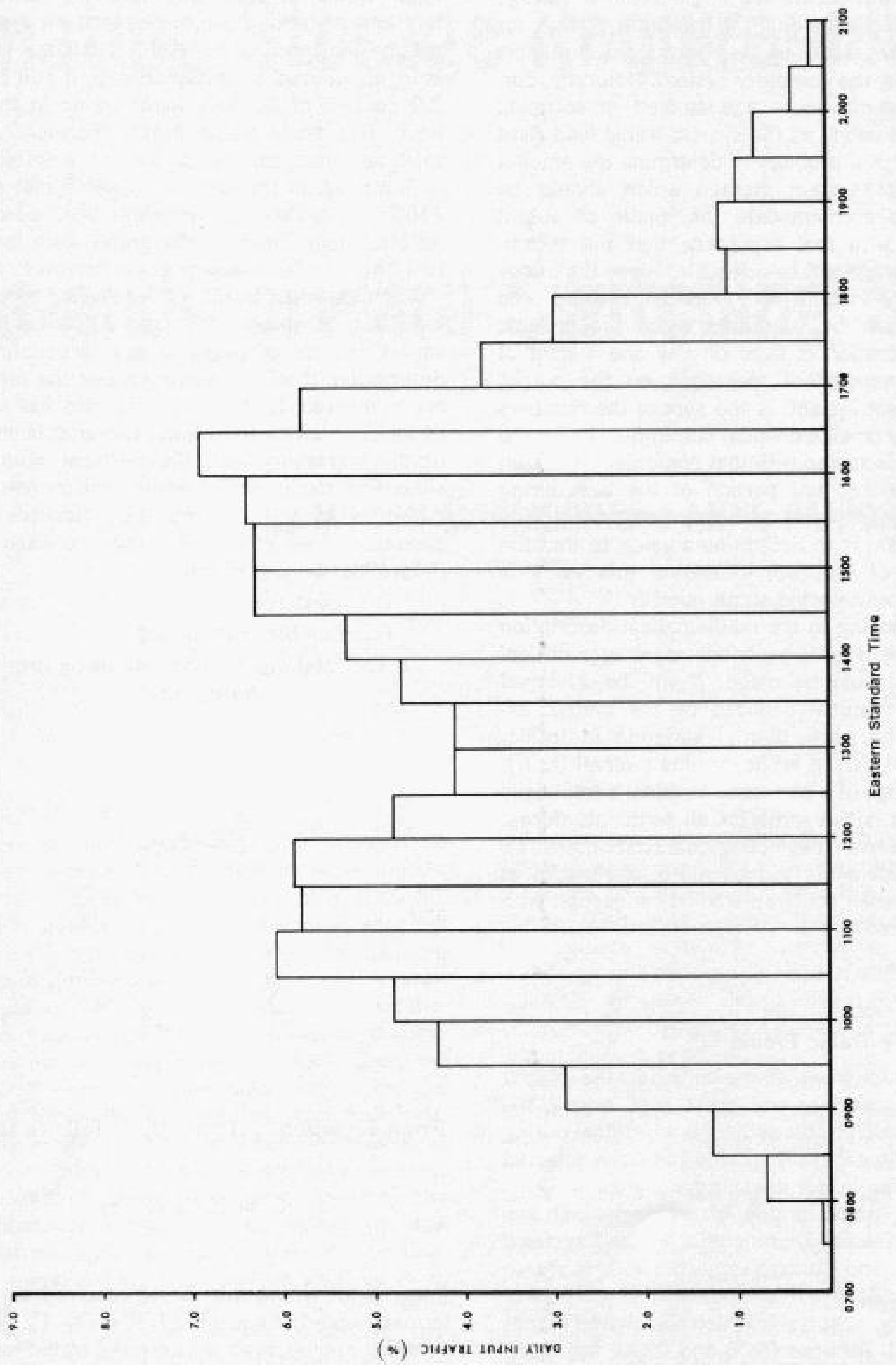


Figure 1—Daily Input Traffic Profile for TCCS/INFO-COM Service

Definitions and Assumptions

The following definitions and assumptions are necessary for the understanding of these derivations. In all definitions, the subscript j may be dropped if no ambiguity results.

(a) The mean rate (msg/unit time) at which messages arrive at the jth terminal (line) for input to the system is designated $\lambda_{Ij}(t)$. It will also be referred to as the mean input rate. The mean input rate is a function of time.

(b) An analogous definition holds for the output message rate, λ_{Oj} (t). The subscripts I and O on any symbol refer to input and output respectively.

(c) The mean total rate (in msg/unit time) at which messages arrive at the jth terminal (line) for input or output is designated λ_j (t), and will be referred to as the mean total traffic rate; $\lambda_j(t)=\lambda_{Ij}(t)+\lambda_{Oj}(t)$

(d) The probability that a message currently being processed will complete its input or output processing during a small time interval, $\triangle t$, is μ ($\triangle t$) where μ is assumed to be constant and in this analysis, is the same constant for all terminals (lines).

(e) The probability that a message will present itself for service at the jth terminal (either input or output) in a small time interval, $\triangle t$, is $[\lambda_j(t)](\triangle t)$

(f) After a period of time, λ_j (t) declines to zero for all j.

(g) A message is said to be on queue whenever it is either being processed by the terminal or waiting to be processed.

(h) All queues are zero at t = 0.

(i) The aggregate number of messages on all output queues and on all input queues may be taken as the sums of all the individual output or input queues.

(j) The mean number on queue at the jth terminal is $M_j(t)$ and the variance of the number on queue is $V_j(t)$. The mean and variance of the aggregate are designated M(t) and V(t) respectively.

(k) The number of terminals (lines) involved in the message switching system is H.

Single Terminal

Consider a single terminal. If all of the above statements hold and messages arrive at the terminal at a lesser mean rate than the mean rate at which they are processed out, the typical queueing situation exists. Theory applicable to such queues is available, and, in particular the mean and variance of the number on queue, have been derived for those queueing systems having the following properties:

1) There is one channel or terminal.
2) Time between arrivals has an exponential distribution with the mean $=\frac{1}{\lambda}$; λ is the arrival rate and is a constant (i.e., not a function of time) over sufficiently short intervals of time.
3) Processing time is exponentially distributed with mean $=\frac{1}{\mu}$; μ is the processing rate and is a constant (i.e., not a function of time).
4) Messages are processed in order of arrival.
5) $\lambda < \mu$.
6) The terminal is in a steady state condition; that is, it is no longer affected by the conditions which existed at the initiation of operation.

Mathematical Model

Let ρ equal the proportion of time a terminal is utilized. ρ is equal to mean processing time divided by mean inter-arrival time. Hence, $\rho = \frac{1/\mu}{1/\lambda} = \frac{\lambda}{\mu}$; it is also called the traffic intensity.

It is shown in References 1 and 2 that under the above circumstances the mean number on queue is

$$M = \frac{\rho}{1-\rho} \tag{4}$$

and the variance is

$$V = \frac{\rho}{(1-\rho)^2} = M(1+M) \tag{5}$$

If $\lambda > \mu$, then $\rho > 1$, and the above formulae no longer hold, since ρ is no longer the occupancy of the line; but it can still be regarded as the traffic intensity. It is noted that when $\rho < 1$, there are periods during which no messages are being processed; i.e., the terminal has open time. If, however, $\rho > 1$, the terminal will, in general, be processing continuously; and furthermore, there will be an increase of the queue size due to the fact that messages are arriving for service at a greater rate than they can be handled.

Basically, then, two modes of operation must be recognized: 1) the queueing mode, when $\lambda < \mu$ and the terminal has open periods; and 2) the fully utilized mode, when $\lambda > \mu$, and the terminal is assumed to be processing full time.

When a terminal is in the queueing mode, there is no problem in determining the mean and variance of the number on queue. The formulae stated above, for mean and variance respectively are applicable.

It will also be necessary to obtain the mean and variance of the number of messages on queue, when the terminal is in the fully utilized mode. In this mode, the time between the completion of the processing of messages has an exponental distribution.

Let $x =$ number of messages which arrive in (t_1, t_2) during which interval the system is in the fully utilized mode. Let $y =$ the number of messages which are processed during this time. Both x and y have Poisson distributions with means equal to $\lambda\ (t_2 - t_1)$ and $\mu\ (t_2 - t_1)$ respectively. Suppose the number of messages on queue at the inception of the fully utilized mode is z and its expected value is b. Then the expected value of the number of messages on queue at t_2 is the expected value of $z + x - y$.
Symbolically:

$$E\ (z + x - y) = b + E(x) - E(y).$$

Assuming z, x, and y are independent,

$$V\ (z + x - y) = V(z) + V(x) + V(y).$$

Since the variance of a Poisson distributed variate is equal to the mean,

$$V\ (z+x-y) = V(z) + E(x) + E(y).$$

Transfer Between Modes

When ρ is close to, but less than unity, the queueing mode formula for the mean gives very large values. For example, when $\rho = .99$, $M = \frac{.99}{1-.99} = 99.0$. Intuitively, one does not immediately anticipate that the mean (i.e., expected) queue at a terminal which is processing messages about as rapidly as it is receiving them should necessarily be high.

The explanation lies in the nature of the expected value as a characterization of a probability distribution, and in the fact that the formula used for the mean is for a steady state condition which cannot be achieved in a short period of time. (The same environment applies to the formula for the variance.)

If the queueing mode formula for the mean gives a large value, it must satisfy a test which determines whether or not it has had enough time to reach that value. If it has not, then the value given by the formula must be rejected and replaced by a number calculated by a method which takes into account the fact that steady state conditions could not have been achieved in the available time.

This may best be illustrated with a numerical example. In Table I, the traffic statistics for 7 time intervals, 30 minutes each, on a typical terminal are tabulated.

Column 2 of Table I lists the mean number of messages arriving for processing (both input and output) at a hypothetical terminal during successive 30-minute intervals. For this terminal, the mean message processing time is $\frac{1}{\mu} = .933$ minutes/msg. The message arrival rates, λ, are equivalent to the numbers in column 2 divided by 30.

Column 4 is the traffic intensity $\rho = \frac{\lambda}{\mu}$
Column 5 is the mean number of messages in queue (M), calculated from the queueing formula in equation (4).

Column 7 is the variance computed from the queueing formula in equation (5).
Columns 8 and 9 are the means and variances, finally computed. These are the essential characteristics of the traffic pattern.

In the 2nd, 3rd and 4th interval it is observed that the difference in volume between two time intervals shown in Col. (3) exceeds the difference in mean queue estimate (M) between two time intervals, as shown in Col. (6). From the 4th to the 5th interval, the converse is true ($9.8 < 12.91$). This complication is explained as follows:

From the 4th to the 5th interval, the rate increased from $\frac{20.3}{30}$ msg/min. to $\frac{30.1}{30}$ msg./min. The increase in rate is $\frac{9.8}{30}$ msg./min. This in-

TABLE I

(1) Successive Intervals	(2) Mean Numbers of Messages	(3) Difference Between Means	(4) Traffic Intensity	(5) Mean Number of Messages in Queue	(6) Difference Between Mean Queue Estimate	(7) Variance	(8) M	(9) V
1st	.7		.0218	.0223		.023	.022	.02
		4.2			.157			
2nd	4.9		.152	.179		.211	.179	.21
		4.2			.216			
3rd	9.1		.283	.395		.551	.395	.55
		11.2			1.31			
4th	20.3		.631	1.71		4.63	1.71	4.63
		9.8			12.91			
5th	30.1		.936	14.62		228.4	11.51	144.0
		2.6						
6th	33.6		1.045				12.96	209.75
		9.1						
7th	42.7		1.328				23.5	284.6

crease in rate operating over 30 minutes would introduce into the system an additional 9.8 messages on the average, over and beyond that entered by the previous rate which provided a mean queue of 1.71. Hence, in the time interval of 30 minutes the mean queue should not have increased more than 9.8 and indeed would probably have increased by less.

The formula dictates that the mean has increased by 12.91, but this is improbable. The apparent contradiction is explained by the fact that the formula is applicable to steady state conditions which could not have been achieved in the 30-minute period. Thus, we reject the value given by the formula and take the mean queue equal to $1.71 + 9.8 = 11.51$. The terminal is still in the queueing mode and the variance is computed from:

$V = M(1+M) = (11.51)\ (12.51) = 144$
which appears in row 5 of Column 9.

In the sixth time interval, $\rho > 1$ and the terminal passes into the fully utilized mode. During the sixth time interval, it is seen from Column 2 that 33.6 messages entered for processing. Since in this period it is assumed that the terminal is fully utilized, $(\frac{30}{.933}) = 32.15$ messages were processed (on the average). Thus, $33.6 - 32.15 = 1.45$ messages were added to the queue which existed at the beginning of the sixth time interval. Hence, the mean queue at the end of the 6th time interval is $11.51 + 1.45 = 12.96$ the value for M, listed in Column 8.

Using $V(z + x - y) = V(z) + E(x) + E(y)$, the variance of the queue size at the end of the sixth interval is $144 + 33.6 + 32.15 = 209.75$.

In the interval 7, the terminal is still in the fully utilized mode so that the mean is $12.96 + (42.7 - 32.15) = 23.51$ and the variance is $144 + (33.6 + 42.7) + 2(32.15) = 284.6$.

The above example displays the logic for the computation of an individual terminal's time dependent queue. The queue starts at zero and builds up in the queuing mode (as traffic rises) until it is sensed that the time rate of change of the mean queue given by the queuing formulation exceeds the volume rate of change. At this point the mean queue computation switches to an accumulation of excess message rate over and above that which was operative in the queuing mode. The situation reverses itself under conditions of decreased traffic, except that all queues are allowed to deplete before switching to the kueing mode formulation.

Terminal Queues and Message Storage Requirements

The random variable of particular interest is the total number of messages on queue (as a function of time). A theorem of mathematical statistics, the central limit theorem, states roughly that the probability distribution of a sum of n independent random variables approaches the normal distribution as $n \to \infty$, with mean equal to the sum of the means and variance equal to the sum of the variances.

In our system the n independent variables are the queues at the n terminals. It follows that the total number of messages on queue at any time, t, can be regarded as normally distributed and

$$M(t) = \sum_{j=1}^{H} M_j(t) \qquad (7)$$

$$V(t) = \sum_{j=1}^{H} V_j(t) \qquad (8)$$

The mathematical model is required to find the number, K, such that the probability of a normally distributed variate with mean M(t) and variance V(t), exceeding K is ϵ, where ϵ is a small positive number.

Thus far, it has been tacitly assumed that all the messages on both input and output queues would have to be stored. However, as noted previously, only output messages require computer storage; the input messages queue outside of the system. To handle this situation, we first compute the mean and variance of the total queue for a terminal [i.e., M_j (t) and V_j (t)] using the total message rate

$$\lambda_j(t) = \lambda_{oj}(t) + \lambda_{Ij}(t) \qquad (9)$$

This overestimates the required computer storage by the amount queued on the input side. Then, we compute the mean and variance of the input queue using only the input message rate λ_{Ij} (t) which ignores any interference on input due to output messages. Since the priority structure on our system prevents messages on output from interfering with input unless the message to be output is already in the course of output transmission, we are confident that the above computation underestimates the average input queue but by an amount less than one message per line. Thus, we take a mildly pessimistic approach and subtract the mean input queue from the total queue to estimate our output queue. That is, with an error less than one message unit per line or terminal, the following equation is true:

$$M_{0j}(t) = M_j(t) - M_{Ij}(t) \qquad (10)$$

With regard to the variance, the situation is not quite as simple. However, we note that the covariance between the input and output queues will tend to be positive (more on input implies more on output). This being true, we can compute $V_{0j}(t) = V_j(t) - V_{Ij}(t)$ and be confident that the estimate of the output queue variance is conservative.

An Example

To illustrate the method of obtaining message storage requirements from terminal queues, let us assume the following data:

1) 20 terminals
2) 56 sec = .933 minutes, mean processing time for both input and output messages
3) Input message rate equals output message rate for all terminals
4) 6000 messages total input
5) The traffic profile, derives from Figure 1 with some adjustments to make the total percentage equal to 100. This is tabulated in Table II.
6) Daily input traffic at the jth terminal is given by

$$\frac{2j(6000)}{(20)(21)} = 28.57j \qquad (11)$$

where $j = 1, 2, \ldots 20$

It is required to obtain the amount of storage required so that, during that period of the day at which the storage is most heavily utilized, the probability that a message will find no storage available is .001.

TABLE II

Successive Intervals	*Total Traffic (percent)*
1st	.1
2nd	.7
3rd	1.3
4th	2.9
5th	4.3
6th	4.8
7th	6.1
8th	5.8
9th	5.9
10th	4.8
11th	4.1
12th	4.1
13th	4.7
14th	5.3
15th	6.3
16th	6.3
17th	6.4
18th	6.9
19th	5.8
20th	3.8
21st	3.0
22nd	1.1
23rd	1.2
24th	1.0
25th	.8
26th	.4
27th	.2
28th	.7
29th	.7
30th	.5
31st	.0
32nd	.0

A program was written to effect the mathematical model described in this example. The partial printout of this program, shown in Fig. 2, lists the mean and standard deviation of the total number on queue at all terminals for the entire day. The columns of particular interest are those labeled "FINAL MEAN" and "STANDARD DEVIATION." Note that the largest mean occurs during the 20th time interval at the end of which interval the mean is 2500.055 and the standard deviation is 122.346. From tables of the normal distribution, it is noted that the probability that a normally distributed variate is greater than the mean plus 3.09 standard deviations is .001. Hence, if storage is provided for $2500 + 3.09\ (122.3) = 2500 + 378 = 2878$ messages, the requirements for providing sufficient storage will have been satisfied. Thus, we are confident that we shall not need a storage greater than 2878 messages.

Further Application

While this example was based on the assumption that all terminals followed the same traffic profile, this need not be always true for the method to be valid.

In fact, all terminals could have different traffic profiles, message processing rates, and output-input ratios. The subscribers need not be terminals, but could be lines leading to other message producing and/or receiving sources. As long as it is possible to compute the mean and variance of the queue of messages related to the terminal or subsystem, the model and techniques described above are applicable.

References

1. "Saaty, T., Elements of Queuing Theory," McGraw-Hill Book Co., Inc., New York, 1961.
2. IBM Technical Publications Department, "Analysis of Some Queuing Models in Real-Time Systems," F20-0007-1.

Acknowledgement

The authors acknowledge, with thanks, the contribution of Mr. E. D. Smith of the Scientific Programming group in P&EO, who programmed the model on Western Union's Univac 1108 computer at Mahwah, N. J.

Milton Morrison, Senior Staff Analyst in Systems Design and Analysis, in P&EO, is responsible for the probabilistic aspects of communications systems design.

Mr. Morrison was associated with Stevens Institute for 11 years as teacher of mathematics and statistician to the Davidson Laboratory; he came to Western Union after ten years with ITT.

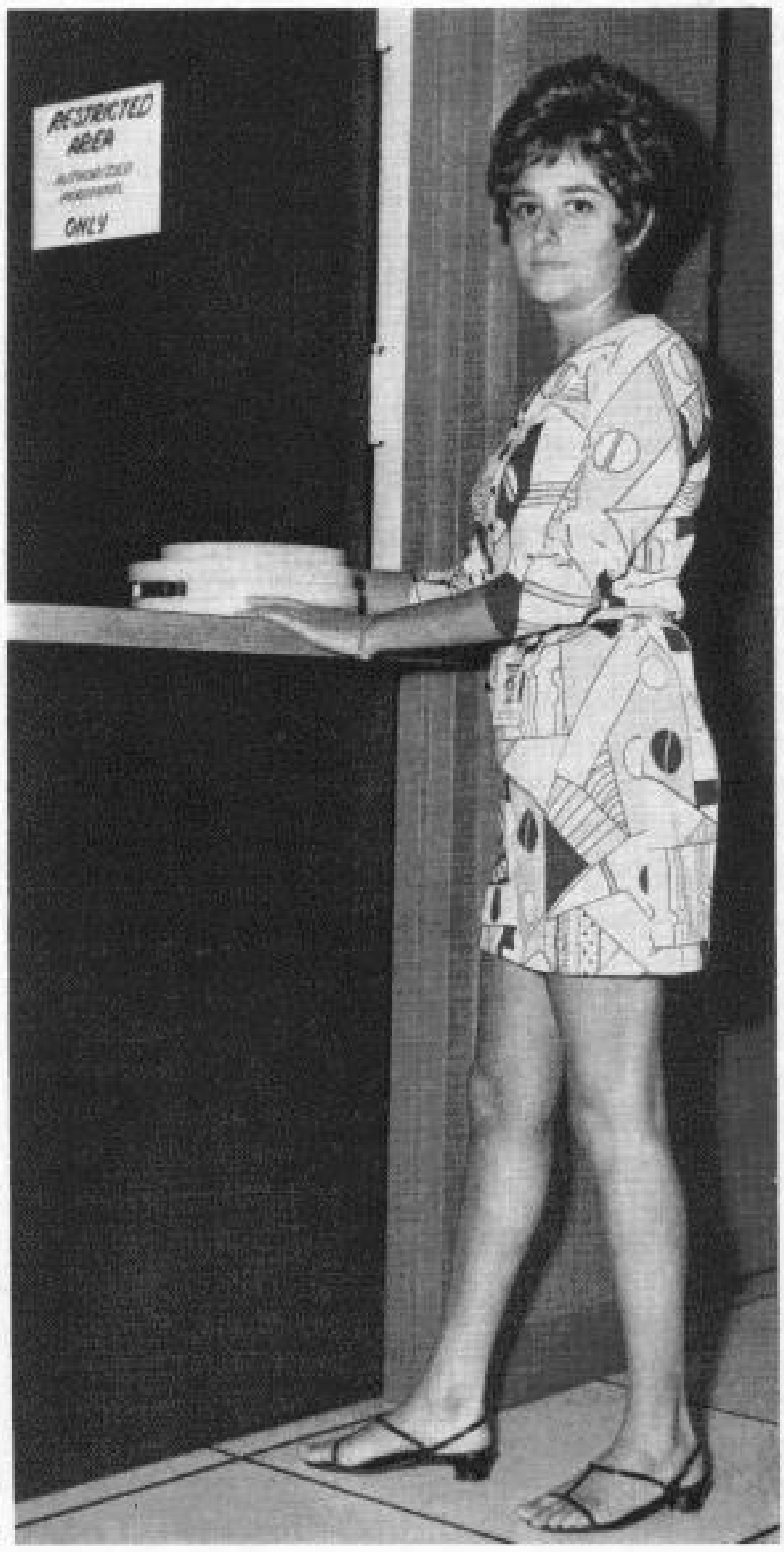
He holds a BA degree from Montclair State College and an MA from Columbia University, and is a member of the American Statistical Association, Institute of Mathematical Statistics, and Operations Research Society of America; he was a delegate to the Fifth International Teletraffic Congress. His previous publications on applications of probability and statistics have appeared in Biometrics, Technometrics, IEEE Transactions on Military Electronics and other journals.

TIME INTERVAL	*FINAL MEAN*	*STANDARD DEVIATION*
1	.194	.764
2	1.734	2.327
3	4.378	3.824
4	51.510	27.121
5	137.818	40.451
6	243.069	49.633
7	449.554	62.743
8	645.489	69.382
9	845.663	76.792
10	958.609	79.080
11	1004.672	79.507
12	1046.272	83.702
13	1137.406	88.479
14	1278.838	94.053
15	1503.377	101.478
16	1737.788	107.281
17	1979.305	113.003
18	2258.264	119.588
19	2463.162	123.207
20	2500.055	122.346
21	2451.678	123.283
22	2231.672	115.985
23	2040.939	117.649
24	1834.849	113.441

Figure 2—Partial Printout for the Mathematical Model

Anne Paniccia, Junior Systems Analyst in Systems Design & Analysis, P&EO, has previously been a trainee at Western Union and is currently working on her degree in Master of Science at Courant Institute of Mathematical Sciences/New York University. The subject of her Thesis: Queuing theory in a time-sharing environment with priority.

Miss Paniccia joined Western Union in 1968 and since then has been involved in the comparative evaluation of compiler and assembler languages available on the Univac 1108, and in the modelling and implementation of systems simulations conducted in GPSS.



TPUT— A SOFTWARE DIAGNOSTIC PROGRAM FOR ISCS

by R. D. O'MEARA

A diagnostic, is an aid in diagnosis, simply stated. In computer programming, a diagnostic is a special purpose program designed to test and evaluate a "given system." This "given system" can vary from the relatively simple (a light bulb) to the very complex (a multi-computer system). Of course, the more complex the system the more care must be used in designing the diagnostic. Western Union has one such complex system called ISCS, for which TPUT was designed and found very useful in measuring the performance this ISCS service.

As part of their modernization plan, Western Union is automating its record communication services. One of these services is the ISCS (Information Shared Communication Service), a shared, store-and-forward, message-switching system, that will eventually service many of the terminals such as TMS, TLS, TWX, and ICS [Telegram Message Service, Telecommunications, Teletypewriter, and Information Communication Service (private)]. ISCS is currently in its Phase I state which consists of three main segments:

(1) the terminals,

(2) the communication lines (COM LINES), and

(3) the computer complex where all the message routing is performed.

Computer Complex

The ISCS computer complex, in Phase I, is actually four separate, linked computer centers. Communication line terminals and two Univac 418 computers comprise the main equipment at each computer center. The communication line terminals or termination (CLT) units are contained in large arrays called a multiplexer, MUX. Several of these MUXs and the hundreds of lines they terminate form the interface between the communication lines and the Communication Center (CC) computer. The electronic signals put on the "com" lines by use of the terminals are interpreted by the MUX for the CC computer. This linkage allows a user to type a character on his terminal device and have the computer receive it in an understandable format that can be processed by the various programs (software) in the two computers at each center or site. The main function of the CC computer is to collect and concentrate the characters it receives and to send the accumulations to the "second" 418 computer, the Processing Center (PC). While the CC deals in characters, the PC works with accumulations of characters, defined as messages. The store-and-forward characteristic of ISCS is inherent in the design of the PC software. A message is received and accepted; then it can be stored on some large storage device for later processing and delivery. As the message load increases, the time to deliver each message lengthens a little; until all of the "com" lines are being used and no new traffic can start. This point is the design maximum response time and is usually quoted to the customers as the time required to deliver an average message or "speed of service".

How ISCS Works

To a subscriber, ISCS works as follows: A message is sent from his terminal over the lines into the computer. The computer analyzes the destination(s), performs any code or speed conversion required, creates records for billing and retrieval, and delivers the message over the lines to the correct terminal. The complexity arises from the handling of hundreds of messages in parallel; some with errors, most without errors and verifying that all requests were satisfied.

The simultaneous mixing of large numbers of messages in a computer system that temporarily might fail, requires a highly reliable message-accounting system to insure complete delivery of all accepted messages.

As the above illustrates, ISCS is a difficult system to evaluate because there are several major subsystems that obviously must be verified before they can be tested with other sections of ISCS. The testing of a system like the computer section of ISCS is almost as complex as the ISCS itself. A growing system like ISCS needs to be continuously tested and evaluated.

Development of the Diagnostic

In developing and designing diagnostics, the "what the system does" is much more important than "how it does it." Therefore, designing a diagnostic for the computer section of ISCS requires a very clear idea of what functions it performs rather than how.

After a thorough study of the actual working of the ISCS system, a special diagnostic program (called *TPUT* (a mnemonic for Through Put) was developed. This program sends and receives messages to and from ISCS just as the customers do. By concentrating on the "simulation" or replacement of the outside environment of the computer section of ISCS, the *TPUT* program aids in the detection of errors that manifest themselves during the processing of the TPUT traffic.

The reasoning behind using a message simulation philosophy for TPUT follows from the inherent design of ISCS, which is a message-switching system. Even though TPUT does not point out the trouble areas, its use allows ISCS programmers to reproduce the problem repeatedly until the problem is isolated and solved.

TPUT

TPUT is a self-contained program located in the Phase I CC computer, in a memory area normally used for character buffering. TPUT requires that a Univac FH—30 magnetic drum unit be attached permanently to the CC. TPUT is a "hitch-hiker" type program designed to be invisible to the regular CC programs. However, since TPUT occupies memory locations and consumes processing time normally used by the CC program, TPUT limits the environment of the CC. At high traffic levels, this is important. To accomplish its objective of generating traffic for the Phase I computer center, TPUT must simulate messages coming to the computer. This simulation requires the TPUT Program to monitor all control and data transmissions between the CC software and the Input/Output (I/O) channels (which are directly connected to the MUX's). When TPUT detects activity that normally would go to a specific COM line (via an I/O channel to a MUX to a CLT), it compares the line number to a table of line numbers it wishes to replace (simulate). If a match is found, the information from the CC is made available to TPUT rather than allowed to be transmitted to the "COM" line. In a similar manner, TPUT gives information to the CC as an I/O channel (driven by a COM line) would. Therefore, TPUT has simulated the interface between the CC and the outside world for those lines chosen; in fact no actual MUX or "COM" line is needed.

TPUT Messages

The remaining task of TPUT is, then, to send and receive messages over the simulated COM lines. These TPUT generated messages take advantage of the unimportance of message contents in a test atmosphere; therefore, only routing information is really required. These routing lines are the only information that need be prepared to enable the operation of TPUT. A unique feature of TPUT is its ability to generate thousands of different messages, from a few choice groups of routing lines representing the geographical nodes of the ISCS. As TPUT receives back the messages it has sent, the time difference is noted. In this way TPUT fulfills its design of measuring the "through put" time of the ISCS system.

Applications for TPUT

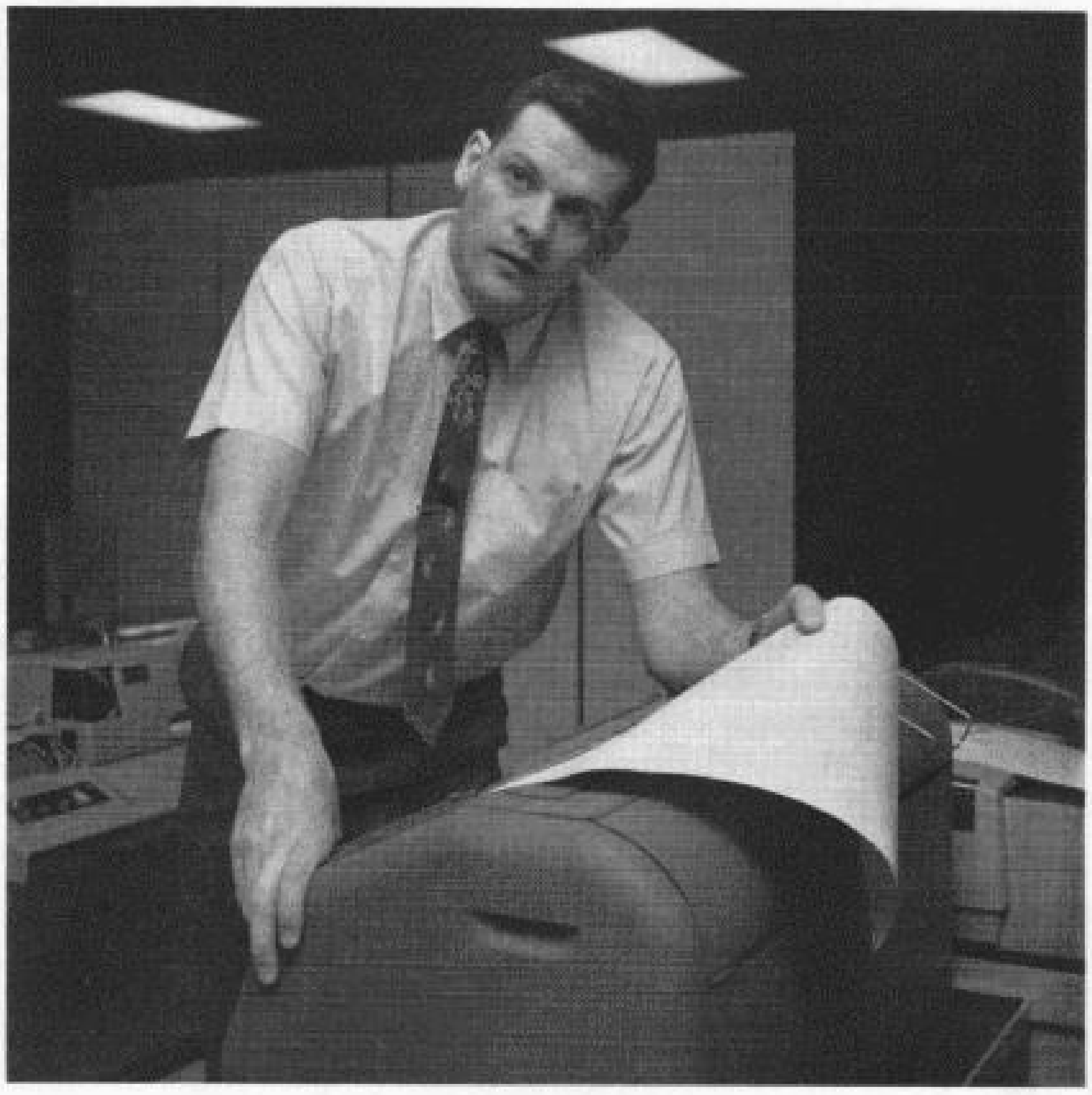
TPUT was designed to fill the gap between test message capability and test message desirability for handling large loads. It is designed to run all the terminals in a simulation mode and hence supply a background of traffic against which the specific test messages may be evaluated. Therefore, a test effort can be devoted to verifying a problem area, instead of trying to load the system in order to reproduce the problem.

TPUT has many other applications as a diagnostic. It can input a standard series of messages so that the ISCS computer system can be evaluated, by its performance in handling this standard load. TPUT can also drive the computer system to its traffic-handling limit, and hence determine for management evaluation how much customer service can be provided. Since the ISCS computer complex is made up of several separate but linked computer centers, with geographically dependent data relating to specific terminals, TPUT can run these computer centers at a test laboratory computer center prior to their installation or cut-over at the sites. This inherent simulation philosophy basic to TPUT allows any terminal type to be simulated. In addition, the traffic generated by the TPUT program can be varied a great deal for any particular test desired.

While TPUT is designed to be a one man operable system, it has the capability for automating many ISCS tests. Some of the possibilities for automation are as follows: automatic data base verification for specific computer center sites, automatic production of billing test cases for improvement of the billing process of ISCS, and the automation of test procedures previously used by duplicating their non-standard messages by means of simulation rather than actual transmission. This last possibility involves verifying the delivery of the rejection notices of these special cases.

Conclusion

TPUT has been a significant addition to Western Union's testing capability. TPUT allows us to test a shared system, such as ISCS, without interference to the customer and at the same time measure its capability in handling high traffic load.



R. D. O'Meara, Systems Analyst in the Planning and Engineering Operation, is responsible for the testing and design of ISCS, computer concepts.

He came to Western Union in September 1968. Previously, he was involved in simulating the Apollo flight to the moon in an automated flight simulator. He was also involved in application programming for Apollo.

Mr. O'Meara received his B.S. degree in Physics from the University of Illinois in 1965.

SYSTEMS DESIGN— A CONTINUAL ANALYSIS PROCESS

by B. Weitzer,
Director Systems Design & Analysis
Mr. Weitzer is responsible for the design of ISCS Projects

The articles in this issue emphasize the specialized and more sophisticated tools and techniques necessary to the design and implementation of the complex computer-controlled communication systems now being implemented by Western Union. New techniques have become necessary because of the difficulties faced by the designers of such systems. One such difficulty is the fact that the resources of such systems are fixed, while the demands or constraints made upon them are random. The designed system must meet various constraints such as: cost effectiveness, reliability, and service flexibility.

In order to meet this challenge the Systems Analyst devises analytical and simulation models, empirical measurement tools, and various effectiveness measurement techniques. Although such techniques are often critical elements of the design process they do not constitute the design process itself. Therefore the Systems Analyst must be careful not to lose sight of the primary design objectives in the design process lest he risk a project inadequacy induced by replacing primary goals by secondary models or techniques.

One major task of technical project management is the "continual evaluation" of the objective, tools, and techniques so that the design process moves forward toward its objective. Although easily stated, the task itself is not easily carried out, because the design process for such complex systems is not truly deterministic. The general approach consists of:

- *Defining the primary goals or attributes desired in the final product.*
- *Defining the constraints under which the product must be developed or operate.*
- *Decomposing the total design problem into sub-problems which hopefully can be optimized (locally) and definitely can be managed.*
- *Penetrating the design to a particular depth, re-composing the parts and extrapolating the results to the end product so that it can be measured against the design objectives and constraints. This, then, gives a "feasible" path through the design process.*
- *Adjusting the design elements to provide a better end result, continuing the design penetration and iterating.*

The major problems faced in this design process are those dealing with the re-composition ("synthesis") of the parts and extrapolation to the end product,—a delicate matter at best. The recomposition must adequately consider the interactions among the parts of the system and the resultant nonlinearities in the re-composition. It is usually in meeting these difficult issues that the most sophisticated of system models and measurement tools get defined and implemented in order to provide definitive check points on the design status. Often these tools and techniques are of sufficient generality and utility to provide guidance to future designers.

However, in the final analysis it is the engineering judgment—"talent"— of the managers and individual contributors which allows the design process to work. Without this judgment, the design approach generally fails. ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■

Supermarket Chain Uses Western Union Desk-Fax

The week-end prior to Thanksgiving Day, the big shopping week of the year, will see the inauguration of a test program conducted by Western Union and the A. & P. Food Stores at 11 supermarkets along the Eastern seaboard.

The test will determine the practicability of providing telegraph acceptance services in supermarkets. If successful, A. & P. envisions the addition of telegraph services to the growing list of services now being provided in their markets across the country.

Western Union's objective is to make their services more accessible to the public, especially in areas remote from telegraph offices.

Desk-Fax tie lines have been installed at the test stores. Desk-Fax units will provide an A & P agent with instant access to a Western Union office without the necessity for manual transmission of the message.

E. J. McQuillan, Director
Service Coverage
Public Office Operation