SIMULATION TO STUDY EFFECTS OF PRICING COMPUTER SERVICES

Mary Snuggs Loomis
University of Arizona

ABSTRACT

The scheme used by a computer center to price its services is generally an important factor in determining revenue, user mix, and resource utilization. Modeling and simulation of users' demands for computing services and their reactions to pricing changes can be applied to study the effects of various pricing schemes, thereby facilitating intelligent selection of a pricing policy for a particular center. In addition to developing a model of computer center activity, we are also concerned here with determining the characteristics which make a simulation language appropriate for expression and execution of the model. One essential aspect of the model is the adaptive behavior of computer center users. We have found currently available simulators (eg. GPSS, GERTS) to be inadequate to express the interactions of this model. We conclude that development of a simulation language which would allow valid expression of adaptive behavior would be a valuable contribution.

INTRODUCTION

A critical task of managing a computer center is effective cost allocation. Costs are generally allocated to computer users via a pricing scheme based upon a weighted consumption of available services. The scheme is an important factor in determining not only the center's revenue, but also the mix of users, resource utilization, and of course user costs. A number of studies of pricing computer services have appeared in the literature; major studies have been conducted by Sharpe [7], Nielsen [3], and Singer, et. al. [8].

Modeling and simulation of users' demands for computing services and their reactions to pricing changes can be applied effectively to study the effects of various pricing schemes, thereby facilitating selection of an appropriate pricing policy for a particular center. The simulation approach can also be used to provide information for planning purposes; for example to determine the effects of a particular cost allocation policy on expected revenue, resource utilization, and user mix with computer system upgrade and/or change in user population. In an environment with several classes of users and modes of processing, each with distinctive requirements of the system, the number of variables generally precludes tractability of comparing relative merits of several proposed pricing schemes analytically. Experimental comparison is also inappropriate in the real environment.

The intents of the work we report here were two: to develop a model of computer center activity as an aid to understanding the interaction of variables and to determine the characteristics which make a simulation language appropriate for expression and execution of that model.

In the following we first discuss the environment of computer center activity which we have modeled. We then present in some detail our experiences with implementation of the model in GPSS [1,2] and GERTS [5]. The reader is expected to have a working knowledge of GPSS; we will review the "less well publicized" concepts of GERTS. Finally we evaluate the simulators in the context of their effectiveness in our work.

COMPUTER CENTER ACTIVITY

ENVIRONMENT

The environment we model is a computer center which supports several modes of processing, eg. interactive, non-setup (i.e. no operator intervention) batch with limited resource requirements, and medium to large-scale batch processing. We will refer to these modes as INT, QIK, and BAT respectively in the following. The services are available to several classes of users, each with distinctive requirements and price-based propensity to use the system.

The independent variables of the model then are: SYSTEM CONFIGURATION of computing services and resources available; PROCESSING MODES and statistics of resource utilization by each; USER CLASSES and statistics of mix of processing modes requested by each; PRICING SCHEME to calculate revenue accrued from resource utilization; BEHAVIOR PATTERNS of each user class's propensity to use the system based upon price variation.

The example we use is an academic population comprising instructional users (students) who typically submit small, non-setup jobs with expectation of immediate turnaround, research users (sponsored and

non-sponsored by outside funding sources) who typically have compute-bound jobs, and administrative users who generally run I/O-bound jobs. Figure 1 exhibits example probability densities for job characteristics of (a) CPU time, (b) CPU region, and (c) tape use for the processing modes INT, QIK, and BAT. For application of the simulator, use of each type of resource incorporated in the pricing scheme would have to be characterized for each processing mode.

Figure 2 reflects several user classes' demands for each mode. These data were derived from the U.C.L.A. Campus Computing Network accounting reports of January and February, 1972. [9] (We recognize that although these date were adequate for our study, data would need to be gathered over a more significant time period in a real application of the simulation.)

User characteristics could be altered or refined to represent other environments, say a firm's computer center with interactive unpredictable management users who query a planning system data base, compute-bound scientific applications users, and I/O-bound business data processing users.

PRICING AND USER BEHAVIOR

Pricing schemes are generally based on weighted consumption of available resources, eg. CPU time, main memory utilization, secondary storage utilization, line printer and card reader utilization, operator intervention, terminal connect time, I/O control system calls. Such a pricing scheme affects user demands for resources. Like purchasers of other commodities, computer users generally want to maximimize service while minimizing costs. Because users can select data storage media, processing mode, and to some extent regulate the size and execution times of programs, they have a tendency to modify their requests to take advantage of a pricing scheme.

For example, if the rate for tape usage were to rise, the user might move her sequential files to disk media. Even though tape footage is cheaper to purchase than equivalent disk space, the center may have artificially reversed the costs to the user, say to discourage increasing the size of the already overflowing tape library, or to reduce operator intervention required to mount tapes. If main memory space were to become more expensive relative to CPU time in an unpaged machine, a user might compact her program's overlay-structuring or decrease buffer sizes. If interactive processing were to become unjustifiably expensive, a user might settle for the longer turnaround of batch processing. She might even take advantage of lower evening rates and overnight turnaround.

A pricing scheme should not only generate revenue adequate to cover operating expenses, but should also ration available resources. The rationale behind the charging algorithm then is not so much requiring payment for services rendered as for prevention of someone else's access to those services.

FIGURE 1

Processing Mode Characteristics

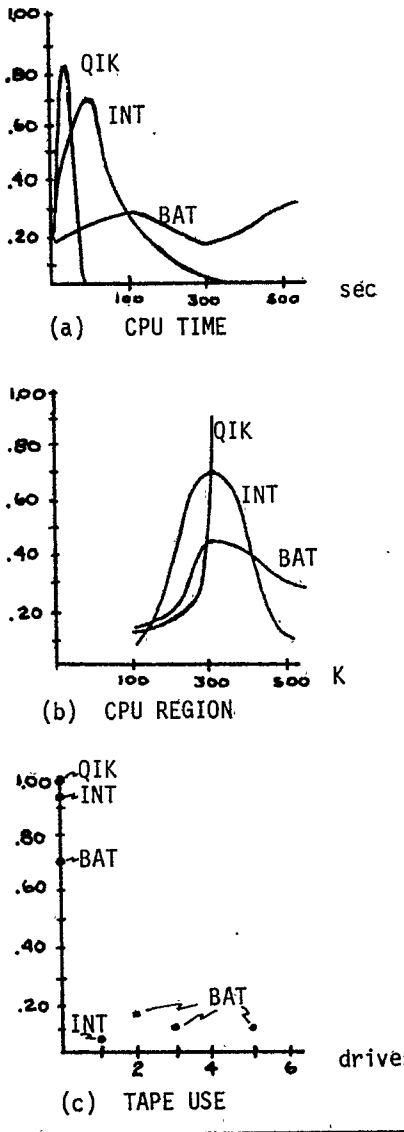


FIGURE 2

User Processing Mode Demands

User Class	Processing Mode - % INT	QIK	BAT
Student	0	100	0
Non-Spons Res	30	10	60
Spons Res	28	20	52
Admin	15	0	85

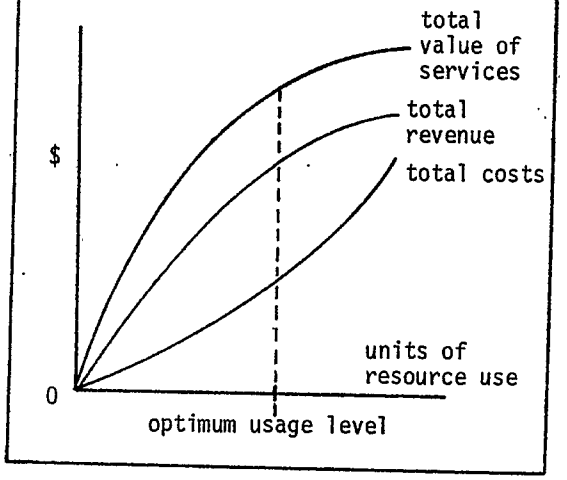
For planning purposes, the existence of high prices for a resource may be indicative of needed expansion. For example, if users willingly pay very high prices for disk space, leaving lower cost alternatives underutilized, then perhaps direct access capacity should be added to the system.

Different classes of users may also place different relative values on money spent for computing services. For example, in an academic environment students and faculty doing non-sponsored research are often funded by "soft money" which is somewhat easier to procure (and therefore less precious) than the real money brought in by sponsored research.

A user's behavior and the demands she makes on system resources are in part determined by the value she places on a particular service. Computer services are appropriately priced by value rather than actual costs (total, marginal, or average); actual unit costs may be difficult to allocate and to depreciate over time. Determination of value is rarely straightforward, but might include consideration of turnaround time, predictability of cost, and even time of day. (Submitting a job at 1 AM suggests higher value on rapid turnaround than does submitting at 10 AM) With "perfect price discrimination" each user would pay a price slightly less than the monetary value of the service to her. Theoretically, the optimum situation then occurs at a usage level which maximizes both value minus revenue (the user's concern) and revenue minus cost (the center's concern). [7] (See Figure 3)

FIGURE 3

Total value, total revenue, total cost, and maximum profit quantity [7, p.65]

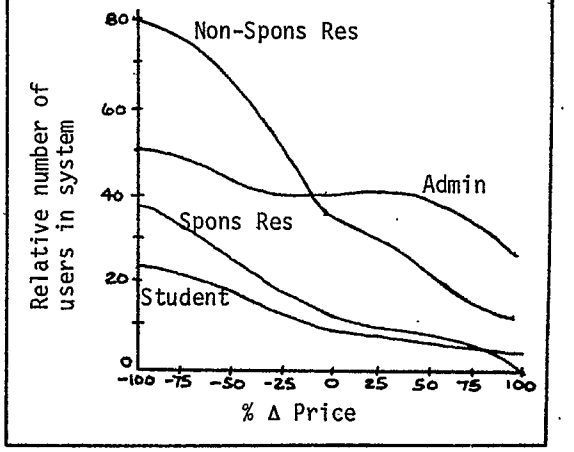


The interaction of pricing scheme and user behavior influences not only resource utilization and user costs, but also the overall user mix. Inflated prices might provoke the disgruntled user to leave the system for an external alternative, no matter how appropriate or convenient the center's configuration was for her needs. On the other hand, a relatively modest pricing policy might induce new users to the system. The freedom of a user to leave (or enter) the system is in part dependent upon the type of money she has. "Soft money" has no purchasing power other than for the center's computing; users with real money may be free to shop elsewhere. Students in general have no external alternative (except to drop class), but sponsored researchers and administrative processing users could very well leave the system if external rates proved attractive. A competitively low rate structure could introduce commercial users to the system. Decreased revenue does not necessarily follow from reduced rates.

Figure 4 gives an example set of behavior curves for student, sponsored- and non-sponsored research, and administrative users. The curves were intuitively derived, which was adequate for purposes of this work. Clearly to apply the model to a real situation would require further acquisition and validation of behavioral data.

FIGURE 4

User Behavior Curves



MODEL

We model the use of computer services as a time-advance discrete simulation. The system-state at any point in time is defined by the: USER MIX, relative numbers of each class of user in the system; USER BEHAVIOR, each class of user's current propensity to use the system, reflected on characteristic behavior curves; RESOURCE UTILIZATION by user jobs; REVENUE accumulated by charging for use of system resources.

Due to the elastic demand for services and adaptive nature of users, pricing schemes should affect steady-state conditions of revenue, user mix and behavior, and resource utilization. Pricing schemes can then be compared based upon their resultant steady-states. Other measures of performance, eg. user satisfaction, user "loyalty" to the computer center (perhaps regardless of cost considerations), internal cost/effectiveness, system reliability and security, have been omitted from the model in the interest of tractability.. However, each of these could be reflected in the steady-state conditions that are considered.

We judged optimum performance of the computer center to occur when the pricing scheme drawing the most revenue was applied. Note that optimum performance from the utility's point of view generally occurs under different conditions than those which optimize a particular user's concern for value of jobs run; the user is rarely concerned with the center's profit margin. The simulator could be modified to accommodate other performance measures.

GPSS MODEL

Figure 5 exhibits the basic flowchart of our GPSS model of computer center activity. The simulator was executed on an IBM 360 Model 91. GPSS function capabilities were used to define characteristics of user class, processing modes, and user behavioral patterns as follows.

User Class

Consider simulation of six user classes. Two functions were used to determine user mix. One user class function, eg.

```
16 FUNCTION RN6,D6
.10,1/.45,2/.60,3/.94,4/.98,5/.9999,6
```

was used to select arriving user class. The value of the independent variable generated by a system random number routine upon customer arrival determines to which of the user classes she would belong. A uniformly distributed independent variable would imply here that 10% of arriving users would be in class 1, 35% in class 2, 15% in class 3, etc.

The other user class function, eg.

```
17 FUNCTION *2,L6
1, 19/2,36/3,21/4,19/5,1/6,1
```

was used to represent the initial number of users in each class and was used in user behavior modification which we address later.

The mode of processing requested by an arriving user was determined by yet another set of functions, one for each user class. For example

```
12 FUNCTION RN7,D3
.15,1/.50,2/.9999,3
```

was used in a run to represent the processing mode mix of sponsored researchers. Here 15% of their work requested non-setup batch processing (mode=1), 35% requested interactive processing (mode=2) and the remaining 50% requested medium- to large-scale batch processing (mode=3).

Processing Mode

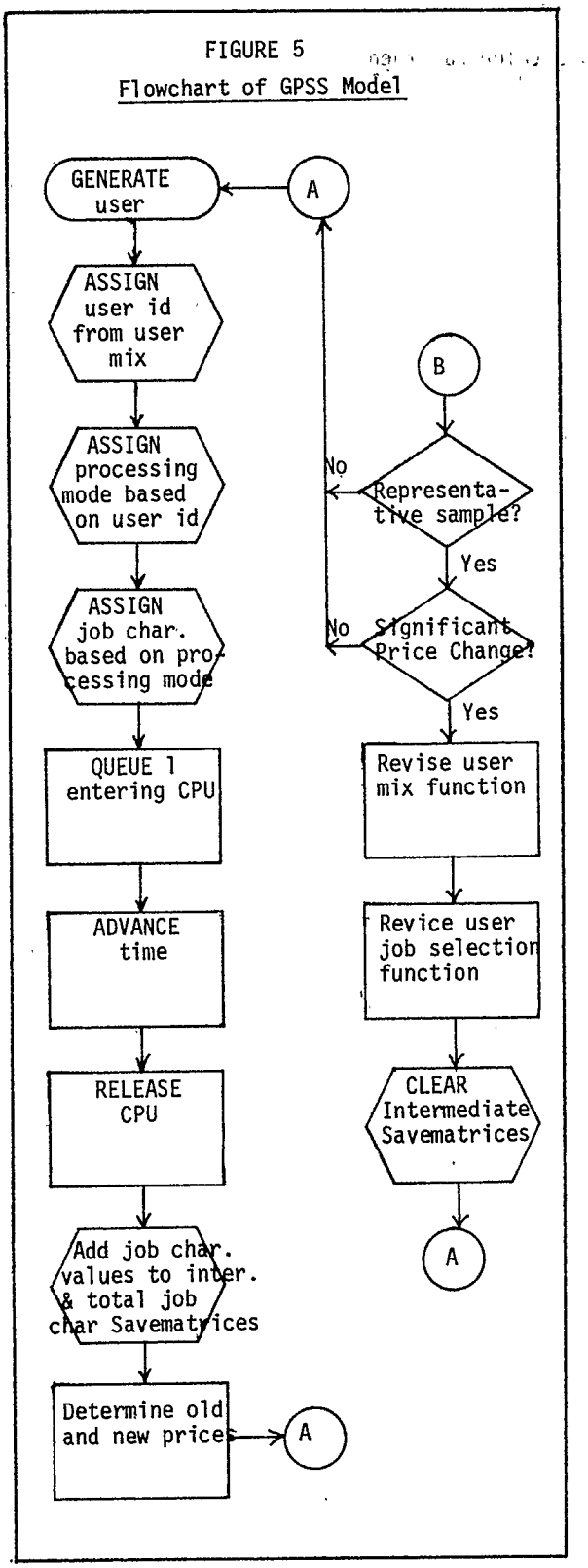
The resources requested by a user's job were determined by another set of functions, one for each processing mode. For example, processing mode function

```
7    FUNCTION  RN1,C3
.05,110000/.85,231000/.9999,151000
```

was used in one set of runs to represent small, non-setup batch processing. The dependent parameter

FIGURE 5

Flowchart of GPSS Model



here represented six aspects of processing characteristics; each digit was used to calculate a variable which corresponded to utilization of a system resource, as follows.

```
ASSIGN 3,FN*9     where P9=processing mode
                  (=7 here)
```

P9's value was then separated into digits and manipulated to give values to the resource variables:

V8=CPU time;
V7=main memory space;
V6=I/O requests;
V5=tape units;
V4=disk tracks;
V3=software premium.

User Behavior

Thus far we have seen that the simulator generated a user, determined her user class, then the mode of processing requested, then the resources required by that process. The user was then queued awaiting access to the CPU. Note that due to the focus and scope of this project, we viewed the computer as an old-fashioned uni-programmed "Black Box." The model could be applied to multiprogrammed systems by adding more processors to the model. A pricing scheme, for example,

```
Charge = CPU and I/O time * Fixed Region
         Factor   + disk-use charge
       = (20*V8 + .02*(V6+1))*1.333+40*V4
```

was then applied to calculate the charge to the user for resource utilization. The simulation continued in this fasnion, gathering statistics on revenue and user costs.

After steady-state was reached the pricing scheme was changed, say to

```
CPU time + I/O request factor + disk use
charge + tape-use charge
= 20*V8+(V6+1)+ (V4/10+V5)*60.
```

User behavior, specifically interarrival time, was modified based upon a set of user behavior curves. The following function corresponds to the student behavior curve in Figure 4:

```
1       FUNCTION    V19,C7
-100,38/-50,27/-25,23/0,19/25,15/50,13/100,10
```

where the independent parameter is percent change in price and the dependent parameter is relative number of student users in the system. That is,

```
GENERATE K3,V2
```

where P2=user class identifier,
and 1 VARIABLE FN*2 is a scaled value determined by user behavior position (the function above)
and 2 VARIABLE FN8*V1, where FN8 is the desired function to add variation about the mean to the interarrival rate of this class of users.

The simulator would then continue, modifying each user class's distribution of arrival time to the system.

Although we could modify a user class's propensity to use the system, we could not find a clean way to modify her demands for a particular system resource. The chain of functions which we found necessary to generate characteristics of a particular arriving user's requests quickly requires a combinatorial number of options.

GERTS MODEL

In this section we first draw from a paper by Pritsker and Burgess [5] to introduce some basic concepts of the GERT Simulation program, then describe our GERTS computer center activity model.

GERT Simulator Program

GERTS III is a general purpose program written in FORTRAN IV for event-driven simulation of networks. Network branches represent activities; network nodes connect activities and represent input and output of activities. There are three types of events associated with simulation of a GERT network: start of the simulation, end of an activity, and completion of a simulation run.

"The start event causes all source nodes to be realized and schedules all activities emanating from the source nodes according to the output type of the source node. The output type for all nodes is either deterministic (all activities emanating from the node are scheduled) or probabilistic (one activity emanating from the node is scheduled)... For each activity scheduled, an end of activity event is put in a file containing all events in chronological order...The simulation proceeds from event to event..." [5, p. 3]

Figure 6 depicts two types of GERT nodes: a circular node is used to represent deterministic output; a teardrop node is used to represent probabilistic output. In the figure,

n = node identifier number
and i = number of releases, i.e. times activities incident to the node must be realized before the node is realized.

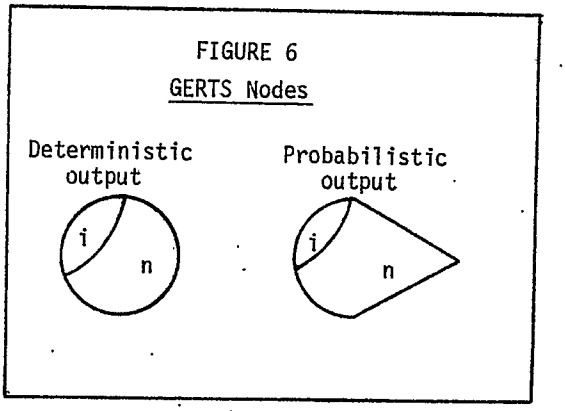


FIGURE 6
GERTS Nodes

A GERT network branch represents an activity (which may be an information transfer). Associated with each branch is a descriptor (p; t_p; t_d; c, a) where

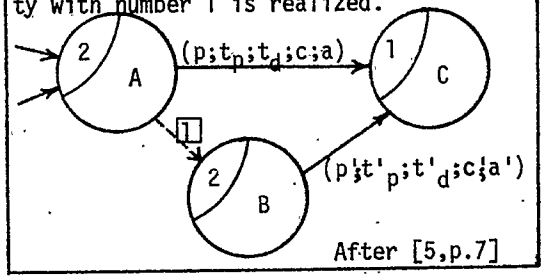
p = probability of realization
t_p = parameter set for time of activity
t_d = distribution type for time of activity
c = counter type
a = activity number

"Activity numbers are given to branches to permit network modifications based upon realization of the activity," [5, p.7] illustrated in Figure 7.

FIGURE 7

GERTS Network Modification

node A is replaced by node B when activity with number 1 is realized.



After [5,p.7]

Let a = number of incident activity branches. If i = 1 and a > 1, then the input side of the node can be interpreted as an OR operation. If i > 1 and i = a, then the node is an AND operator. If i < a, then only some of the input activities need be realized. If i > a, then some of the input activities must be released multiple times before node n can be realized.

Each time an end of activity event occurs, the number of releases for the end node of that activity is decreased by one. When the number of releases remaining is zero, the node is realized, "...activities emanating from that node are scheduled and the simulation is continued. The simulation ends when a prescribed number of sink nodes have been realized." [5, p.4]

GERTS Computer Center Activity Model

The GERT network developed to model computer center activity is depicted in Figure 8. The environment was restricted to two modes of processing, (INT and BAT) and two classes of users (STUDENT and RESEARCH).

For each user generated by the simulator, the following must be determined: user class, job processing mode, resource utilization by that job.

The user mix is represented by the probabilities of realizing the branches out of node 4. The branch from node 4 to node 5 is realized when a STUDENT user is generated; the branch from node 4 to node 6 is realized when a RESEARCH user is generated. Likewise, the probabilities of realizing the branches out of nodes 5 and 6 represent the processing mode mix within user class. Activity 1 occurs when a STUDENT BAT job is generated; activity 2 occurs when a RESEARCH BAT job is generated; activity 3 occurs when a RESEARCH INT job is generated. These activities control flow through the network such that the job characteristics for the appropriate user class and processing mode are considered during simulation.

The network of Figure 8 allows investigation of a pricing scheme with five factors: CPU time, CPU region, I/O request time, tape use, and a constant term. Depending upon the particular type of job being processed, i.e. whether activity 1, 2, or 3 was realized, an appropriate path is selected through the resource utilization nodes of the network.

Because arithmetic calculations cannot be explicitly specified using GERTS, it was necessary to indirectly assess charges for a job's utilization of resources. We used the GERTS IIIC facility which allows a cost per unit time to be associated with a branch in the network. We let a branch represent use of a particular resource. We set the cost of that branch equal to the coefficient of the corresponding resource in the pricing scheme; we specified the time to traverse the branch parametrically using the characteristic distribution of utilization of that resource for the particular type of job (recall Figure 1). We could then interpret the cost to realize the end node of the branch as the amount to be charged for use of that resource. For example, in the following pricing scheme:

Charge = 1.33*(CPU time) + 0.33*(I/O request time) + ...

the coefficient of I/O request time is 0.33. The branch representing I/O request time for, say, a RESEARCH BAT job connects nodes 29 and 30. (See Figure 8). Thus the cost per unit time associated with that branch would be 0.03; the parameter associated with branch would be the distribution of I/O requests by RESEARCH BAT jobs.

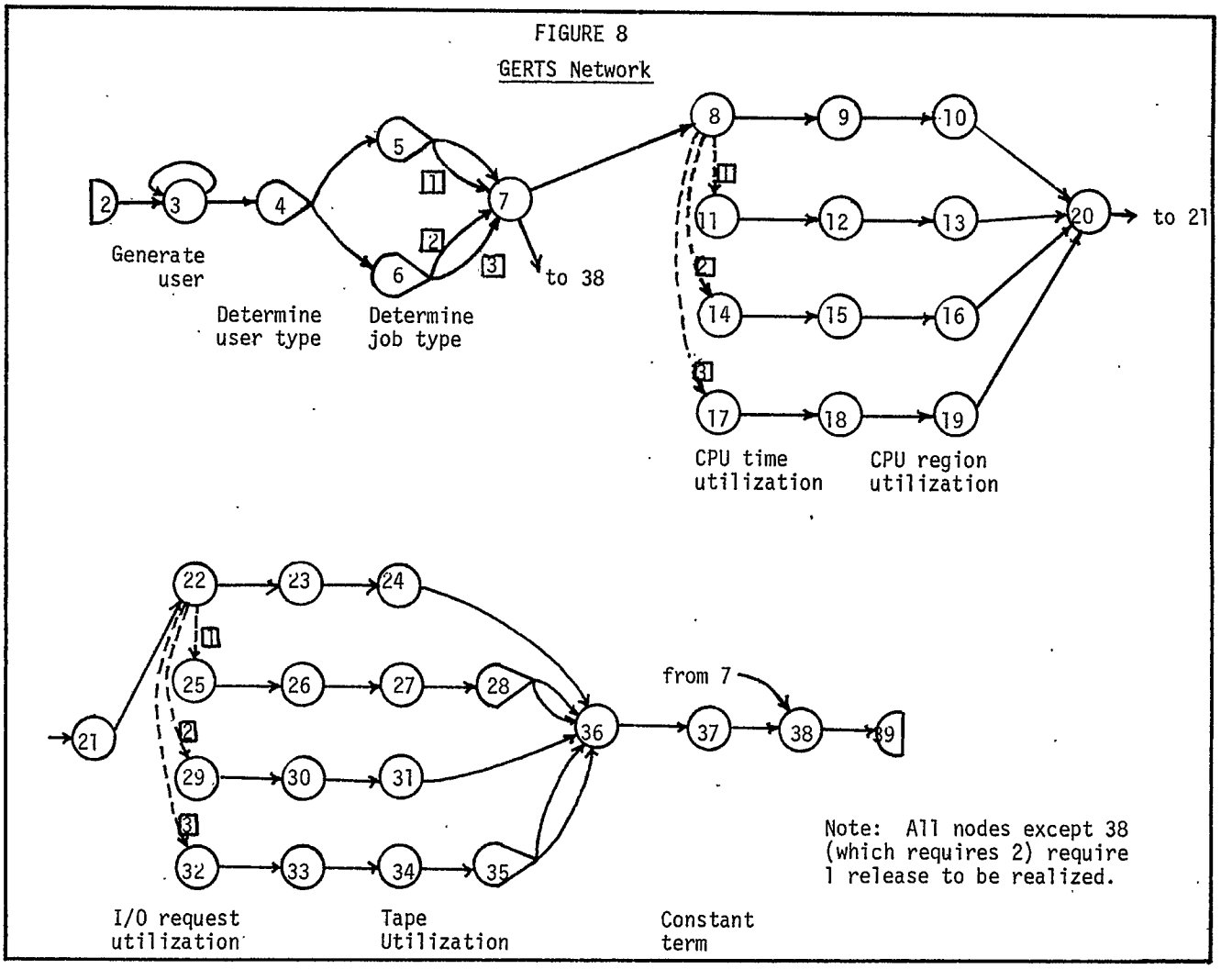
The main limitation posed by serially charging for resources is that a pricing scheme with multiplied factors, eg. elapsed CPU time * CPU region factor, cannot be modeled. This is a serious drawback. Additionally, in order to introduce a constant term (eg. software premium) into the pricing scheme it is necessary to insert an essentially null node into the network (node 37 in Figure 8), with parametric time 1 and the associated cost rate equal to the desired constant.

We found that GERTS is not suitable for modeling user behavior; there is no acceptable provision in GERTS to dynamically modify distributions (i.e., parameters) or probabilities of branch realizations. Modification of the network through activity occurrances is inadequate for this. This inadequacy could be countered by manually simulating changes in user behavior, then altering parameters and distributions between runs. If one considers behavior modification to be a slow, quantized process, then perhaps this solution is reasonable. In effect each simulation run would then be a time-slice of the overall picture. A more satisfactory solution would be to use a simulator which does allow for dynamic modifications of distributions.

The model is also somewhat invalidated since queues can not actually form within the system. Use of GERTS IIIC (which allows costs for activities) is exclusive from use of GERTS IIIQ (a GERTS variation allowing queue formation). GERTS would be much more generally applicable were cost, queue, and resource allocation facilities combined into one program.

FIGURE 8

GERTS Network



CONCLUSION

We have applied both GPSS and GERTS III to our model. The major value of these exercises to us has been in gaining understanding of the problems of simulating adaptive user behavior and computer center activity. We have found neither simulator entirely appropriate for expression of the model. Most significantly, neither provides for the dynamic modification of probability distributions which is necessary in modeling changes in job interarrival time distributions, users' distribution of requests for particular system resources, and probability distributions of processing nodes and user classes in the system.

We conclude that simulation of computer center activity is valuable; a notable application is its use in selection of a pricing scheme. One essential aspect of the model is the adaptive behavior of computer center users. However, we found GPSS and GERTS inadequate to express the sociological interactions of this model. A simulation language which would allow valid expression of adaptive behavior would be a valuable contribution.

Acknowledgements

The guidance of U.C.L.A. Professors R.T. Nelson of the Graduate School of Management and L.P. McNamee of the Computer Science Department during the course of this work was appreciated.

REFERENCES

[1] IBM General Purpose Simulation System/360, Introductory User's Manual, GH20-0340-4, IBM, 1969.

[2] IBM General Purpose Simulation System/360, User's Manual, GH20-0326-4, IBM, 1970.

[3] Nielsen, N.R. "Flexible Pricing: An Approach to the Allocation of Computer Resources," AFIPS Conf. Proc., 1968 FJCC, Vol. 33.

[4] Nunamaker, J.F. and A. Whinston, "Computer System Management: A Macro Planning Cost Allocation Procedure," Management Informatics, Vol. 2, No. 4, 1973.

[5] Pritsker, A.A.B. and R.R. Burgess, "The GERT Simulation Programs: GERTS III, GERTS IIIQ, GERTS IIIC, and GERTS IIIR," Department of Industrial Engineering, Virginia Polytechnic Institute, 1971.

[6] Pritsker, A.A.B. and P.J. Kiviat, Simulation with GASP II; A FORTRAN Based Simulation Language. Prentice-Hall, 1969.

[7] Sharpe, W.F. The Economics of Computers. Columbia University Press, 1969.

[8] Singer, N., H. Kantor, and A. Moore, "Prices and Allocation of Computer Time," AFIPS Conf. Proc., 1968 FJCC, Vol. 33.

[9] Campus Computing Network, University of California, Los Angeles, CA, 90024.