# A STUDY OF PRICING POLICY FOR A SELECTED INTERNAL COMPUTING ENVIRONMENT

---

A Thesis

Presented to

the Faculty of the Graduate School

The University of Houston

---

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Industrial Engineering

---

by

William Francis Stanton

August 1971

## ACKNOWLEDGMENTS

I wish to express my gratitude to the following parties for their support and assistance in the preparation and completion of this thesis:

To Dr. Willard N. Ander, my thesis advisor, for his encouragement, patience, and guidance. Without his support, I would have been unable to realize and develop the subjective portion of my research.

To Dr. Charles E. Donaghey and Mr. Joseph B. Wyatt, members of my committee, for their advisement in the finalizing of the manuscript.

To Dr. Scott T. Poage, my department chairman, for his personal aid in navigating the administrative labyrinth.

To Dr. Richard A. Kuebbing, fellow employee, for his helpful suggestions as to thesis content.

To Getty Oil Company, my employer, for appreciating my situation during the final stages of preparing the manuscript.

To Mrs. Lillian Yancey, my thesis typist, for an excellent job in the final typing of the manuscript.

To my wife, Aniela, and my two children, Valerie and Rachel, for their continued support during my graduate career.

A STUDY OF PRICING POLICY FOR A SELECTED
INTERNAL COMPUTING ENVIRONMENT

---

An Abstract of a Thesis Presented to
the Faculty of the Graduate School
The University of Houston

---

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Industrial Engineering

---

by

William Francis Stanton

August 1971

TABLE OF CONTENTS

Chapter | Page

## LIST OF FIGURES

# Chapter 1

## INTRODUCTION

The investigation represented by this thesis was directed to certain aspects of resource allocation which relate to the use of centrally-supplied computing resources by internal organizational components of private business enterprises.

This introductory chapter is concerned with the staging of the research that was performed. Prior to a statement of the problem, a background section is included which relates evolution of computing usage and associated resource allocation mechanics to some common notions of private enterprise internal computing which were investigated during this research. To establish a clearer feeling for the problem area, in that this research was restricted to an assumed private enterprise computing environment, broad characteristics of the assumed environment are presented during the background discussion. Furthermore, in an effort to establish validity of the selected problem area, some literature-supported examples of ineffective resource allocation are cited.

Following the statement of the problem is a discussion of the theoretical framework within which the writer chose to relate to the research problem.

Items discussed in the theoretical framework section are set forth in a descriptive and definitive fashion and pertain to such facets of the assumed computing environment as physical resource characteristics, job mix characteristics, timing considerations, capacity determination and the physical manner of processing user work.

## BACKGROUND

It is within the background discussion that the writer has chosen to define and establish validity for the selected problem area. The items which are presented in this background section include evolution of computing usage, delimitation of internal computing usage, hierarchy of resource allocation levels, funding of the computing facility, facility management considerations, broad characteristics of the assumed computing environment, examples of ineffective resource allocation, the necessity of management's awareness for computing facility performance, and the necessity for pricing policy.

### Evolution of Computing Usage

Electronic data processing has been actively pursued for over twenty years because of the utility of computational speed to society. The class of electronic digital computers developed in the period beginning in the late 1940's offered a tremendous increase in the rate of performing logical and arithmetical operations. Engineers, mathematicians, and computer scientists soon realized that the digital computer could be used to perform computations never before attempted because of seemingly endless manual calculations.

Coupled with the computational speed offered by the digital computer was the capability to store large quantities of information which could be automatically and quickly accessed. Private business management recognized that the computer would be the key to the maintenance and accession of large volumes of information. Office personnel who once had to manually maintain large filing systems would now spend more time merely coding transactions for automatic file updating by the computer.

### Private Enterprise Internal Computing Usage

Computer usage by private business is widespread. In most instances, a firm will purchase or lease computing equipment from a computer manufacturer or supplier. Once so obtained, the computing capability of the equipment may be developed and either sold for profit to those who cannot justify such equipment acquisition or used solely by internal organizational components.

This investigation was concerned with the type of internal computing usage found in private business firms. Other types of internal computing usage do exist however; examples of such other types of internal computing usage are found in governmental and educational institutions. Since governmental and educational computing costs are subject to certain restrictions as indicated by Kanter (1968), these internal usage environments were not considered.

Levels of resource allocation. To convey an appreciation for the scope and nature of the short-range resource allocation mechanics chosen for this investigation, an introduction is given to the rather broad set of resource allocating mechanisms prevailing in private business.

At the uppermost level, the firm's top management allocates funds periodically to best satisfy company goals or objectives. Wright (1967) says that at this level the allocation of funds is strategic in nature and not merely motivated by profit. Wright does point out however, that within the framework of the primary or strategic allocation, profitability does form one of the principal objectives of the financial function of management.

Profitability considerations dictate selecting the more profitable projects for investment of capital. Determination as to the relative strategic importance of computer resources in terms of cost-effectiveness and profitability are usually vague and subjective. Management, therefore, has had to believe in the long-term value of automation; for example, at the incept of computerized automation within the firm, it was easy to see that the computer was capable of accurately and quickly performing routine clerical operations.

Just below the more strategic fund allocation is the allocation associated with project selection within either the profit or the overhead centers. Concerning a profit or an overhead center, each has an operating budget with which to meet its objectives. It is this level of allocation that is seen to establish a relative value of each active project to the firm as a whole.

At the level of resource allocation toward which this research was directed, there exists a using community of individuals, projects or similar entities desiring, from time to time, to take advantage of available computing resources. Characteristic of the allocation problem at this level is the requirement to distribute the scarce computing resources to user demands as a function of time; user demand is evidenced collectively by the work load of pre-structured computing tasks or jobs submitted by the user community for processing by the computer system during some finite period of time. Bases for allocation of the computing resources to the user demand or job mix depend, in general, on many diverse criteria such as implicit or explicit priorities, administrative processing rules, a general tendency toward a first-come-first-served (FCFS) processing policy, minimization of turnaround time, and maximizing certain performance measures.

Funding of computing resources. Two methods of funding a supply of computing resources are isolated for consideration in this thesis; these methods were assumed to have been common cost recovery schemes for the type of computing environment selected for this research.

One of the cost recovery schemes assumed to have been in common usage is indirect or overhead funding. Overhead funding is simple to implement and could be a reflection of how computer processing philosophy had evolved during the 1950's and 1960's; the growth of private business internal computing quite reasonably began with routine clerical and accounting applications which traditionally were funded as overhead. Very briefly, implementation of the indirect funding method is achieved by distributing computing costs and other overhead items back to the operating components within the firm on a basis not necessarily associated with the usage of computing resources by these operating components.

Contrasted to the indirect or overhead method of funding computing operations is the direct or average cost method. Unlike the indirect method, the direct approach is based upon some fixed period during which the actual costs of computing will be passed on to only the actual users in proportion to the relative amounts of computing resources used.

It is the viewpoint of the writer that, during the period from 1950 to 1970, along with the growth of private enterprise internal computing, a trend was underway representing a transition from high initial overhead cost or burden to more justification from direct usage generated by line operating components or profit centers; in Chapter 3, the writer expands on his viewpoint.

<u>Computing facility management considerations</u>. With the private enterprise internal computing situation introduced, a few points of concern to computing facility management are listed next in question form:

1. What are the needs of the user community in terms of computing resources?

2. Is the present supply of computing capacity economically appropriate for present and future needs?

3. Is the present supply of computing capacity supporting peak loads of usage?

4. Are the existing computing applications well designed?

5. Is the method of charging users for access and use of the computing facility equitable and conducive to effective resource allocation?

6. How much lost production time is caused by equipment failures, environmental effects, operator errors, system software problems, and user programming errors?

7. How effective are programming personnel during program development?

8. How is the available capacity of computing resources being used?

Of the questions posed in this list, the writer's research effort was directed to questions three, five and eight.

Characteristics of the Assumed Computing Environment

The particular computing environment toward which the research effort was directed is delimited by the following characteristics:

1. A digital computing system with one central processing unit or CPU.

2. A work load or job mix consisting of approximately equal amounts of overhead processing and processing requested or sought by line operating components or profit centers.

3. An online input and output capability consisting of card reading, card punching, printing, disk reading and writing, tape reading and writing and console typewriting.

4. An off-line capability consisting of plotting from magnetic tape input.

5. A window-batch type of job submission arrangement as opposed to remote-batch or conversational job entry.

6. A multiprogramming capability consisting of being able to process one user job concurrently with system input and output.

7. A funding scheme for cost recovery which is the indirect or the direct average cost method.

8. With direct funding, a measure of one resource consumption variable is the basis for charging users in relation to their use of the computing facility and this measure is the time lapse externally during which a user job

is considered to have sole access to the computing facility.

9. If direct funding is applicable, the users would be subject to the operating budget constraint imposed on their respective profit and overhead centers; however, if overhead funding is applicable, the users would be seen as requesters of service from an electronic data processing (EDP) organization for which the receipt of such service would not impact their operating budgets.

Literature references to ineffective resource allocation. This section purports to establish validity to the isolated problem area by presenting recent (i.e., since 1967) viewpoints of other researchers expressing concern over the ineffective resource allocation resulting from specific practices associated with the use of computing resources.

Either of two funding approaches (i.e., the indirect method or the direct method) is assumed to prevail as the cost recovery scheme in the selected computing environment; these funding approaches are critiqued in the quotations that follow:

1. Nielsen viewpoint on indirect funding. The following direct quotation is taken from Nielsen (1968):

> . . . , under indirect costing the expenses connected with the acquisition and operation of a computer are charged against various overhead accounts, so that the user does not "pay" in proportion to his usage. Thus the computer is looked upon as something of a "free good," and the demand for service readily exceeds the supply. Since the computer can provide only a certain maximum quantity of service in a unit of time, a form of de facto allocation must take place.

For example, long turnaround times may serve to discourage a sufficient amount of usage to bring supply and demand into balance. Certain administrative rules may also be used to enable the center to provide a more suitable service for the majority of its users (e.g., only jobs of 10 minutes or less can be run during the prime shift).

Further, in any environment there are always certain users who request special treatment, such as exemption from the standard scheduling procedures. Often a center's management is forced to pass judgment upon the importance or worth of these projects in deciding whether or not to grant such requests. (Alternatively, personal friendships with computer operators or other such considerations may decide these questions.)

In summary indirect costing results in perpetual saturation, so that management has no guide as to when additional capacity should be installed. Furthermore, allocation of the available computing resources must be made on somewhat arbitrary grounds rather than for maximum user benefit.

2. <u>Singer, Kanter, and Moore viewpoint on indirect funding</u>. The following direct quotation is taken from Singer (1968):

> When a facility is used widely and the cost of its services is difficult to impute to individual users, the facility is frequently called an "overhead expense" and its cost is then allocated to users on an arbitrary basis. When computers are treated as overhead, the full costs (including amortization) of the computer center are included in the firm's general overhead pool which is imputed to individual projects on a basis such as total labor costs, total man-hours, or total operating costs of each project . . . It should be obvious, however, that overhead charges can offer the proper incentives to neither the user of the computer nor the administration concerned with supplying computer time. Each user will prefer to substitute computer time for other resources, thus reducing his basis for overhead charges, and the overall effect must be to increase the demand for computer time. (In addition, overhead charges will discriminate against projects which are not computer-intensive, thus creating an inequitable set of charges.) If each user substitutes the same ratio of computer time for direct charges, the result will be to leave the pattern of charges unchanged, but to bias upward total use of the computer. The supplier of

> computer time will then be misled into overestimating the demand for the computer, resulting in overinvestment in subsequent computer facilities. Finally, the overhead rates that the firm must charge . . . will be inflated by the misallocation of resources and overinvestment, leading eventually to declining revenues and to reduced profits in the case of a firm.

3. Smidt viewpoint on direct funding. The following direct quotation is taken from Smidt (1968a):

> If charges for a computer are determined by allocating its total cost over the total usage during a given time interval (usually a year) the charges provide incentives that are exactly the opposite of what is desirable. When the computer is new the fixed costs are allocated over a small volume of work leading to a high cost per unit of work. When the computer is old (and operating near capacity) approximately the same fixed costs are spread over a much larger volume of work leading to a low cost per unit of work. Insofar as users respond to the costs charged, they tend to economize on the use of the computer in the early days when excess capacity is available and to be liberal in their use of it late in its life when capacity is being pressed.

4. Nielsen viewpoint on direct funding. The following direct quotation is taken from Nielsen (1968):

> On the other hand, under direct costing the user pays the cost of his usage; the more he runs, the greater his charge. Thus management has some gauge as to what the real demands on the center are. However, the allocation problem still remains. Since the cost to provide a given amount of computing is nearly independent of the turnaround or service given to any particular job, resort must still be made to administrative procedures for determining the service which a job is to receive.
>
> A further effect of direct costing stems from the fact that the provision of computer services can be characterized in the short run as a high fixed cost, low marginal cost operation. Thus, the cost per unit of computing is greatly influenced by the total usage of the facility. When demand is heavy, the cost is low, thereby encouraging even further usage and worsening the turnaround situation. When demand is light, the cost is high, discouraging use of an already lightly

loaded facility. In situations where users are required to use the cheapest facilities available, work is driven off lightly loaded systems to more heavily loaded ones; just the opposite of what would be desired for the most efficient utilization of the available resources.

5. Singer, Kanter, and Moore viewpoint on direct funding. The following material is taken in summary form from Singer (1968):

1. Computer centers are unable to provide service to additional users at marginal (social) cost.

2. The resulting pattern of charges encourages use during peak periods and discourages use when the computer is idle.

3. Overinvestment is likely to result if suppliers are guaranteed recovery of average cost.

4. Direct funding or average cost charging is a popular internal accounting device due to its ease of administration but the incentives it offers are unlikely to promote efficient resource allocation.

Another instance of ineffective resource allocation is the diseconomy associated with deferred service. It is quite common for a job to be readied for processing at a particular point in time and then, according to some arbitrary and inconsistent administrative ruling, be constrained to undergo an inequitable waiting period without proper economic compensation. Singer (1968) cites this situation as an imbalance in social and private value. Facility management is imposing external costs on the user community (i.e., other than to the immediate user and the supplier) by arbitrarily denying access to other users.

With respect to administrative rulings issued by the management of the computing facility, there is a built-in misallocation associated with criteria used in such instances as bases for short range resource allocation decisions. Nielsen (1968) emphasizes that administrative rulings such as "No jobs longer than 10 minutes during the prime shift because we want to provide good turnaround" and "Process Joe's jobs as soon as possible after they are submitted, since he has a crucial project and a tight deadline" arise from the "inability of installations to live with a FCFS procedure." The usual default choice of FCFS is judged as inappropriate by those who may have tried to avoid invoking an effective allocating mechanism. Nielsen (1968) summarizes on this point by stating, "The trouble is that the administrative regulations . . . designed to temper the effects of the FCFS procedure are often made or determined in practice by the individuals least qualified to make those types of decisions."

Necessity of managerial awareness of computing facility performance. Schroeder (1971) reports on a recent study conducted by A. T. Kearney & Company to consider the effectiveness with which EDP managers manage the EDP function. This study included a stratified random sample of 89 companies with 155 computers. The firms included in the study were of varying size and represented several major industries. The size of computing facilities considered, in terms of machine rental per month, ranged from under $10,000 to over $40,000; the computers in the survey represented a combined capital investment of approximately $110 million purchase price.

Considered in the Kearney study were several aspects of performance such as rerun analysis and scheduling, however, these topics were considered to be outside the scope of this thesis research. Pertinent results to the thesis research from the Kearney study are tabulated as follows:

1. The computers were operated only 64 percent of available time. Operation, here, refers to the state of being manned by operating personnel but not necessarily performing work.

2. The computers were idle 16 percent of the operating time. Cost associated with this idle time was estimated to be $394,000 annually.

Schroeder concludes that "The lost utilization indicates that management is not giving its attention to the performance of this expensive equipment. This is substantiated by the fact that 42 percent of the respondents report the data submitted is estimated because accurate records of computer utilization are not maintained."

Related to Schroeder's conclusion, the writer's research effort was directed in part to simple and practical methodology with which to meaningfully represent for management the allocation of available computing capacity to user work over absolute time.

Necessity for pricing policy. Nielsen (1968), Smidt (1968a), and Singer (1968) share a common belief that reasonable and equitable approaches to the allocation of scarce computing resources to the user community can be brought about through some form of pricing. Pricing, here, is used in the sense as conveyed by Nielsen or "rates which are set so as to govern usage rather than to reflect strictly the cost of providing the service."

The main thrust of the thesis research effort was directed to pricing considerations and methodology as related to the assumed computing environment.

## PROBLEM

With the characteristics and validity of the selected problem area introduced, a statement of the research problem is given as follows:

Pertaining to the assumed computing environment:

1. What considerations could be cited from previous research relating to the tailoring of a pricing policy to serve as an economically reasonable allocating mechanism for the short range distribution of computing resources to the user community?

2. What pricing policy methodology was either suggested by previous research or could be formulated by the writer for future experimentation and study?

3. Develop and experiment with a computer program to graphically depict the gross allocation versus time of system computing resources to user jobs.

## DELIMITATIONS

Basic to the character of this investigation was the premise that the research effort would not be directed toward the solution of a specific problem but rather toward the analysis of the problem area in the sense of providing direction to future experimentation and study.

During this investigation, the sources of information were the literature, discussion and interviews with the thesis advisor and members of the thesis committee, and viewpoints of the writer based upon professional experience.

The time period during which this research was conducted began in February 1970 and continued through June 1971.

## THEORETICAL FRAMEWORK

Within this section, the writer has related his viewpoint as to how certain technical details of the problem area will be considered in any later discussion offered directly by the writer.

### Physical Computing Architecture

To discuss the allocation of digital computing resources, some definitions need to be established about the physical computing architecture in question; such computing architecture has the schematic form indicated in Figure 1. Defined as follows are the hardware components shown in Figure 1:

1. Memory Module - The memory module is random access storage for instructions and data. Storage is divided into sequentially addressable cells of fixed size. This storage facility is referenced by the central processing unit and the data channels. A memory reference consists of either transmitting a cell of information to the memory or retrieving a cell of information from the memory module. The time period during which this reference takes place is called a memory cycle. During a memory cycle, only one data channel or the central processing unit may be performing a memory module reference.

2. Central Processing (CPU) - The CPU consists of digital logic capable of addressing the memory module for retrieval or replacement of cells; the addressed cells may be data to be examined or modified or may be instructions for the CPU to perform. The CPU controls the operation of the overall computing system in processing the instructions it retrieves from the memory module.

3. Data Channels - The data channels also consist of digital logic capable of accessing the memory module for storage and retrieval of cells. A data channel is also a logical path between an input-output control unit and the memory module. The cells of information flow may be commands for the channel itself to perform, commands for the input-output control unit to perform or data transmitted to or from some input-output unit.

4. Input-Output Control Units - Each such control

unit has unique digital logic supporting the specific type of input-output unit it controls; this unique logic is necessary in order to provide the standard logical interface to a data channel for a particular input-output unit. Functionally, a control unit initiates and monitors operation of its input-output units and controls the flow of any data across the interface with its assigned data channel.

5. Input-Output Units - These units are used for either one-way transfers of data such as with card readers, card punches, and printers or two-way transfers such as with magnetic tape units and magnetic disk drives. Console typewriters are a special case of one-way data transfer. Characteristic of one-way data transfer units is their inability to store information for automatic return to the memory module at a later time; two-way data transfer units, however, have such a capability.

## CPU and Data Channel Timing Considerations

Even though the CPU is in control of the overall computing system, it must contend with the data channels for available memory cycles. Because data movement input-output operations quite often require continuous, uninterrupted service by data channels, CPU operation is occasionally delayed. The delay of CPU operation caused by channel interference can occur when one or more data channels must be granted memory access for information exchange during such memory cycles when the CPU is also requiring memory access;

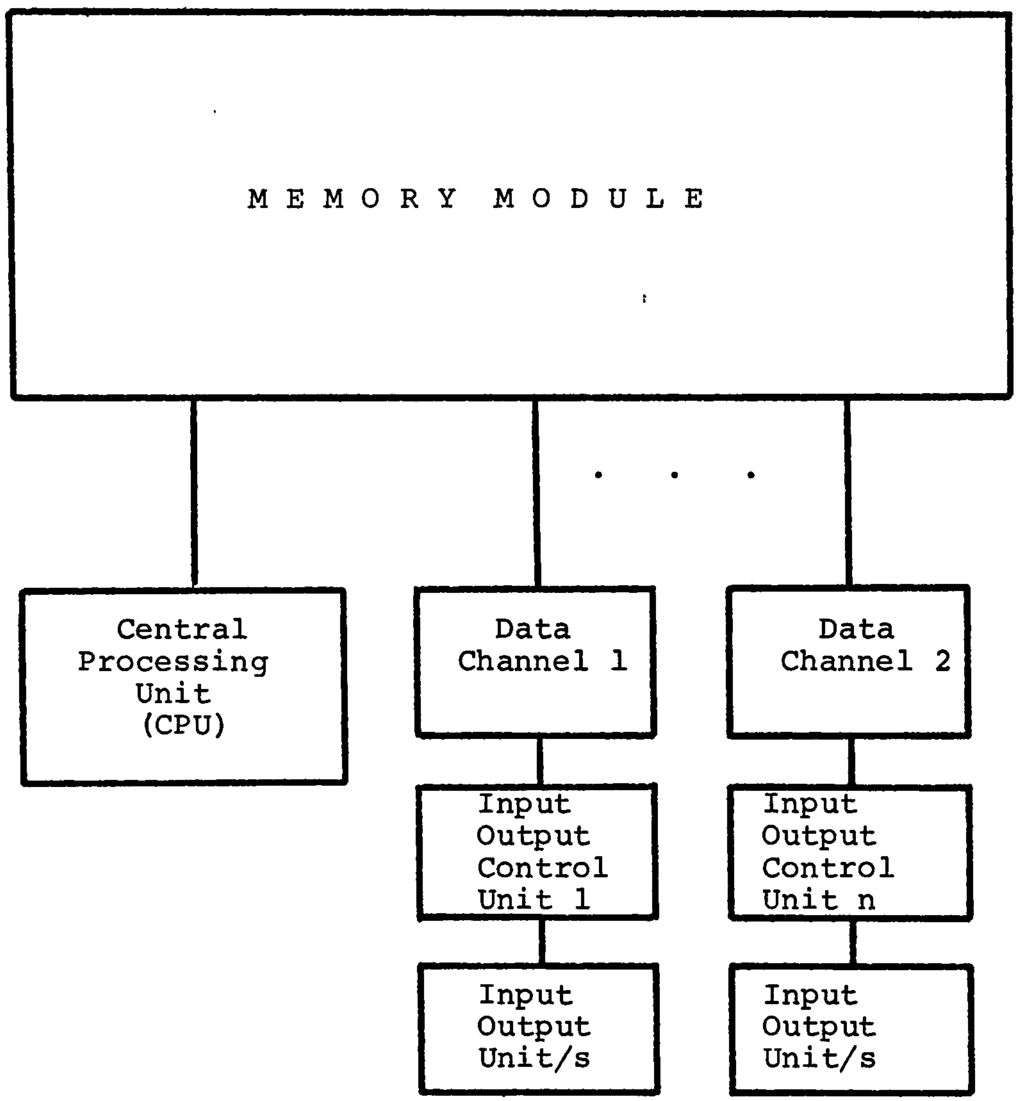



Figure 1

Physical Computing Architecture

it is only during such memory cycles that the CPU is not capable of immediate operation involving use of the memory module. Contrary to the near continuous operation of the CPU, data channels operate only when such operation is initiated by the CPU.

Since the CPU and data channels are capable of performing operations independent of memory references, memory is not necessarily referenced during each memory cycle. A data channel, when active, is usually performing operations concurrently with the CPU; it is because of this simultaneity that a significant amount of data transfer processing can be performed in parallel with the execution of user program logic by the CPU.

Physical Processing

The actual processing of user work, to a great extent, is dictated by the computing system's supervisory software. In the assumed computing environment, while a given job is being processed, there are system allocations over the absolute time base to other than the current job being processed by the CPU. In the interest of performance enhancement, printing requirements of jobs previously processed have been stored or saved on direct access disk space; the supervisory or control program will attempt to keep the printer and its associated data channel busy while the CPU is performing its main task of processing the program logic for the current user job. Similarly, the system software will also attempt to read the card deck job stream into an internal queue during the CPU processing of program logic for the current job. Consequently, occasional channel interference is incurred by the current job which is associated with printing or job deck reading for other jobs. Other time allocations not associated with the current job occur when the supervisory software performs such other tasks as job accounting or performance measurements. Figure 2 illustrates the CPU and data channel activity considerations just described.

## Characterization of the User Job Mix

A job can be defined as the smallest independent unit of user demand. As to content, each job is a formally structured set of system understandable requests and instructions. In this context, a job is independent in that it is a complete specification of a particular user requirement not necessitating interaction with other jobs.

A user first conceptualizes the job and subsequently takes definitive steps to accomplish the required structuring necessary for job execution by the computing system. Demand for use of the computing system is not assumed to exist until the moment a job has been completely structured and submitted for actual processing by the computing system.

In this section, discussion is directed toward the attributes by which user jobs or collectively the user job mix can be identified; as such, each job can be viewed as

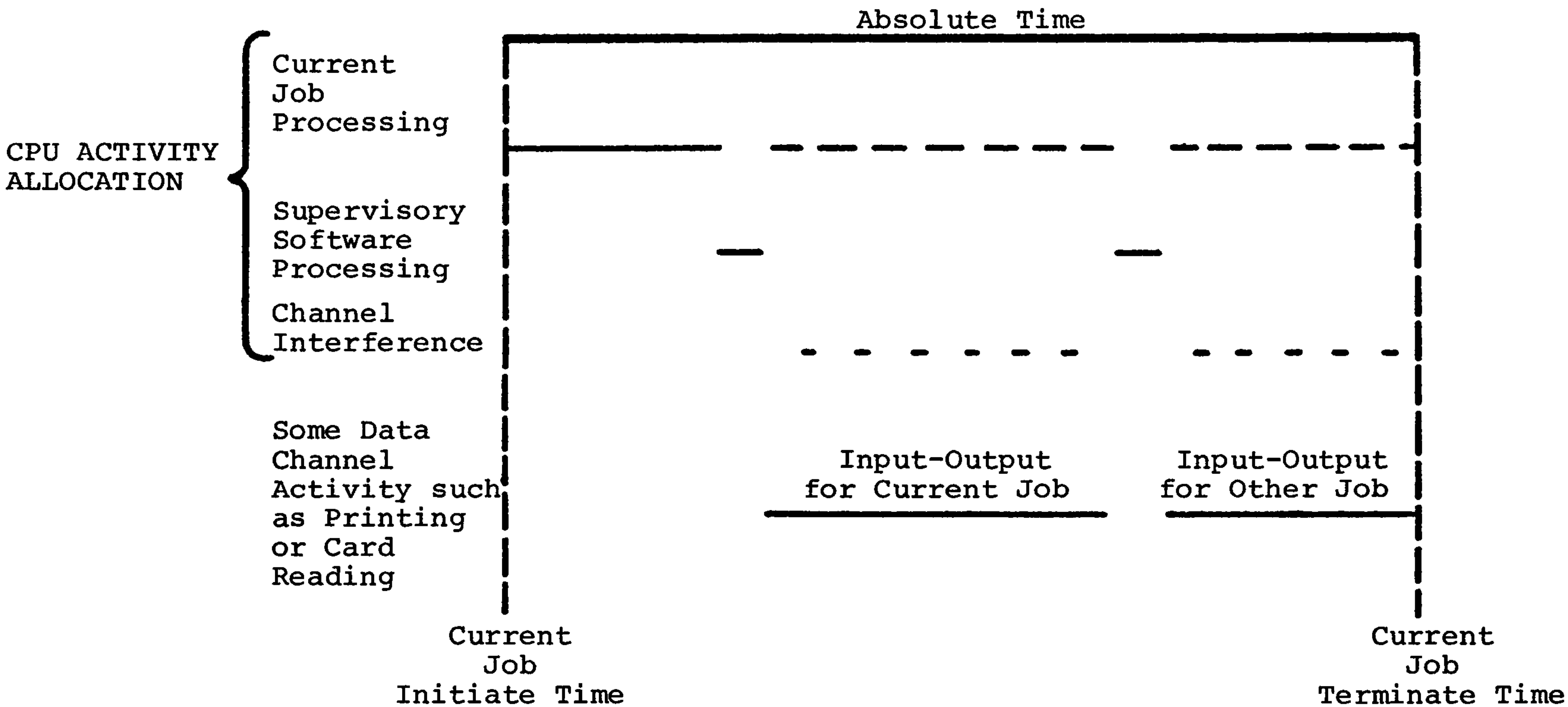



Figure 2

CPU and Data Channel Activity Allocation
Between Current Job Initiate and Terminate Times

some vector of identifying attributes.

To begin with, jobs are either regularly submitted (e.g., weekly payrolls and monthly accounts payable) or irregularly submitted (e.g., program development and certain profit center applications). Hence, one of the attributes characterizing a given job could be termed job submission frequency.

As indicated in Figure 2, the absolute time period between the initiation and termination of a given job includes system resource allocations to jobs other than the given job; this can make the measurement of a job's processing time by the difference between initiate and terminate times somewhat inaccurate. Conveniently, a job's demand on the CPU as measured, for example, in CPU-seconds is quite independent of performance enhancements such as multiprogramming and multiprocessing. Hence, another important job characterizing attribute is CPU time or the time during which the CPU actually spends processing the program logic associated with the given job.

Many other important job characterizing attributes may be cited along with job submission frequency and CPU time. Other such attributes are the input-output activity loadings on the data channels. Channel loadings can be realistically approximated by input-output volumes (e.g., cards read, cards punched, lines or pages printed, disk access and data flow). Other job attributes could include the number of tape units required, the amount of scratch disk used, and the memory space (cells) required.

Selection of job characterizing attributes is assumed to depend on the computing environment and the use intended for such attributes. For the attributes just introduced, the intended use has been to give an indication of the relative amounts of computing resources which are allocated to a given job each time it executes. Certainly other types of job attributes could be formulated for diverse use. For example, the relative performance achieved by a given user job might be appropriately portrayed by some measure of efficiency of CPU and data channel utilizations. However, selection of attributes for the job mix in the sense suggested has been directed toward the allocation of computing resources to user jobs rather than toward other considerations such as job performance.

These attributes pertaining to resource allocation are not invariant; that is to say, they are not necessarily constant in value. This assumed variability in attribute values can be realistically represented by random variation according to some assumed probability distributions; this random variation of attributes is not assumed to be independent or uncorrelated.

Some attributes (e.g., job submission frequency) are probably best represented by discrete probability distributions and others (e.g., CPU time in small time units) by continuous distributions. A reasonable discrete probability distribution has been assumed to be the Poisson; for the continuous case, the Gaussian or Normal distribution has been assumed. Each of these probability distributions will tend to have a centrally-located mean in the range of possible values; this fact seems intuitively appealing in terms of the selected job attributes.

The foregoing description of job mix characteristics for the purpose of relating allocated resources to user jobs is summarized by the following notation.

(1)

$$J(i_j) = \begin{bmatrix} E \\ F \\ W \\ R \\ P \\ M \\ D \\ \vdots \end{bmatrix} \qquad 1 \leq j \leq k; \qquad 1 \leq i_j \leq n_j;$$

$j$ - positive integer user subscript

$k$ - number of users

$i_j$ - the $i^{th}$ job for user $j$

$n_j$ - the number of jobs associated with user $j$

$J(i_j)$ - vector of job attributes; each attribute indicated is considered to be a random variable with mean, m, and variance, v

Vector attributes:

E - CPU time
F - job submission frequency
W - print lines written
R - cards read
P - cards punched
M - memory space (cells)
D - disk space (cells)

(2)

$$N = \sum_{j=1}^{k} n_j \quad ;$$

N - total jobs

(3)
$v_F = 0$ for regularly generated job demand

$v_F \neq 0$ for irregularly generated job demand

Note: All the job mix variables are a continuous function of time, t

Work Load or Job Mix Turnover

Turnover of work load is another notion about the job mix that is pertinent to later discussion. At any point in continuous time, t, there is a user job, $i_j$, which has the smallest mean or average job submission frequency, $\min(m_F)$; this smallest mean submission frequency represents the longest mean time increment between successive submissions of the same job or max(MTBS). Within the time span, max(MTBS), it is assumed that all other jobs are submitted one or more times (i.e., $F \geq 1$).

Also, it may be pointed out that the job mix is assumed to be very dynamic in that each user is developing new jobs and discontinuing the use of previously developed jobs which have become obsolete. This phenomenon can be represented as a continuous function of time using the following notation:

(4) $$n_j(t) = n_j(t_o) + (n_{Cj}(\Delta t) - n_{0j}(\Delta t))$$

$n_j(t_o)$ - number of active jobs for user j at base time $t_o$

$n_{Cj}(\Delta t)$ number of jobs created over time increment, $\Delta t$ for user j

$n_{Oj}(\Delta t)$ number of jobs obsoleted over time increment, $\Delta t$, for user j

$\Delta t = t - t_o$

Computing System Allocation not Directly Associated with the Work Load

As mentioned previously, certain absolute time allocations during the time between current job initiate and terminate times are devoted to endeavors not directly associated with the current job; during the processing of the current job, computing system allocations are also devoted to input-output tasks for other jobs and some indirect supervisory software activity. The over-all computing system allocation during the time period between job initiate and terminate times is assumed to be essentially work load related.

There are, however, other over-all computing system allocations not necessarily within the limits of a job's initiate and terminate times; these other allocations will be seen to be not directly related to the work load. Such other allocations have been categorized as follows:

Note: Each of the time allocations shown below is treated as a two parameter (i.e., mean and variance) random variable with respect to both duration and frequency of occurrence, or

$m_D$ - mean duration time of occurrence, D

$v_D$ - variance in duration time of occurrence, D

$m_{F'}$ - mean frequency of occurrence, F'

$v_{F'}$ - variance in frequency of occurrence, F'

System Not in Use - The system is either on or off, in serviceable condition, but is not being used.

$NU_{off}$ - not being used and off

$NU_{on}$ - not being used and on

Maintenance - The system is not available to the user community due to either preventive (scheduled) or emergency (unscheduled) maintenance requirements.

$M_U$ - emergency (unscheduled) maintenance

$M_S$ - preventive (scheduled) maintenance

Operational Overhead- The time during which the system is otherwise available and in serviceable condition but is not capable of being accessed by the user community until operating personnel complete some external action required for continued processing.

$O_o$ - operational overhead

Software Overhead - The time spent by the supervisory or control software not directly associated with a particular job.

$O_s$ - software overhead

The rationale for presenting this set of system time allocation categories was to provide a framework within which to discuss capacity considerations.

## Capacity Considerations

Johnson (1970) has defined capacity as "the total information work executable per unit time with a balanced work load. A balanced work load is that set of tasks which fully utilize all of the separately accessible resources of a computer system. Such work load might be imaginary."

Singer (1968) defined capacity "to mean the ability to process some maximum number of jobs (per time period) of given quality of service, or turnaround time."

These two definitions of capacity express the fundamental notion upon which presentation of some specific capacity considerations for the assumed computing environment can be based.

To view capacity of the over-all system, consideration must first be given to the fact that due to simultaneity of CPU and data channel operation, there are separate capacities associated with the CPU and each data channel.

Previously, it was implied that the actual data channel capacity was equal to or greater than any of its input-output units; hence, what is essentially involved is the capacity of the input-output units associated with each data channel. To estimate the capacity sufficiency of, for example, a printer whose rated speed of printing is (a) lines per minute, a determination must first be made of the total estimated print demand or,

$$(5) \qquad D_p = \text{total estimated print demand over max(MTBS)}$$

$$= \sum_{j=1}^{k} \sum_{i=1}^{n_j} (m_W m_F)_{i_j}$$

Furthermore, the system capability of meeting this estimated demand is calculated by determining the maximum number of lines theoretically possible over max(MTBS) and subtracting those time allocations during which printing cannot take place or,

(6) $S_p$ = estimated print capacity over max(MTBS) $= \max(\text{MTBS})(a) - O_{op} - O_{sp} - M_S\ M_U - NU_{off}$

Note that operating overhead and software overhead with a p subscript indicate that these items are subsets of $O_o$ and $O_s$. Examples of $O_{op}$ are the time to change forms on a printer or the time to correct a forms tear. $O_{sp}$ pertains to allocations such as the time it takes the supervisory software to begin another print sequence on the printer.

Now by comparing $S_p$ with $D_p$, the sufficiency of print capacity can be estimated by other than more arbitrary means.

Similarly, CPU capacity appraisals can be made by first estimating the demand over max(MTBS) or,

(7) $D_{CPU}$ = total estimated CPU $= \sum_{j=1}^{k} \sum_{i=1}^{n_j} (m_E m_F)_{i_j}$

Also, the supply of CPU capacity is estimated as,

(8) $S_{CPU}$ = total estimated supply of CPU capacity over max(MTBS)

$= \max(\text{MTBS}) - O_o - O_s - M_S - M_U - NU_{off}$

By comparing $S_{CPU}$ and $D_{CPU}$, a determination as to the sufficiency of CPU capacity can be made.

Other capacity determinations for disk and tape can be made and formulated in an analogous manner.

Under investment or over investment in supplying computing capacity has an effect on resource allocation. For purposes of this study, it was assumed that capacity is periodically reviewed and the result is an appropriate supply of capacity in relation to the demand and other factors such as installation growth. Over the time horizon dealt with in this investigation, capacity was assumed to be fixed.

Assumptions Concerning the User Community

1. The decentralized decision makers or users (profit centers or overhead centers) have goals which are consistent with the goals of the larger organization.
2. The users seek to maximize their goals subject to some budget constraint, in circumstances in which they are charged for their use of the computer.
3. The user's budget is provided by a larger organizational component which is itself subject to a budget constraint.
4. The size of a user's budget reflects a judgment by the larger organization as to the relative importance of the goals the user is attempting to achieve.
5. The users behave in a rational or logically consistent manner.

## DESCRIPTION OF THE INVESTIGATION

Literature collected and researched either directly pertained to or was analogously related to the practical problem of pricing internal computing use for more effective and equitable resource allocation. The literature collected and researched over the period of this investigation is included in the thesis REFERENCES section.

The time horizon for the pricing problem was the short-range as the job work load is processed by the computing facility.

Particular attention was devoted to what considerations should be tendered in order to develop an equitable pricing policy for a particular computing environment. Also sought was methodology as to how the pricing structure would actually be established for a particular environment.

Since it is necessary to know how, when and where resources were allocated, methodology was sought as to how measurement of resource consumption could be accomplished.

Some experimentation was done specifically to measure work flow or jobs through an actual computing facility. This effort was accomplished by writing a computer program in FORTRAN which would plot usage history against time based on job initiate and terminate times.

## Chapter 2

## SELECTED LITERATURE VIEWPOINTS ON PRICING POLICY CONSIDERATIONS

The material in this chapter is a recounting of literature references selected by the writer which pertain to considerations to be tendered in the formulation of a pricing policy for computing resource usage related to the assumed computing environment; it was not the intent of the writer to have presented an exhaustive expository of material related to the pricing phase of the research problem. Explicitly, the writer, following his investigation of the literature included in the REFERENCES section, chose to present viewpoints which he felt appropriately described a comprehensive collection of pricing policy considerations; accordingly, it was the opinion of the writer that the references Diamond and Selwyn (1968), Nielsen (1968), and Nielsen (1970) met this selection criterion. The nucleus of these selected references form the basis for material found in this chapter.

Two viewpoints, that of Diamond and Selwyn (D and S) and that of Nielsen, have been recounted in this chapter. The viewpoints are presented in sequence; any key terms not already defined or not felt to be common knowledge to individuals engaged in this area of research will be introduced as such terms arise in the recounting of each viewpoint.

## THE DIAMOND AND SELWYN VIEWPOINT

Diamond and Selwyn (1968) presented a rather generalized outline for pricing policy formulation and analysis. The D and S article also discusses actual and planned pricing experience with large, sophisticated MIT computing environments such as the Project MAC Compatible Time Sharing Service (CTSS) and Multics; even though these computing environments are quite different and much more complex than the assumed environment chosen for this investigation, many practical points of experience were presented which are pertinent to any environment.

Before beginning a discussion of the topics in the D and S outline, an introductory statement is given next in order to develop an appreciation for how D and S perceived the interaction of users with the <u>utility</u>.

### <u>User Community Interaction with the Utility</u>

<u>Utility</u> means any computing facility supplying a broad range of computing services to a user community. The user community is seen to use the computing facility either as a <u>programming</u> <u>service</u> or as a means of using <u>utility subsystems</u>. The programming service notion suggests that the user himself develops logic or code to perform a desired task. On the other hand, use of the utility as a means of accessing subsystems refers to the idea that the user is merely requesting that a task be performed by a specialized pre-developed subsystem of logic supplied by the utility for

common use. In either case, the user desires a task to be performed and he will either have to develop the task performing capability or request a utility supplied task performing capability. Thus, the concern of the user is generally directed toward the task to be performed rather than toward the steps the computer must perform in order to accomplish the task.

Also, the users are seen to be periodically evaluating the trade off among satisfaction of various computing application requirements in relation to an existing budget constraint. A budget constraint is used in the sense that such a budget is solely for allocation to computing requirements as opposed to general budget funds which would be applicable to other endeavors within the firm. Smidt (1968) refers to a budget exclusively for allocation to computing resources as "soft money" and general budget funds as "hard money."

There are three main topics presented in the D and S pricing policy consideration outline; these topics are the Pricing Framework, Service Framework and Pricing Policy Selection Criteria.

## Pricing Framework

D and S maintain that a pricing framework must be formulated in order to compare and analyze alternative pricing policies which might be under consideration for the utility. Here, pricing policy is taken to mean a set of rules whereby users are charged for access and use of the utility.

Within the pricing framework, D and S postulate three dimensions; these dimensions pertain to the following measures:

1. The actual amount of resources the user requires

2. The relative demand upon the total capacity of the system at any given time

3. A measure of the value of the user-utility interaction process.

<u>Pricing dimension one</u>. Reflecting on the first dimension of the pricing framework, it can be seen that direct funding (average cost pricing) is certainly responsive to resource consumption since the user is being charged in relation to resources consumed. Such is not the case for indirect funding in that the user is charged on an indirect basis of pooled overhead accounts provided for allocation back to profit centers in relation to criteria such as man-hours of direct labor. D and S point out that the resources-consumed dimension is the simplest of all pricing framework dimensions toward which a pricing policy can be properly made to respond.

<u>Pricing dimension two</u>. The second pricing framework dimension says that a pricing policy should be tuned to weigh a given user's demand in relation to the available capacity of the utility at any point in time. This relative

demand notion embraces not only the demand imposed by the given user, but the balance of the demand imposed by all other users which have gained access to the utility at any point in time; the balance of the demand imposed by all other users is not a significant factor for the assumed computing environment. This dimension also suggests an appreciation for the diseconomies associated with those users who desire access but due to maximum capacity limitations must wait for or be refused service. Clearly, the relative demand dimension is associated with the notion of priority. The pricing considerations for priority control are usually complex and dependent on the environment. For the sake of generality, D and S did not discuss at any length the treatment of priorities; some facets of priority pricing have been formulated analytically and example references to such formulations are Greenberger (1966) and Marchand (1968).

It is noteworthy to consider any such priority schemes which existed in the assumed computing environment. For the assumed environment, administrative rulings were imposed to cope with the underlying inability to accept a mere first-come-first-served job processing scheme. Presently, the point to be made is that a necessity for priorities existed and a mechanism to satisfy this need was established.

Pricing dimension three. The third dimension of the pricing framework is the most complex with respect to proper responsiveness on the part of the particular pricing policy. Pricing dimension three pertains to the value of the user-utility process and is concerned with the nature of the work being done by the utility. The essence of the dimension calls for how the user views the system that is serving him. D and S have formulated this value dimension to embody the extent to which an application-oriented user need is satisfied. D and S meant for this dimension to be directed more toward the case where the user perceives the utility as a means of accessing pre-developed application subsystems such as stock quotation, legal information retrieval, and invoice preparation. Here, pricing concern is not directed toward resource consumption (e.g., CPU time) or priority but toward what is valuable to the user in meeting his application subsystem requirements. Therefore, dimension three is associated with an appreciation of the results the user expects from his requests (e.g., fast, accurate, and repetitive service upon request). This programming service and application subsystem value dimension says that the pricing policy should assess the user in terms of the value he sees himself benefiting from; this value, in the case of application subsystem use, will be couched in possibly a different user value framework for each such application subsystem.

## Service Framework

With the fundamentals of the D and S pricing framework introduced, attention is now turned to the concern for the nature of services that the utility is supplying. D and S require that a utility be reasonably well "characterized in service space." That is, in order to formulate a pricing policy, we must first have a framework within which to compare alternative policies and second, we must have a well defined statement of the services the utility is supplying.

Similar to the dimensions of the pricing framework are the dimensions of the service framework. The order of presentation of the service dimensions is in reverse order to the order of presentation of the pricing dimensions with respect to the relative importance of these dimensions to the utility's users. Pricing dimensions were presented in ascending order of such importance (i.e., resources consumed, priority, and value dimensions).

Service dimension one. The first dimension of the service framework relates to the previously introduced distinction between the user supplying or the utility supplying the desired task performing capability. When the user supplies the task performing capability, the utility is being used as a programming service; if the utility is being used as a means of providing this capability, the user is accessing a utility subsystem. As previously implied, service dimension one is of most importance to the user.

Service dimension two. The basic idea behind the second service dimension is user access. Dimension two requires a specification as to how a given utility might be accessed by the using community. Certainly, there are a number of facets to user access; D and S suggest that these facets of access be characterized as

1. the system components eligible for accession,

2. the rate at which the system components may be accessed,

3. the time periods when the various system components may be accessed,

4. and, the availability of priority access.

These access considerations were by no means intended to represent a comprehensive listing of all points pertinent to the user access service dimension, but this listing was intended to convey the concept and scope of this dimension.

Service dimension three. Service dimension three concerns the physical demands that user jobs in execution place upon the utility. Arbitrarily, D and S propose that this service dimension include such specific demands as CPU time, main memory residence space, mass memory residence space, data path usage, and input-output unit usage. Since the D and S article is associated with more complex computing environments, certain of the cited demands may not require such emphasis in the assumed environment.

## Pricing Policy Selection Criteria

D and S now explore considerations in keeping with criteria applicable for selection of a pricing policy for a specific utility. Pricing policy selection criteria as related to by D and S are not stated in such definitive terms or dimensions as were the pricing and service frameworks. In close examination of the D and S viewpoint, the writer feels that D and S intended that pricing policy selection criteria should always take the form of an open ended enumeration of considerations.

Pricing policy selection effort must consider both market-oriented and operation-oriented criteria. Several points are contained in the D and S discussion of market-oriented pricing policy selection criteria; these various points include cost predictability, payment for services used, service maximization, payment for common costs, payment for the value of service, and payment for priority access. Operation-oriented criteria will be discussed after discussion of the market-oriented criteria which is presented next.

Market-oriented criteria. Market-oriented pricing policy selection criteria pertains to a feel for user preferences with respect to the services offered by the utility. Pricing policy might have to be tuned by application of market segmentation whereby prices and services are tailored to the needs and desires of various user classes within the user community. To apply market segmentation, each user class would be viewed by utility management with respect to its characterizing usage requirements. An example of this idea is the targeting of a utility subsystem to a certain user class or market segment.

One of the main aspects to market-oriented criteria is the rather common desire by users that they be presented with a pricing structure permitting them to predict or anticipate their cost. The user preference to be able to anticipate cost is intuitively appealing from the standpoint of analogous situations in everyday business. If we look, for example, at common stock transactions and consider a market order to buy (sell), it is noticed that even here, though the buyer (seller) cannot predict exactly the trading price per share, he most definitely would have some idea of the range of possible prices to which he might be subject. The case where price anticipation would not apply would exist where either the user had unlimited funds or was not subject to a budget constraint.

Another very important user preference is the desire to pay only for resources consumed. By resources, D and S refer to resource micro units such as CPU time units, lines of print, disk space, and data channel usage units. The interest on the part of the user in wanting to pay only for the resources he used, is to be able to attempt to minimize his computation costs. There exists, however, a potential conflict with recovery of costs connected with leased application subsystems on solely a resources consumed basis; these subsystems can be so costly relative to other costs that investment recovery may not be possible with a resources consumed pricing policy.

Users prefer to maximize service for a given expenditure. Actually, this strategy is another way of looking at the user preference to minimize cost. On this subject of <u>service maximization per unit cost</u>, D and S offer some insight with reference to the comparison of two very simple pricing policies - flat rate and resources-consumed pricing. Flat rate pricing refers to charging the user a fixed amount permitting this user to have unlimited access to the utility for some specified period of time; there is a constraint, however, limiting the access for a particular user in that there is a limit to available capacity. Resources-consumed pricing is the charging for the resource micro units (e.g., CPU time, memory space, disk space) used during the execution of a given user job. The basis for the comparison of the two pricing policies is which policy is better from the standpoint of maximizing service per dollar. D and S propose that there are conditions under which either policy may provide more service per dollar. The support for this argument is given in O'Sullivan (1967) related to a time-sharing computing environment. D and S conclude that it cannot be stated a priori which pricing policy is better.

D and S next point out the user preference for an <u>equitable distribution of operational and software overhead costs</u> back to the users; D and S maintain that the approach for distributing the overhead cost must be done on some basis whereby resource usage is well correlated with the overhead caused or generated by individual users. Within the assumed computing environment, overhead is generated by external requirements such as tape mounting and card deck loading and internal job requirements causing hardware interrupts which activate the supervisory or controlling software.

Throughout the D and S viewpoint, the concern for price sensitivity to user value received is continually stressed. Now as all aspects of criteria for pricing policy selection unfold, it should, at least, be reasonable to suppose that there exist pricing policies which appreciate the value of the utility to the user. Assuming that the goals, interests, and value systems of the using community are consistent with the goals of the computing installation and that of the organization as a whole, some basis must be chosen whereby value received measured in necessarily different terms for diverse user classes can be mapped into a common basis for value appreciation. This notion can be seen to be associated with considerable complexity.

The last of the market-oriented pricing policy selection criteria refers to the user preference to receive priority access to the utility at premium prices. This preference concerns the user desire for priority with respect to the amount of resources he can receive relative to the other users at any point in time. Premium prices are necessary for this preferred access which temporarily denies service to other users.

Another notion of priority required from time to time by certain users is the price determined capability to gain access to the utility when the system is fully exploited or operating at capacity; D and S say that this situation represents a special case of the previous priority preference.

Operation-oriented criteria. The basis for operation-oriented pricing policy selection criteria is the concern of management. A goal must first be defined whereby its degree of attainment may be measured. D and S discuss operation-oriented criteria in terms of the profit motive of a commercial computing utility selling its services to outside customers. For the assumed nonprofit internal usage situation, the management goal in behalf of the computing facility might be broadly stated as-- . . . to continually attempt to provide the most efficient general purpose computing facility consistent with available resources. This statement pertains however to all aspects of computing facility management; for this investigation, pricing policy analysis was the main topic and therefore discussion here was limited to how a given pricing policy would relate to such a goal.

Any pricing policy is by D and S to be composed of two elements -- the price mechanism and the unit prices. Now D and S, being interested in value-oriented prices, suppose the possibility of several pricing policies within the over-all pricing system for the utility; as stated before, this notion of value pricing for different market segments is associated with the problem of having to map different value-received bases into a common value statement. In this regard, D and S conclude that to make the over-all pricing system responsive goal attainment, one of the requirements is to make a determination of the relative demand upon the utility by the user market segments or classes.

Effectiveness of the pricing policy is one aspect of management appraisal relating to goal attainment. Cost effectiveness is a necessary element consistent with over-all business goals. In terms of the pricing policy, cost effectiveness can be achieved by making the pricing policy as simple as possible. D and S maintain, as do Nielsen (1970) and Smidt (1968a), that the more complex the pricing system becomes the larger will be the user work flow degradation due to pricing calculations. D and S point out that complexity of the pricing system will most likely be related to the number of items on the computing services "menu" -- the more items, the higher the software overhead associated with accounting for services performed in behalf of the user community.

## Pricing Policy Effect on User Behavior

D and S have gleaned some fundamental knowledge of the behavioral response of users to pricing policy.

In January 1967, about forty-five Project MAC users from the Sloan School of Management experimented with a market mechanism developed by Selwyn (1968) whereby users were given fiat money (i.e., redeemable only for the use of computing resources) with which to bargain for utility access. Itemized as follows are some of the points concluded from this experiment relating to user behavior within the induced market environment.

1. It is possible to influence user behavior with pricing.

2. The rather small market did not represent pure economic competition.

3. Because of the lack of a pure competitive market environment, some additional control of user behavior was necessary.

4. The users did trade among the several system resources.

Further experimentation was performed with the market mechanism by combining pricing and rationing in enabling users to purchase access to system resources. The rationing imposed allowed users to gain access to the system resources up to some maximum level for each user. This was necessary since without a pure competitive environment, monopolistic practices would be able to flourish. Additional rationing and control was levied by limiting the amount of additional resources that could be acquired at any point in time by a given user.

## THE NIELSEN VIEWPOINT

Nielsen (1970) is an updating of an earlier article, Nielsen (1968), in order to better focalize the computing resource allocation problem. Nielsen's latest article provides an easy-to-read and practical statement of what to anticipate and provide for in formulating a pricing policy. To give more insight into his views, Nielsen has indicated what might be expected from specific pricing policies and he has also reflected on his experience at the Stanford Computing Center and the Stanford Linear Accelerator Center. His latest article concludes with a discussion of present misconceptions about computer resource allocation.

Much of Nielsen's thinking about the present problem of the misallocation of resources was presented earlier in Chapter 1. Emphasis here will be directed toward interpretation and presentation of his latter views (Nielsen (1970)) which are applicable to the assumed environment. Nielsen (1970) emphasizes that the particular details of the resource allocation problem vary from computing environment to computing environment but that the basic concepts which he presents are applicable to most all computing environments. Nielsen indicates that his views are most applicable to environments serving a number of users on a job shop basis. Dedicated systems such as for information retrieval are more application-dependent and less generalizable to other systems.

## Analysis of the Resource Allocation Dilemma

Nielsen begins by pointing out that there exists a shortage of computing capability as evidenced by users who either must wait for access to the facility or, during processing, are unable to gain access to the desired quantity of resources. Individuals in computing circles have agreed that there is a need to more appropriately allocate the limited computing resources to the much greater demands of the users; however, until recently, this general agreement has not been supported by much interest. Nielsen said that this lack of interest was due to the existence of pre-third generation computing environments and the viewpoint held by computing center management that the resource allocation problem was outside its area of concern.

Environmental difficulties. Nielsen would look upon the assumed environment as a single stream processing system with off-line batching. He cites this environment as being responsible to some degree for the lethargy in moving to improved computing resource allocation; because access to single stream systems is quite limited, the flexibility of centralized resource allocation is also limited. He points out, however, that increased motivation toward improvement of resource allocation methodology is being stimulated by the advent of multi-access computing environments.

The misunderstanding in operating philosophy. Nielsen implies that there is reason to suspect that some computer center managements have not recognized the undesirable aspects to certain resource allocation approaches cited in Chapter 1. These managements fostered operating philosophies with obvious earmarks of a misunderstanding as to what an economically sound operating philosophy should consist of. As a result, these managements felt their attention would be better focused on much more important endeavors than resource allocation enhancements -- resource allocation enhancement was clearly a problem for the economist and not the management of a computer center. This misunderstanding was easily evidenced by the existence of the default allocating approach known as FCFS; by taking no direct steps to institute economically sound resource allocation policies, does not mean there will be no resource allocation approach. FCFS is not at all an undesirable allocating approach providing the value of the demand for the scarce resources is the same. Because all work processed at a computing facility is not of the same value, it soon became apparent to center managements that arbitrary administrative rulings, as introduced in Chapter 1, were necessary in order to adjust the FCFS approach such that the relative importance of different jobs could be appreciated by establishing some priority.

## Levels of Allocation

Nielsen cites a two-stage allocation. At the top level is a policy group which makes decisions about the relative amounts of resource utilization by the various users. At the lower level, the users would make detailed resource allocation decisions. It is interesting for the reader to note that computer center management would not concern itself with allocation of resources but with providing as large a quantity as possible of the most appropriate resources. Nielsen feels that this two-level procedure weighs proportionately those considerations best made at each level of allocation; in other words, those who are best qualified to make specific decisions are doing so.

Global resource utilization units. Nielsen continues by stating that in order to allocate at the upper level, the policy group will need to think in terms of some global resource utilization unit. He says it matters not so much what form this unit takes (e.g., dollars, yen, points, or hours) and may for lack of better terminology be termed a Computer Unit (CU). With the global resource utilization unit established, it then becomes necessary to estimate or forecast the total amount of computing which can be done at the computing facility over some period of time. Here, Nielsen's idea is similar to that expressed by the writer in Chapter 1 relating to the available computing capacity over max (MTBS). For example, if it were possible to supply x CU's over a given period of time, the policy group would allocate to users in terms of some percentage of this available capacity.

Exchange rates. Resource distribution at the lower level cannot be achieved effectively in terms of the global CU's. There must be a translation scheme whereby CU's are convertible into specific resource consumption units. Nielsen refers to this conversion scheme as the exchange rates or prices.

Given the exchange rates or prices, the user is free to make decisions concerning his detailed resource allocation requirements in view of or constrained by the policy guideline or budget of CU's.

Nielsen assumes that even though users are acting in their own behalf, their resulting behavior, tempered by the pricing system, will move the computing facility management toward supplying an appropriate mix of resources economically consistent with goals and priorities of the larger organization.

Advantages. Important to the Nielsen viewpoint on two-level allocation are the associated advantages. First, the most knowledgeable people best qualified to make certain allocating decisions are doing so. Second, there will exist an economic motivation to make good decisions. And third, such dual-level decision making would result in the best interests of the entire organization.

User motivation with price controls. As to the application of the exchange rates, Nielsen expresses the importance of being able to motivate desirable user behavior. Nielsen's use of the word -- motivate -- suggests that he anticipates a premeditated control possibly as a function of time. In any case, he envisions prices being altered for different types of service (e.g., current or overnight service). In having different rates for current and deferred service, the users are constrained to consider the relative economics. The end result is that turnaround is reduced for those users who really need quicker response. Consequently, the value of the output as perceived by the users would be greater; hence, with existing capacity, the computing facility can increase the value of the services it provides.

Funding mechanics. The mechanics of how the two-level allocation takes place is a matter which can be provided for with quite a bit of leeway. Nielsen offers two types of funding mechanics. On the one hand, a center might be totally financed for a given period of time on an overhead basis. This method implies that some study has been made as to the order of magnitude of computing capacity required during this period. With capacity acquired and funded, the computing committee (i.e., policy representatives in behalf of computing needs) allocates CU's (i.e., fiat money) to the user groups, and the result is that an economically proper allocation mechanism still exists.

Under a more rigid economic environment, the computing facility might have to function as a profit center, justifying its own existence solely on the basis of direct dollar charges to the users. Certain problems can arise with this latter funding scheme. Such problems are concerned with situations in which some or all of the funding for user budgets come from outside the organization component responsible for supplying the computing capacity. Here, the policy level fund allocation is not within the organization itself. Examples of this funding problem have been noted in university situations where grants are received by research components within the university to allocate specifically to computing applications; if the dependency on such outside funding support is too great and abrupt changes in such funding levels occur, then drastic overinvestment or under-investment can result as evidenced by recent National Science Foundation budgetary reductions for computing research (Huggins (1971)).

## Setting Resource Prices

In this section, Nielsen makes no specific statements as to how prices are to be actually established; it is in keeping with Nielsen's over-all viewpoint that such matters are ultimately dependent upon the characteristics and goals of the particular computing environment.

Pitfalls to a free market mechanism. Nielsen seeks initially to discourage the use of a free market allocating mechanism whereby pure supply and demand determine the prices from moment to moment. As disadvantages to this mechanism, he cites the effort on the part of the users to bid for desired resources and the potentially drastic price changes resulting from demand ebbs and surges. Nielsen sees the user as not being able to adjust significantly his requirements for a mix of resources in the short-run. Because of this present constraint, prices should be relatively stable over periods during which the user can make rather insignificant adjustments in resource mix requirements.

Long-term contracts. Since users in private business are expected to commit themselves as to project deadlines within existing budget limits, there is somewhat of a case for such users to expect to gain access to a certain quantity of computing capability within narrow limits of the associated expenditure variation. This notion might make advisable the conveyance of long-term or advance computing contracts guaranteeing the availability of a certain quantity of computing service with fixed pricing limits. This approach would shift the risk to the computing facility to gain assurance that the particular user could expect the necessary computing service to be supplied within his budget. Nielsen goes on to say that financing embellishments could be incorporated whereby long-term contracts could be marketed much the same way that warrants are marketed on the stock exchanges; this means that a user might enter into a long-term contract at either a premium or a discount depending on existing rates and anticipated direction of price changes.

Computing facility cost recovery. Considering the case where the computer is completely funded by internal sources, Nielsen suggests that the amount of service which can be provided in a given period of time must be approximately valued at the amount of CU's allocated by the policy body. What Nielsen is implying here is that prices are directly addressing the requirement for cost recovery in the long-run. Nielsen warns, however, that an internal pricing policy should not attempt to recover cost on a short-run basis, for this would lead to the pitfalls of misallocation associated with direct funding or average cost pricing.

Advantages of price controls. The subject of short-run price manipulation or control is concerned with the setting of prices to achieve various allocating objectives. A common objective to the setting of short-run prices is to discourage or encourage particular types of use. A somewhat related short-run pricing strategy addresses the problem of system utilization improvement. Here, the idea may be to reduce idle computing time caused by some users running short of computing funds. These users, who might not otherwise access the computer, could be encouraged to use the system by offering lower and lower prices for poorer and poorer service. Eventually the pricing becomes attractive enough so that additional units of service are procured thereby causing the full resources of the system to be better utilized.

Pricing can also be used as a management tool in an effort to maximize the service value provided. With certain resources in heavy demand and other under-utilized, pricing adjustments can be made to redistribute the demand to effect an improvement in utilization of the over-all resource mix.

Conceivably, the most significant and important advantage to pricing is the indication to computing facility management on the patterns of computing resource interests of the user community. Management, by continually examining usage data, gets a clear indication of the resource preferences among the users. Knowing the resource preferences puts management in a position to provide greater or more valuable computing service to the user community without necessarily increasing expenditures on computing.

## Selecting Resources to be Priced

Nielsen continues his emphasis on generality by maintaining that the selection of resources to be priced is a matter for consideration in relation to the particular computing environment.

Resource selection based on service pricing. The emphasis on generality is not necessarily correlated with simplicity. Some complexity is introduced in that Nielsen says that resource utilization should be controlled by service prices and a given service may require several resources; this situation, he says, is an allocation problem in itself. Nielsen leaves this subject with the reader as a point for future contemplation.

Minimizing the number of resources. As general principles to resource selection, Nielsen suggests choosing as few resource measures as possible.

Since resource usage measurement can be an involved and production degrading process, selecting as few resource measures as possible reduces the associated accounting and logging burden; this matter of accounting for and the logging of resource usage will be discussed in Chapter 3.

User awareness of resource consumption. Accordingly, the user should be able to consider resource consumption over which he has control. If there are too many resources to consider or if there appears to be little correlation between the requested service and the trade-offs among resources, the user may choose to react rather indifferently to or even ignore the pricing structure.

Temporal Resource Allocation

In the previous section, physical resource selection criteria was introduced; it was in terms of the physical resources that the users make decisions in contemplation of the satisfaction of their project requirements.

Another type of resource needs allocating; this resource is temporal in nature and as such is concerned with the time when the particular physical resources were dispensed. Since physical computing resources can only be allocated to one user at a given point in time, an equitable scheme is required whereby the order of access is established which will maximize the value of computing performed. Temporal or time-related resources also exhibit scarcity; this scarcity is inherent in that all existing demand cannot be satisfied simultaneously. Finally, as with physical resource allocation, the detailed short-run temporal allocation decisions should be made by the users; it is the users who will have the information and motivation to make these decisions providing the pricing system is used as a rationing mechanism.

Nielsen gives an example of three users desiring to simultaneously access the same physical resource. If the price for this resource is gradually raised, eventually two of the users, after weighing their requirements versus the price, will decide to wait till access may be had at a better price. What has happened is that this procedure has caused each of these users to place a value on either being served immediately or waiting for the other user to finish. The benefits to this decentralized ordering procedure are:

1. The users themselves each establish a value associated with immediate service that results in a service order upon which they will mutually agree.

2. This ordering will maximize the value to the organization of the computing performed.

Notion of critical services. Nielsen suggests that certain critical resources or services should be priced dynamically to solve the temporal resource allocation problem; in this sense, Nielsen has previously implied that other resources are non-critical and should be priced with a fixed or static pricing policy.

To better characterize a critical resource, some examples will be discussed. To begin with, the CPU is an important critical resource in that it must first be accessed before any other resource is used during the processing of a given job. Other earmarks of criticality are exhibited by resources such as the printer where the total work load might easily consume or require more absolute time than the use of any other resource including the CPU; this is another way of saying that if printing time is critical, then completion of total job requirements is ultimately dependent on the completion of printing requirements.

Furthermore, critical resource usage is usually associated with backlogs of waiting demand; this situation is common because these resources are usually popular or in highest use in relation to the available capacity which should be scarce. This notion brings to mind a clearer picture of resource criticality for the assumed environment; it should be easier now for the reader to see why the CPU and the printer are excellent candidates for critical resources associated with an inherent temporal allocation problem. It should not be inferred by the reader, however, that there are no internal requirements for queueing of, say, disk and magnetic tape service loads which might cause such resources to become more critical than printing.

Pricing policies for temporal allocation. The temporal allocation problem, as implied before, is concerned with determining and allocating according to the value associated with different priorities or qualities of service.

Pricing for the temporal problem might be handled with constant rates per shift which could be changed each shift to allocate different priorities of processing.

A more dynamic form of pricing might reasonably be required to allocate resources during fluctuation of demand for critical resources. A basic scheme for implementation of this concept would be to reflect higher exchange rates for periods of higher than average demand and lower exchange rates for periods of lower than average demand. These lower and higher rates might be in relation to the cost recovery price based on average usage over the max (MTBS). Pricing policy associated with the temporal allocation of critical resources is similar to pricing policy for non-critical resources in that the strategy is not to confound the user with a complex and dynamic multi-rate pricing system.

With most of the fundamental temporal pricing ideas formulated, some of Nielsen's views on example pricing policies with which to implement temporal allocation may be explored.

Nielsen first considers a dynamic temporal pricing policy which permits a user desiring access to the computing facility to enter into a contract dialogue with the facility. The user might inquire about a job requiring so much CPU and printing capability, and also may include something about the urgency of his job; the facility would then respond with a price for this request in relation to present and anticipated future demand. This policy approach is somewhat ideal and might be impractical for the assumed environment because of either the system degradation due to the contractual type of overhead processing or the additional hardware and software required to provide such dialogue capability. Nielsen mentions that in his opinion an effective <u>free market pricing policy</u> has yet to be developed using a reasonable expenditure of system resources.

Nielsen next considers a <u>bidding system</u> whereby the user formulates several values connected with different levels of service. A facility algorithm would then "review" the bids and select those jobs for execution which had the most attractive bids. Nielsen indicates as pitfalls to this policy, the time it takes to prepare the bids and the gamesmanship that tends to distract users from the primary effort of solving their computing problems.

As the most promising approach to short-run pricing for temporal allocation, Nielsen seems to favor _a series of service queues based on external job priorities_. There would be an external job priority associated with each service queue; these external job priorities are not to be confused with internal job priorities established by say a scheduling algorithm within the operating system software. Nielsen feels that the main resource allocating advantage to this system is that each user in selecting a queue for his job at a given price, has the opportunity to consider his requirements and budget in relation to the requirements and budgets of other users. This approach, he points out, will not be effective unless potential users can know the loading in the queues at any time and something of the recent behavior of users.

In summary to his views on temporal pricing approaches, Nielsen states that the short-range allocation problem can be handled by permitting the prices of the more critical services to vary with demand. Care should be taken, however, to protect the user from too frequently varied prices.

## Contemporary Misconceptions Concerning Computer Service Pricing

Because Nielsen sees computer service pricing as a potentially important asset in the allocation of computing resources, he discusses certain misconceptions about pricing which should not become widespread.

Concerning the nature of pricing. Pricing, Nielsen says, is a means for allocating scarce computing resources to a competing user community; it is also a means to bring about decision making at certain levels within a given organizational environment. The nature of pricing, however, does not include a dependency on using real money and requiring that the real money be used to recover center costs by way of user charges for computing service received. Nielsen says there are several instances where CU's (or fiat money) are distributed to users to redeem as desired for computing service.

Facility responsiveness to user needs. A computing center can realize direct justification from the users in terms of fiat money. Nielsen gives an example illustrating this concept -- instead of CU's being allocated to users and the center being funded independently on an overhead basis, the parent organization would allocate fiat CU's to users who would redeem them as usual; however, the computing facility would in turn redeem the fiat CU's, received from users, for real money with which to cover expenses. This example points out the fact that an internal computing facility can be made to compete for business even though the charging basis for usage does not impact the financial budgets of the users. Nielsen conjectures that even though this example could be expected to confront computing center management with a more compelling situation, the quality of service which the users could expect should be improved.

User reaction to fiat money price changes. There may be those who maintain that since fiat money does not impact the users in actual monetary terms, there would result a built-in indifference to price fluctuations. This would only be true if users could get as much fiat money as they wanted; in other words, the fiat money would not be scarce. Nielsen reminds the reader that something only has value if it is scarce. Consequently, if fiat CU's were scarce, the users would treat this resource as though it were actual personal funds.

A potential for inflation. For a computing environment in which funding comes entirely from internal sources, it is well for the policy committee to be aware of the potential for inflation. More than likely, it is at times tempting to offer more CU's to a user who has exhausted his budget. This is satisfactory providing that the total outstanding CU's are maintained constant; maintaining constant CU inventory can be achieved by drawing the required units from a reserve of CU's or taking an equal number of CU's from other budgets. If, however, CU's are added to the outstanding inventory of CU's, then an inflationary situation arises -- more potential demand has been given the users than the facility is able to satisfy. An obvious response to the inflationary move is an increase in prices. Raising prices, however, will not affect all users equally. As the users face the loss in value of the CU, they tend to lose faith in the pricing system -- a result which defeats the purpose for which pricing was intended.

Uncertainty with respect to cost or service. Nielsen, in addressing the short-run temporal allocation problem, stated a requirement for prices to fluctuate. Prices that vary, however, introduce uncertainty; the uncertainty introduced was associated with a potential user being unable to predict the cost of his future computing requirements.

Another type of uncertainty which arises is related to the FCFS procedure in not being able to predict the level of service or turnaround time a given job will receive.

Nielsen does not deny that uncertainty can exist but he says that it does not have to be compounded. Concerning the uncertainty related to the expected cost of computing and level of service, Nielsen offers two guidelines:

> A user can either pick a level of service and pay whatever it takes to obtain it (giving him certainty with respect to service, uncertainty with respect to cost), or he can pick a price level and accept whatever level of service he can obtain at that rate (giving him uncertainty with respect to service, certainty with respect to cost).

Uncertainty of computing cost -- a tolerable necessity. If a pricing policy offers more control over level of service, the tendency should be for the users to exercise this control; in doing so, users increase the certainty of service but, on the other hand, decrease the certainty of computing cost. The service and cost certainty trade-off can be evaluated in view of its effect on the user situation. More control over level of service provides an advantage in that personnel costs and project completion deadlines will have smaller variances. Nielsen says that as experience is gained with the pricing system, users will develop a pattern of usage permitting them to do a better job of estimating computing cost in a flexible pricing environment. Long-term computing contracts, mentioned earlier, can also be used to improve the uncertainty of computing cost.

User disagreement as to importance of projects. Nielsen mentions that disagreement among users may arise concerning the relative importance given their projects by the allocating policy committee. Such conflict in personal value judgment may occur more frequently when a computing facility serves some projects receiving outside funding. Nielsen summarizes on this point by saying that this condition is not a drawback to pricing but a broader issue upon which pricing focuses concern.

Pricing systems involve continual education. A pricing system, in order to meet its objectives, must be continually supported with education. New users need to understand the system and existing users must be kept aware of changes and new services.

A problem is inherent in that some users may react unfavorably to the pricing system; Nielsen comments that this user type views the pricing system merely as a means to "always" get priority over others instead of a means of using resources more efficiently. Even in this instance, Nielsen says that most users will be better off and no one will be worse off for having implemented a pricing system.

## Pricing System Experience

Nielsen concludes with some rather interesting experience he gained with pricing systems at two installations of the Stanford Computation Center.

The Campus Facility pricing system. The following is a direct quotation from Nielsen (1970) concerning experience with pricing at this installation:

> The Campus Facility serves the general educational and research computing needs of the University. It has a user community of some 5000 students, faculty members, and researchers. It is totally dependent upon charges for service to meet its expenses. That is, all users are funded (either externally or internally) and buy computer service with dollars. Funds provided by the University are given directly to users, who in turn buy service from the Campus Facility. A basic set of rates exists for the range of services available. This base rate structure is adjusted approximately quarterly. At present the rate schedule places prices on some 18 services, such as terminal rental, leased communication lines, card punching, and batch processing service.
>
> Currently, only two services have what one might consider to be fluctuating rates. For both batch computing and high speed printing there is a set of priority queues, with differing rates attached. Service

is first-come first-served, beginning with the highest priority queue. Roughly speaking, there are six execution queues and four printing queues. There is presently no dynamic pricing for time-sharing services, although this is planned as the load builds up.

The Stanford Linear Accelerator Center (SLAC) Facility pricing system. The following material is a direct quotation from Nielsen (1970) concerning pricing system experience at this installation:

> The SLAC Facility, on the other hand, is an entirely different installation, serving the physicists at SLAC. It has a user community of approximately 200 users, and the operation is completely funded internally. That is, computing appears as a free good to the members of this community. Despite the large capacity of the computer center (IBM 360/91) and the small size of the user community, it was felt necessary to begin moving toward a pricing system based upon the use of Computation Units.
>
> The service provided by this facility is heavily weighted toward batch processing. Accordingly, the planned base rate schedule will consist of only six rates. These are mostly oriented toward measures of computing (CPU cycles, memory space-time, and I/O operations) and toward set-up operations (e.g., disk mounting). However, there are still two services which will require the use of priorities. There will be four priority levels for batch processing service, the priority charge being stated as a percentage surcharge to be applied to the sum of the standard batch processing charges incurred. There will also be a two-queue priority structure for printed output.
>
> Prices are attached to the various services as described above. An estimate is then made of the "value" of the service that can be delivered by the system over a three month period at these rates. This "value" is then the amount of Computation Units which will be distributed quarterly to the various projects at the Laboratory. The lowest priority of service is essentially at no cost. This "bails out" the user who runs out of his allocation during the quarter, and it provides a buffer for any misestimation of the amount of service which can be provided at non-zero rates.

## Chapter 3

## PRICING POLICY METHODOLOGY FOR THE ASSUMED COMPUTING ENVIRONMENT

Part two of the research problem required a statement of pricing policy methodology for the assumed computing environment; such methodology was to have been either suggested by previous research or formulated by the writer. It was the intent of the writer that such methodology would serve as a basis for future experimentation and research; no experimentation with this methodology was conducted by the writer.

### RATIONALE FOR METHODOLOGY FORMULATION

In laying the groundwork for this chapter, the writer felt that the reader should keep in mind the rationale which served as a basis for the formulation of methodology.

In Chapter 1, it was pointed out that the assumed computing environment was subject to undesirable resource allocation mechanisms. In this Chapter, the writer has merely presented methodology by which an improvement of the resource allocation mechanisms might be realized. As discussed by Nielsen (1970), pricing policy implementation is associated with certain "costs," the effects of which may negate any salutary benefits of pricing.

Accordingly, the writer has not intended to imply that the suggestions in this Chapter are in any way the best or optimal. To search for optimality requires a criterion and constraints agreed upon by the individuals involved. Pricing policy implementation is a real world situation and hence is subject to the scrutiny of real people--all with different value systems. A rather simple set of assumptions about the user community and management was given in Chapter 1; the purpose of these assumptions was to rule out the consideration of uncommon possibilities. Thus, lacking a specific real world statement of values, the material in this Chapter can only be thought of as potentially pointing the way for a rigorous search for optimality.

Finally, the writer, based on his experience and research, has arbitrarily selected those attributes and principles of pricing which he felt to be most pertinent to the assumed environment. Thus, it is not to be inferred that such selections are not open to question; in fact, this point brings to mind the purpose of this material--to generate and stimulate more interest in the potential value of pricing to resource allocation.

## REVISITING THE PHYSICAL MANNER OF PROCESSING USER JOBS

In this Chapter, it is well to provide more detail on this topic which was introduced in Chapter 1.

Magnetic disk and tape devices are capable of moving data to and from the memory module at much higher rates than the data movement rates associated with unit record equipment (e.g., card readers, card punches, and printers). Some representative examples of relative data movement speeds are:

| Input-output unit | Data Movement Speed (character frames/second) |
|---|---|
| disk . . . . . . . . . . . | 312,000 |
| tape . . . . . . . . . . . | 180,000 |
| card reader (1000 cards/minute) . . | 1,333 |
| printer (1100 lines/minute) . . | 2,900 |
| card punch (300 cards/minute). . . | 400 |

Because of the relatively slow speed of the unit record equipment, the supervisory software or control program attempts to perform all such operations simultaneously with the CPU execution of the program logic for the current user job. Tactically, this is accomplished by doing all card reading prior to the execution of a given job and all printing and punching after execution of the same job. Any requests for unit record service during user job processing are simulated by reading input card images from a disk input queue and writing punch and print records to a disk output queue. Other input-output requirements associated directly with tape and disk units are performed directly with those units during the CPU processing of the program logic for the current user job. What has been accomplished is that all input-output requirements during CPU processing of a user job are associated with access of either magnetic tape or disk units. This overlapping of slower speed unit record input-output operations with CPU execution of user jobs is known as Simultaneous Peripheral Operations On Line (or SPOOLing). This SPOOLing concept just described is depicted in Figure 3.

As mentioned in Chapter 1, user job access to the computing system resources is done on a modified FCFS basis; computer operations supervision and operating personnel determine the order of processing by administrative ruling and personal appraisal when the FCFS allocation rule conflicts with the local value system.

## ORGANIZATIONAL OPERATING PHILOSOPHY

A computing environment is also characterized by its organizational structure and location within the firm; this concept, the writer refers to as organizational operating philosophy.

A common environment found in contemporary private business has an autonomous centralized organizational structure and is considered a staff service organization for the entire firm. Internally, this service organization might be composed of a business applications group, a technical applications group, and a computing facility operations group. The services to be supplied by this organization would include advisability studies, systems design, programming, and computing facility operations support. Being a staff organization, funding could very well be carried out on an overhead basis as was assumed for the type of environment toward which this research was directed; the costs which such funding

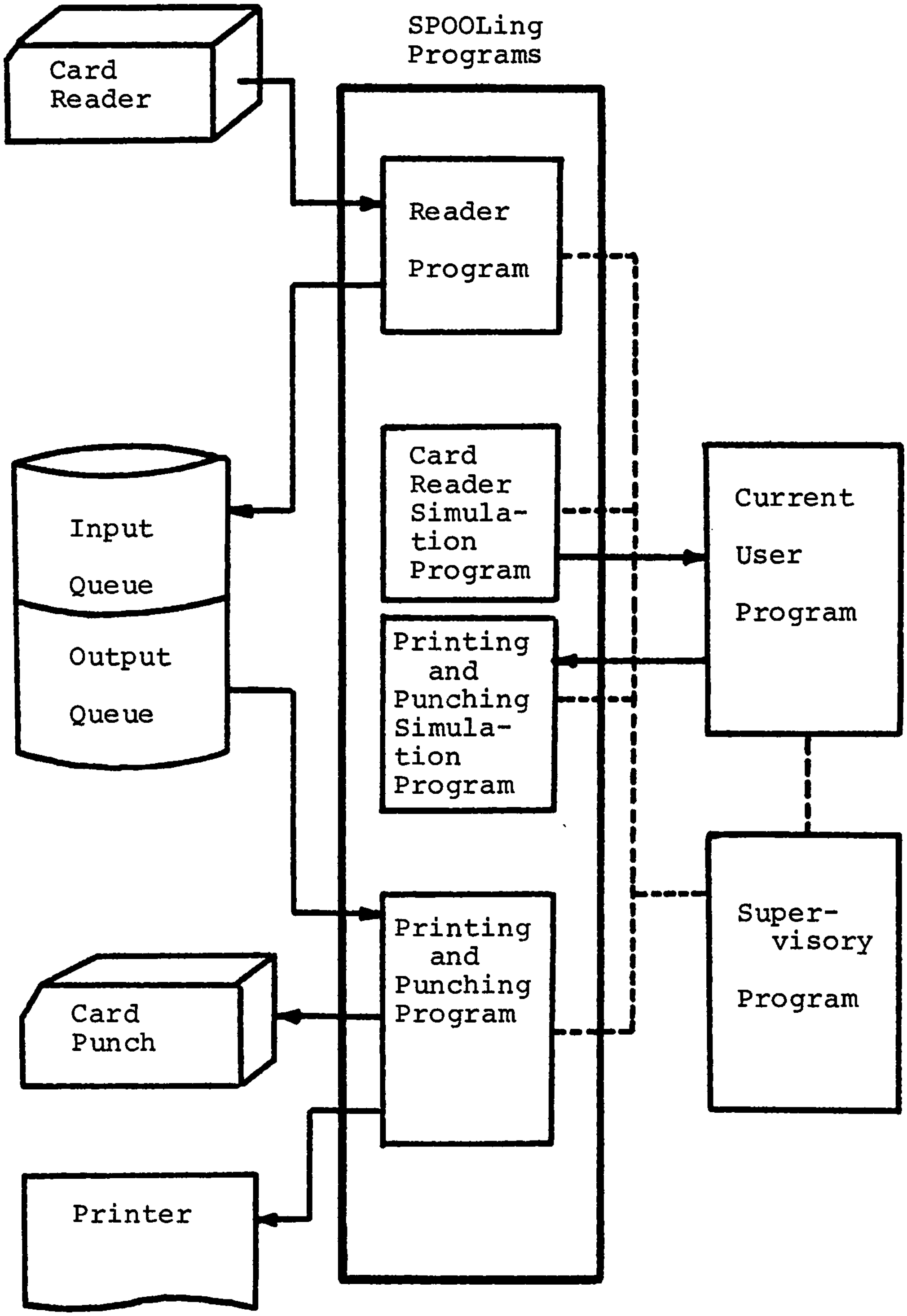



Figure 3

Overlapping Unit Record Operations

might cover would include computing equipment, supplies such as stationery and forms, amortization for start-up and facilities cost, and wages for clerical people, equipment operators, programmers, analysts, supervision and management.

Schroeder (1971) stated that "The effectiveness of the problem solving, application system development service provided by EDP functions is heavily dependent upon the way in which the EDP manager views his own role in the business." Schroeder cited two such EDP manager roles.

Some managers might view their role as related to the technical computer environment. Schroeder said that in this instance, an EDP manager is likely to look at his function "as one of supplying better information faster."

The role which Schroeder endorsed as proper for the EDP manager is a role in which such a manager would consider his job "as one of solving user problems, such as too much inventory, excessively high cost of operation, inadequate customer service." Schroeder implied that this role cannot be attained effectively with an organizational operating philosophy such as previously described in this section. Schroeder suggested that to best achieve this role, "the EDP function must be fully absorbed into the operations of the business." Schroeder amplified his viewpoint as follows:

> EDP cannot stand alone or be isolated from the line and other staff activities. We have found that the degree of EDP absorption into the business is revealed by the extent of its planning for the future and how well those plans are integrated with the operating functions' plans. Only a minority of EDP departments have really effective long range plans.

The writer's interest in this regard stems from his belief that pricing policy for computing resource allocation cannot be realistically considered if the organizational operating philosophy was to be disregarded. Hence, the writer feels that he should take a position on this subject as part of his suggestions within the framework of formulating pricing policy. Accordingly, the writer is suggesting that the merits of Schroeder's viewpoint be seriously considered by top management within any firm currently operating with an autonomous EDP environment.

In Chapter 1, the writer introduced his idea that a trend is underway whereby total overhead support for computing is being changed to a balance of overhead support and direct support impacting operating budgets of the firm's profit and overhead centers. This idea reinforces the writer's support of Schroeder's viewpoint. For by giving the operating centers (i.e., a collective term for both profit and overhead centers) an opportunity to control the dollar amount they are charged for computing, the people doing computing application development should become more and more under the control of the respective operating centers. Initially, in order to stimulate this trend, operating centers might contract with the EDP department (i.e., somewhat as suggested by Nielsen (1971)) for support not to exceed a certain dollar amount; if, for example, there was a contract overrun, the excess might somehow reduce that operating center's overhead charge for computing.

As the trend continues, the organizational operating philosophy will evolve toward a situation in which the only autonomous computing organization would be the computing facility operating staff. This staff would be responsible for services such as efficient operation of the computing facility, user education and assistance, and negotiations with higher management in an attempt to maintain a user acceptable level of available capacity that is consistent with the goals of the firm.

During the remaining discussion in this Chapter, the writer will assume that organizational operating philosophy has taken on an appearance in which all ultimate users are within the respective operating centers and that the only autonomous computing organization is the computing facility operating staff. The nomenclature and functional description of the organizational operating philosophy is given as follows:

managerial function

1. reviewing capacity requirements with those responsible for the computing operations function.
2. reviewing the needs of the firm in terms of computing with those responsible for the project control function.
3. reviewing the organizational operating philosophy for computing service within the firm.
4. this function would be carried out at a level where the strategic viewpoint of the firm could be appreciated.

| | |
|---|---|
| project control function | 1. negotiating with operating centers in establishing job processing priorities within respective computing projects.<br><br>2. coordinating with those responsible for the computing operations function in the determination of applicable costs distributed to resource consumption units.<br><br>3. assisting operating centers in the preparation of their budget requests for computing service.<br><br>4. establishing, reviewing, and modifying pricing policy for computing service.<br><br>5. this function would be carried out at a level where the needs of the firm can be viewed in terms of outstanding computing projects; as such, this function would be closely coordinated with the managerial function. |
| computing operations function | 1. operating and maintaining the computing facility.<br><br>2. educating and assisting users in the utilization of the facility.<br><br>3. coordinating facility requirements with those responsible for both the project control and managerial functions. |

## TYPES OF COMPUTING REQUIREMENTS

The reader should have an appreciation for the types of computing requirements found in the assumed environment.

For instance, accounting and fiscal personnel are primarily concerned with the timely and accurate maintenance of the firm's financial records. This type of computing requirement generally consists of the maintenance of large files and the preparation of reports. Urgency of this requirement type is usually quite high; these computing tasks are commonly associated with the settlement of financial obligations, and the reporting required by government agencies and outside auditing firms. Examples of such requirements are payroll, accounts payable, tax and operating statement reporting.

Another type of computing requirement is of a technical nature. The operating centers often entertain computing tasks associated with general engineering, operations research, and statistical analyses; the requirement involves more complicated computation in conjunction with possibly large amounts of data. Urgency, in this case, requires a relative appraisal in terms of the needs of the firm.

Finally, there are applications which are classified as miscellaneous reporting; this reporting is usually in the form of simple summaries, tabulations, and recaps. Urgency here, as with technical applications, requires a relative appraisal in terms of the needs of the firm.

## FUNDAMENTAL PRICING POLICY OBJECTIVES

The following tabulated points represent the writer's selected objectives for the formulation of the example pricing policy for the assumed environment:

1. Provide a pricing policy framework within which the users,

   a. can readily perceive their computing usage in terms of the consumables chosen for the pricing policy.

   b. are subject to a real dollar constraint.

   c. are charged only in relation to the resource units consumed.

   d. can predict their cost of computing.

   (Note: Users will receive preference relative to their job urgencies and will be compensated for deferred service in the assignment and treatment of job processing priorities.)

2. Only operating centers engaged in computing will bear the cost of the computing facility. At the time of billing, each such operating center's share of the total computing cost will be shown as the sum of two charges -- the direct use charge and an overhead charge based on direct computing use.

3. Prices for resource units will be established on the basis that cost will be recovered without an allocation for overhead only if the particular resource is utilized to its estimated full capacity over a given period of time.

4. In order to control user behavior related to the utilization of computing resources, prices may be changed from time to time by those responsible for the project control function.

## RESOURCE UNITS AND ASSOCIATED COSTS

In this section, suggested resource units and related costs will be introduced. Here the reader should realize that such resource units and costs, although not arbitrary, are intended only as a guideline whereby the writer can more tangibly discuss pricing. The reader should also bear in mind that each of the resource units can reasonably receive a cost distribution for operating personnel support during a particular shift; this relative distribution can be estimated for a particular environment but the major portion will probably accrue to job processing time. Listed next is a description of resource units and related costs:

| Resource Unit | Description and Related Costs |
|---|---|
| job processing time | This period represents the elapsed time during which all job directives are processed for the current job; such processing implies the dedication of all system resources to the current job, except for the unit record operations for other jobs. It is reasonable, therefore, to choose costs for this service which reflect the consumables during this period. Other than operating personnel support (i.e., for a given shift), the related costs will be for hardware such as the CPU, the memory module, magnetic tape units, disk units and the console typewriter. |
| lines printed | This measure is related to hardware costs for the printer, its control unit and data channel. (Note: Forms cost will be accounted for separately on a page basis.) |
| cards read | This measure is related to hardware costs for the card reader, its control unit and data channel. |
| cards punched | This measure is associated with card stock cost and hardware costs for the card punch, its control unit and data channel. |

(continued)

| Resource Unit | Description and Related Costs |
|---|---|
| plotting time | This measure is related to a resource which is used entirely independent of CPU associated hardware. Also, the characteristics of the output from this resource make time a better measure of usage than some physical size of plot. The costs would include all related hardware (e.g., the plotter and the magnetic tape unit driving the plotter). |
| permanent storage time | This measure represents the time during which a user reserves modules of permanent storage which may be manually attached to and detached from the corresponding input-output units. The related costs would include those for magnetic disk packs and reels of magnetic tape. |
| pages of forms | Forms usage might be related to costs for the different varieties of available forms. |

## ESTABLISHING PRICES FOR RESOURCE UNITS

The resource units just described can be categorized in two ways.

Resource units such as JPT, lines printed, cards punched, cards read, and plotting time are all associated with some maximum production level per selected absolute time interval. These maximum production levels exist because the CPU and data channels are individually capable of only so much production during a given period of time.

Other resource units such as pages of forms usage are merely indications of piece-wise usage and are not time related to any maximum usage.

To arrive at unit prices for resources related to a fixed capacity, the maximum production and accrued costs must be estimated for the same base period of absolute time. The capacity levels associated with maximum production can be estimated by the methods suggested in Chapter 1. The unit price is computed simply by dividing the accrued cost by the estimated capacity.

To arrive at unit prices for resources used or consumed is piece-wise fashion, the cost of some lot size is estimated and divided by the number of pieces in the lot.

## DECISION MAKING FOR THE PROVISION AND ALLOCATION OF RESOURCES

Pricing policy must interface with a decision making process that results in the provision for and allocation of computing resources; in effect, this resource control decision making represents a discrete multi-stage process. At each stage, decisions are made -- computing needs must be appraised, facility capacity levels must be evaluated, the pricing policy and priority access system must be reviewed and available computing resources must be allocated to users in the form of budgets. Between stages, the users access the available computing resources subject to their respective budget constraints, the prevailing pricing policy, and the prevailing priority access system. Nothing more than a conceptual viewpoint is suggested by this notion of multi-stage decision making. For example, changes to the pricing policy, the priority access system, and the user budgets may be made more frequently than changes in capacity.

In viewing this process in more detail, one point must be realized from the beginning--a computing facility already exists and must be paid for.

## Appraising Computing Needs

Periodically, operating centers will be queried as to what they think their computing needs will be over some future planning horizon. The planning horizon may be immediately upcoming or for some more distant time period. As to length, this needs appraisal planning horizon should be reasonably consistent with accounting cycles and time periods during which capacity is relatively static. As indicated in the outline of organizational operating philosophy, the appraisal of user computing needs will be accomplished by those responsible for the project control function.

Needs appraisal can be conveniently and effectively expressed in terms of job processing time. Using the history of job submissions, a given operating center can first assess its recent JPT usage. Subsequently, the operating center would attempt to identify any anticipated change in JPT requirements. Forecasted JPT levels for all operating centers engaged in computing would then be accumulated since JPT values can be considered additive for the assumed environment.

## Assessing the Sufficiency of Capacity

Having expressed the needs of the user community in terms of an estimated aggregate JPT value, the sufficiency of computing capacity can be more accurately assessed. Capacity, in the sense of job processing time, would then be viewed as the maximum JPT deliverable over the upcoming planning horizon. It is at this point, that those responsible for the managerial function, the project control function, and the computing operations function consider the requirements for more or less capacity. The results of this subjective consideration would be a decision as to capacity level by those responsible for the managerial function. After providing for any required capacity adjustment, provision must be made for cost recovery and computing budgets must be allocated to operating centers.

Cost Recovery

Cost recovery or funding of computing operations will be provided in terms of actual costs for the operating staff, supplies, equipment and maintenance of the facility environment. These costs will be imputed to resource units based on guidelines similar to those presented in the section on resource units and related costs.

As the resources are accessed, the operating centers will accumulate charges for such usage based on the existing unit prices. At the end of an actual billing cycle (e.g., quarterly, semi-annually, annually), operating centers will be billed for their use of each fixed capacity resource in terms of two charges--one charge will be for direct use and the other charge will be for the overhead cost distribution based on actual use of the particular resource. Billing for the use of non fixed capacity resources such as forms usage will be accomplished by a single charge based on the unit price.

## Budgeting Available Computing Resources

For possibly a shorter period of time than the needs appraisal planning horizon, those responsible for the project control function will negotiate budgets with operating centers. This budget allocation will depend on the subjective interest of the firm and the computing needs expressed by the operating centers. Using the schedule of resource units and prices, and usage history relating to resource mix requirements, a dollar budget allocation is made.

As implied previously, the aggregate dollar allocation will not exceed total cost which is analogous to over-all capacity. An operating center may, however, exhaust its budget. Such an operating center may be allocated a budget increase from either a budget reserve or under-used outstanding budgets; if a larger budget cannot be arranged, the operating center may be denied access to the facility for the remainder of the budget period.

## Pricing Strategy

As should be apparent, the main computing resource units such as JPT, cards read, cards punched, and lines printed, are all associated with some maximum deliverable quantity of usage during a selected period of absolute time.

This maximum deliverable quantity of usage per given period of absolute time can be looked upon as the fixed capacity level associated with this type of resource.

The fundamental intent behind the pricing strategy is to pass on to the operating centers an explicit awareness in terms of real dollars for how much the maximum deliverable usage exceeds the aggregate usage for a particular fixed capacity resource. For a fixed capacity resource, periodic total dollar cost is analogous to capacity and aggregate direct dollar usage is analogous to that portion of available capacity which was actually used. Upon periodically noting the difference in total dollar cost and aggregate direct dollar usage, operating centers will be constantly mindful of computing use in relation to available capacity; they will and should be concerned about why the difference between these two dollar quantities might be widening.

Another reason for pricing by allocating cost to periodic maximum deliverable usage of fixed capacity resources is to insure that more costs are not recovered than exist. Allocating costs to other than estimated capacity runs the risk of over-recovering cost. For instance, the average cost pricing method, the pitfalls and characteristics of which were described in Chapter 1, is basically a technique to provide in advance for cost recovery. This method allocates cost to expected or anticipated usage instead of maximum deliverable usage. If it happens that actual usage is equivalent to anticipated usage, the average cost pricing method will exactly recover cost. If the estimate of actual usage is low, cost will be over-recovered and such excess recovered cost will have to be credited somehow to the operating centers. If the estimate of actual usage is high, cost will be under-recovered, in which case the additional cost will have to be distributed to the operating centers according to some basis such as actual usage.

A drawback to average cost pricing not previously discussed is that the operating centers will not be made aware of capacity in relation to direct usage; the very philosophy behind average cost pricing is to identify only the total cost absorbed by direct usage. This drawback can be eliminated by distributing total cost to estimated capacity rather than anticipated usage.

One other key notion to the pricing strategy involves the modification of unit prices. For a fixed capacity resource, unit price is the result of dividing a periodic total cost by a periodic capacity. When such periodic costs or capacities change, these unit prices will also change; in this way the users will be more aware of what relative value they are getting for their computing dollar. Unit prices for fixed capacity resources would obviously need changing upon redefining, for example, the accrual of costs to specific resource units. Unit prices for resources used in piece-wise fashion would fluctuate in relation to piece-wise costs.

Note that it is not the intent of the pricing strategy to encourage or discourage usage with price modifications that are not supported by changes in cost or capacity. Encouragement or discouragement with price changes will imply a real change in economic factors.

Priority Allocation

Priority must be allocated since all user jobs do not have the same urgency--if all user jobs did have identical urgency, the FCFS default mechanism would be sufficient to accomplish satisfactory allocation.

As suggested by Nielsen (1970), one way to acknowledge differing job urgencies is to have more than one input work queue where each queue is associated with a particular priority level. Within each queue, the order of service is FCFS.

A problem that must be dealt with is how the actual priorities will be allocated to users. One way to allocate priority is to assign a dollar price to priority levels and allow users to decide what priority is needed subject to their respective dollar budget constraints for computing. Priority is a resource but is not related to actual costs to be recovered as are fixed capacity resources and resources used in piece-wise fashion. Priority is the privilege of service preference accorded certain users over other users. To develop a dollar price for priority, a balance must be established between the private value of preferred service for a current user and the social cost of deferred service to users with lower priorities waiting for service.

Attaching dollar values to priority was deemed a dimension to pricing for future experimentation and research.

By whatever mechanism priority is to be allocated, the users should be limited to only so much priority access and should be able to select priority levels for jobs from a mix of assigned priorities. This notion implies that a determinable amount of priority is defined. To define a finite amount of priority, there will be a priority level for each job submitted over some period for which allocated priority "budgets" are applicable. As with dollar budgets for other resources, priority budgets will be established for those responsible for the project control function through subjective appraisal of project requirements in relation to the needs of the firm.

An additional aspect of priority needs to be considered. With fixed priority levels assigned to jobs, there is the possibility that certain jobs with low priority will be caused to wait inordinately long periods for service. This difficulty can be eliminated by implementing a priority level promotion scheme whereby jobs appreciate in priority according to accumulated wait time.

Note that priority considerations are not limited to CPU service and can be used to allocate for different urgency associated with other service such as printing.

## A STATISTICAL METHOD FOR EXPERIMENTATION WITH PRICING POLICY AND PRIORITY ACCESS

Box (1957) and Barnett (1960) discuss the fundamentals of a statistical technique in the area of design of experiments called Evolutionary Operation (EVOP). EVOP can be used to explore the effects of selected variables on one or more measures of effectiveness. The technique involves making relatively small changes to the variables and determining the resulting measures of effectiveness. The responses or measures of effectiveness are to be either maximized, minimized, or kept within a certain range; the responses do not have to be quantitative--they may be qualitative such as with textures and colors.

If the selected variables seem to have little effect on the responses, a change in the levels of the variables may be required or a change in the model itself may be required.

EVOP would seem to have application in exploring the resource allocation effects of a given pricing policy; for instance, it may be desired to evaluate the effect of certain prices and priorities on measures of effectiveness such as demand backlogs or waiting times to be minimized.

There may be a drawback to using the technique in that it takes time for the user community to react to changes in price or priority; for example, a factorial model with two levels and two variables would require four different combinations of say, prices, for one EVOP data point which is repeated as a requirement of the statistics--this experiment may take too long from a practical standpoint.

## USER AIDS FOR IMPROVING SERVICE

Users would attempt to schedule their job submissions during periods when it could be realistically anticipated that the job and print queues would be lightly loaded. To give the users an anticipation aid, backlog times could be collected by the system software; these cumulative backlog times could be plotted similar to the capacity allocation plot discussed in Chapter 4.

Another technique to aid users in deciding when to submit jobs would be to have a display device which would be queried for estimated queue lengths.

## SUMMARY

It cannot be concluded that such a pricing policy would benefit the firm over and above the tradeoffs associated with implementation and administration costs. Computer systems software would most likely have to be modified, a staff might be required to handle the project control function and the like.

It does seem worthwhile, however, from an academic viewpoint to explore internal pricing on a more formal basis; the economic benefits to effective computing resource allocation could offer significant advantages to the future of internal computing.

## Chapter 4

## A GRAPHICAL ANALYSIS OF USER JOB LOADING ON SINGLE STREAM COMPUTING SYSTEMS

Part three of the research problem required that a computer program be developed and experimented with to graphically depict the allocation versus time of aggregate computing resources to user jobs.

Schroeder (1971) pointed out that one of the reasons for poor utilization in contemporary private business internal computing environments was "management's failure to understand the true capacity of equipment on hand." Schroeder acknowledges the existence of effective but complex performance measuring schemes; he implies, however, that it may be better to consider simple ways to display the utilization of the facility by its users--such easy-to-contemplate displays of usage history may not obscure the awareness that managements should possess.

### SELECTION OF A UTILIZATION MEASURE

As discussed in the Theoretical Framework section of Chapter 1, aggregate computing resources are allocated to certain states along the absolute time base. Such aggregate resource residence states were suggested as follows:

1. System not in use--not manned and off, $NU_{off}$

2. System not in use--manned and on, $NU_{on}$
3. System processing a user job--job processing time JPT
4. System maintenance--scheduled, $M_s$
5. System maintenance--unscheduled, $M_u$
6. Operating overhead--$O_o$
7. System software overhead--$O_s$

It is a relatively simple task to measure periods of $NU_{off}$, JPT, $M_s$ and $M_u$. Some difficulty, however, may arise in the measurement of $NU_{on}$, $O_o$, and $O_s$. The reason for the measurement difficulty centers partially on deciding whether certain "grey areas" belong to this or that category; this problem has only been mentioned here and was not a topic to be resolved during this investigation. It should be noted, however, that $NU_{on}$, $O_s$, and $O_o$ constitute states of the aggregate system resources during which computing is not being performed that is directly inputable to specific users.

Job processing time (JPT) associated with the assumed batch SPOOLing environment is defined as the absolute time base increment absorbed by CPU servicing of a given user's job directives. It should be recognized that SPOOLed unit record input-output operations are either not in progress, or are progressing concurrent with CPU job processing, or are progressing without CPU job processing. As stated in Chapter 3, the SPOOLed overlap of CPU job processing is with unit record input-output operations for other than the current job.

The writer selected JPT as a simple-to-measure quantity representing the allocation of aggregate system resources to user jobs. JPT is a representative quantity in that the CPU can be viewed as the most critical computing resource to system operation--the CPU must first be accessed prior to gaining access to any of the other system resources.

## DETERMINATION OF VALUES FOR THE UTILIZATION MEASURE

It is not uncommon for system supervisory software supplied either by computer manufacturers or software development firms to have the capability to capture and accumulate various utilization measures among which are the time when the first job directive is beginning to be processed and the time subsequent to processing of the end-of-job directive; these two times have been referred to previously in Chapter 1 as job initiate and job terminate times.

The supervisory software for the environment upon which the programming and experimentation was based had the capability to capture the required data. Periodically (daily), accumulated job-wise facility usage data (e.g., job initiate time, JPT, cards read, and pages printed) was written on magnetic tape storage for subsequent analysis; this usage data was identified or associated with job numbers within specific projects.

# DESCRIPTION OF THE BASIC CONCEPT OF THE GRAPHICAL ANALYSIS

It is desired to be able to identify JPT periods for one or more independently operated computing systems.

A given computing system with a sequenced set of either job initiate and job terminate times or job initiate times and JPT values can be thought of as either in or not in the state of processing a user job. At any point in absolute time, an integral number of computing systems may be in the JPT state. Hence, it is desired to maintain a cumulative balance of computing systems in the JPT state as a function of absolute time. It should be noted that, in this work, the nearest second was the discrete resolution for specifying absolute time.

The displaying of the cumulative JPT state at instantaneous discrete points in absolute time may provide a plot which is too detailed for the observer. Such a plot would show constant JPT levels with points at which the level "instantaneously" changed due to the initiation and termination of jobs; the envelope of this detail plot has the shape of a discontinuous step function.

If it decided to average JPT level over some arbitrary plotting interval, the area under the instantaneous envelope must be determined and subsequently divided by the plotting interval. The desired effect, in this instance, would be to "net-out" a longer more detailed plot.

## THE ENVIRONMENT AND AVAILABLE DATA

The environment used in this research consisted of seven independently operated and batch SPOOLed UNIVAC 1100 series computing systems. Two of the systems were UNIVAC 1107 computers and were used almost exclusively for business and management applications. The remaining five systems were UNIVAC 1108 computers and were dedicated primarily to technical and scientific applications.

Each of the systems collected job utilization data on a daily basis which included the following data items:

1. user organization
2. project number
3. submitter identification
4. job number
5. computing system identification
6. date
7. job initiate time
8. JPT
9. magnetic tape units required
10. cards read
11. cards punched
12. plot images made
13. memory required
14. pages printed

All daily utilization history was merged together onto a single magnetic tape file. The number of daily jobs encountered on all seven systems during the period of experimentation included several days in the period from April 1970 through June 1970.

## DISCUSSION OF THE PROGRAMMING

The programming discussed in this section was done using the FORTRAN V programming language available for UNIVAC 1100 series computing systems. The actual code consists of one main or calling program and five subroutine subprograms. A copy of this code has been included in the thesis as Appendix A.

The emphasis in this section is directed toward a description of the main programming requirements. Other more or less secondary programming requirements (e.g., error checking and specialized print-outs) will be mentioned but not given any detail discussion. First, the main programming requirements to be satisfied will be presented. Second, a brief statement will be given as to how the actual code satisfying the requirements was organized.

The first requirement entailed the extraction and building of data lists of job related utilization measures. Accordingly, job related utilization data on magnetic tape storage had to be placed in FORTRAN discernible data lists. The main data lists were the computing system associated job initiate times and JPT values; for each job associated with its particular computing system was an initiate time given in seconds and a corresponding JPT value also given in seconds.

Once the building of the initiate time and JPT value lists had been accomplished, a requirement then existed to establish a sequence for these lists; it could not be assumed that the extracted magnetic tape entries were in any special order. Therefore, it was necessary for these data lists to be sorted in initiate time sequence for each computing system.

The main requirement involved a quasi simulation of the operation of the seven computing systems. The quasi nature of this simulation arose from the fact that the event structure was defined using actual job processing history. In a more common variety of digital simulation, the idea is to sample events from appropriate probability distributions in order to predict and analyze performance.

The simulation model controlled its clock on an asynchronous basis; this meant that the simulation clock was stepped forward by an amount equal to the time difference between successive events.

An event structure for the simulation was defined in terms of job initiations and job terminations. To begin the simulation, the starting job initiate times (i.e., the first events to occur on each of the computing systems) were scheduled to occur in the future; when each of these events occurred, the corresponding job terminate times were scheduled. Using the principle of scheduling a successor event on a given computing system based on the occurrence of its associated predecessor event, the simulation was performed. For each job initiated, the cumulative JPT level was incremented by one; for each job terminated, the cumulative JPT level was decremented by one. The simulation was terminated after the asynchronous clock had reached one day or 86,400 seconds.

Resulting from the simulation was a list of cumulative JPT levels corresponding to a list of times in seconds when these JPT levels had to be updated.

If averaging of the "instantaneous" JPT levels was specified by selecting a plotting interval other than one second, an integrating and dividing step was required. As previously mentioned, this requirement was satisfied by computing areas under the JPT envelope between the selected plotting interval boundaries across the absolute time base; each such integrated segment was then divided by the plotting interval.

The final requirement involved a mapping of each interval's JPT level into discrete ordinates for actual printing of the plot values.

The secondary features of the over-all code referred to earlier include, in addition to error checking, a capability to list the job initiate and job terminate times in sequence by computing system, and a capability to list the total number of jobs processed on each computing system.

Organization of the required logic to satisfy the programming requirements was implemented as follows:

| Code | Programming requirements satisfied |
| --- | --- |
| Main Program | Called the required subroutine subprograms in the order necessary to achieve the desired results; this subroutine calling order was XTRAC, ORGTM, and PCMAC. |
| XTRAC | Performed the data extraction and list building. |
| ORGTM | Performed the sorting, simulating (i.e., using SCHED and RMV subprograms), integrating and averaging. |
| PCMAC | Performed the mapping of each interval's JPT level into discrete ordinates for the actual printing of the plot values. |
| SCHED | A code called by ORGTM to insert upcoming events into a threaded list; this code maintains the list of scheduled events in order by future occurrence times. |
| RMV | A code called by ORGTM to access the earliest upcoming simulation event from the threaded list maintained by SCHED. |

## THE PLOTTING RESULTS

Some example plots have been included in the thesis as Appendix B. Upon observing the example plots, it will be noticed that the more common plotting interval was 1800 seconds or thirty minutes; this interval was chosen since it gave 48 plot ordinates for the 24 hour day which fit nicely on one page of 11 inch long paper.

## UTILITY OF THE GRAPHICAL ANALYSIS

As stated earlier, the main reason for development of this plotting concept was to give management a better feel as to the extent to which available computing capacity is being used. Utility of this particular plot would seen to be applicable to the situation where management needs a corporate overview of several computing facilities within a particular firm.

One of the main advantages to this graphical display is that it offers a direct intuitive appreciation for the relative extent to which aggregate computing facilities are servicing peak loads of user jobs; management might have an interest in this situation where a basic concern was not to maintain capacity merely to support peak utilization.

How management actually uses the information afforded by such a plotting concept is, of course, dependent on the policy and value system of the particular firm. Notwithstanding the unique concern of management, it would appear that this approach would generate a concern for the comparison of computing levels over similar periods of time and the development of performance standards.

The writer hopes that this work will stimulate further ideas for ways that will more readily provide the awareness which management must have for how the supply of computing capacity is allocated.

## ENHANCEMENTS

Certain enhancements come to mind when viewing the plotting approach in retrospect.

For example, it would be helpful to have an average JPT level per day for each computing system and for the aggregate set of computing systems. Of course, the aggregate

JPT level per day may be easily estimated by intuitively passing an averaging line through the envelope and determining the ordinate value.

Also, a simplification of the multi-system plot could be used to examine single system utilization. As a matter of tactics, the ordinate scale would conceivably be expanded in spanning its value range from zero to one. The quasi simulation logic, here, would be rather trivial in that only one upcoming event would be required in addition to the signal for the end of the simulation.

## EXTENSIONS

The ideas associated with the subject plotting concept were related to the single stream batch SPOOLed environment characterized in Chapters 1 and 3.

In the sense of batch user job multiprogramming, this type of plot would take on different significance; in this multiprogramming environment, JPT values are in essence job residence times and, as such, the resulting plot would indicate the extent of user job multiprogramming as a function of time. To arrive at a plot analogous to the single stream batch SPOOLed case, JPT values would be measured for periods when a given system was processing one or more jobs.

Accordingly, for the multiprocessing case (i.e., where one or more slave computers are processing batch work under the control of a master computer), the subject plotting approach is readily adaptable providing the slave computers are not user job multiprogrammed.

As far as considering non batch environments such as found in time sharing application areas, the concept of a job could be redefined in the sense of some measurable continuous amount of computing associated with a given user.

## REFERENCES


Barnett, E. Harvey, "Introduction to Evolutionary Operation," Industrial and Engineering Chemistry, Vol. 52, No. 6, pp. 500-503 (June 1960).

A brief discussion of what evolutionary operation is and an example of how it is implemented.

Baumol, William J., Economic Theory and Operations Analysis, Englewood Cliffs, N. J.: Prentice-Hall, (1965).

Discusses basic mathematical and conceptual notions associated with microeconomic theory. Introduces the fundamental concepts of such specialized economic topics as general equilibrium theory, input-output analysis and welfare economics. In particular, this reference gives a formulation of the problem of maximizing a utility function subject to constraints.

Box, G. E. P., "Evolutionary Operation: A Method for Increasing Industrial Productivity," Applied Statistics, Vol. 6, No. 2, pp. 81-101 (1957).

This is the original article on the technique, evolutionary operation, which Box himself developed. This comprehensive and readable treatment of the topic is a must for those considering application of this technique.

Diamond, D. S. and L. L. Selwyn, "Considerations for Computer Utility Pricing Policies," Proceedings-23rd National ACM Conference, pp. 189-200 (1968).

Presents an excellent framework within which to consider pricing for general application. Also, relates to experience with pricing methodology at Massachusetts Institute of Technology.

English, J. Morley (ed.), Cost-Effectiveness, The Economic Evaluation of Engineered Systems, University of California Engineering and Physical Sciences Extension Series, New York: John Wiley & Sons, Inc. (1968).

A comprehensive presentation of the principles of cost-effectiveness with emphasis on the economic evaluation of large governmental systems undertakings as in aerospace. Discusses systems engineering, decision theory, value theory; presents an excellent hypothetical example relating to the supersonic transport.

Freested, Willis Carl, "Decision-making in the Computer Charge Rate/Turn-around Time Tradeoff," An unpublished paper submitted at the 1970 AFIPS Fall Joint Computer Conference (1970).

A quantitative statement of user strategy in an effort to achieve maximal project cost effectiveness.

Gordon, Robert F., "A Pricing Scheme for Multi-programming Systems," Computer Services, pp. 26-30 (November/ December 1970).

Discusses a pricing scheme requiring identical charging each time the same job is processed.

Greenberger, Martin, "The Priority Problem and Computer Time Sharing," Management Science, Vol, 12, No. 11, pp. 888-906 (July 1966).

Discusses different centrally determined strategies for automatic allocation of CPU access to competing jobs in a time sharing environment. Discussed are priority rules such as round-robin, first-come-first-served and shortest-job-next. Concludes with some brief mention of problems which pricing might overcome.

Hart, L. E., "The User's Guide to Evaluation Products," Datamation, Vol. 16, No. 17, pp. 32-35 (December 15, 1970).

Provides brief sketches of major computer performance evaluation products and services. Includes hardware monitors, software monitors, and simulation techniques.

Hootman, Joseph T., "The Pricing Dilemma," Datamation, Vol. 15, No. 8, pp. 61-66 (August 1969).

Introduces the difficulties associated with formulating a pricing policy.

Huggins, Phyllis, "University Centers Face NSF Cut Off of Research Funds," A newspaper article, Computerworld, pp. 1-2 (March 3, 1971).

Reports on some problems of outside funding.

Johnson, R. R., "Needed: A Measure for Measure," Datamation, Vol. 16, No. 17, pp. 22-30 (December 15, 1970).

Expresses the need for a theory of computer performance. Discusses what has been done in the past such as executing standard instruction mixes. Describes the techniques of simulation, monitoring and analysis.

Kanter, Herschel, Arnold Moore and Neil Singer, "The Allocation of Computer Time by University Computer Centers," The Journal of Business, Vol. 41, No. 3, pp. 375-384 (July 1968).

Discusses the resource allocating inequities imposed on university computing centers by governmental requirements for average-full-cost pricing.

Kassouf, Sheen, Normative Decision Making, Foundations of Administration Series, Englewood Cliffs, N. J.: Prentice-Hall, Inc. (1970).

Mr. Kassouf is affiliated with the School of Social Sciences at the University of California. His book introduces normative or prescriptive decision theory. Normative theory prescribes how decisions should be made given the related goals and values of the rational decision maker. This book covers topics such as decision making under certainty or uncertainty, decision making under conflict and collective decision making.

Kolence, Kenneth W., "A Software View of Measurement Tools," Datamation, Vol. 17, No. 1, pp. 32-38 (January 1, 1971).

An excellent paper on the status and techniques of computer utilization measurement using software.

Malinvaud, E., Statistical Methods of Econometrics, Vol. 6, Studies in Mathematical and Managerial Economics, Amsterdam: North-Holland Publishing Company, (Sole distributor in United States and Canada: Chicago: Rand McNally & Company), (1966).

This book was originally published by Dunod, Paris, 1964. Takes the viewpoint that the aim of econometrics is the empirical determination of economic laws. Useful to those who wish to understand the general nature of econometric methods, but who do not have the time or means for an exhaustive study of their properties. Discusses in a very readable manner topics such as linear estimation, analysis of time series, non-parametric statistics, autocorrelation, multivariate analysis and analysis of variance. Contains an excellent set of references. A must for those doing experimental work in computing economics.

Marchand, Maurice, "Priority Pricing with Application to Time-shared Computers," AFIPS Conference Proceedings, 1968 Fall Joint Computer Conference, pp. 511-519 (1968).

Introduces priority queueing as a remedy for the shortcomings associated with a first-come-first-served job processing rule. Develops a quantitative model for the conditions which priority prices must satisfy in order to be Pareto optimal.

Nielsen, Norman R., "Flexible Pricing: An Approach to Allocation of Computer Resources," AFIPS Conference Proceedings, 1968 Fall Joint Computer Conference, pp. 521-531 (1968).

A general statement of pricing policy considerations relating to the allocation of centrally-supplied computing resources to a user community. Environmentally, the article is directed toward internal job shop situations such as found in private business and in educational institutions.

_______, "The Allocation of Computer Resources--Is Pricing the Answer?," Communications of the ACM, Vol. 13, No. 8, pp. 467-474 (August 1970).

An extension and broadening of Nielsen's earlier article in 1968. One of the best references on pricing policy considerations for non profit, internal job shop computing environments. Relates to actual experience with pricing at two different computing installations of the Stanford Computation Center.

O'Sullivan, T. C., "Exploiting the Time Sharing Environment," Proceedings-22nd National ACM Conference, pp. 169-175 (August 1967).

Discusses ways in which the user can "take advantage" of billing approaches used by commercial time sharing vendors.

Sass, C. Joseph, "Benefits of Equitable Charges for a Batch Computer," Data Management, pp. 23-25 (March 1971).

Presents a billing approach to equitably recover computing costs. Discusses the inequitities of flat rate or wall clock billing. This approach bills only on the basis of hardware resources used and establishes a fixed cost per job submission. Mr. Sass is Assistant Professor of Operations Analysis in the College of Business Administration at the University of Toledo (Ohio).

Schroeder, Walter J., "The Part Time Computers," Business Automation, Vol. 18, No. 1, pp. 18-22 (January 1, 1971).

This timely article discusses the state of private business computing installations with respect to how effectively the computing dollar is used and managed. Report results of a recent study done by A. T. Kearney & Company covering 89 companies and 155 computers.

Sharpe, William F., The Economics of Computers, New York: Columbia University Press (1969).

Presents a theoretical background for computing economics. Discusses topics such as the characteristics of the computing industry, the sale and rental of computers, computing cost effectiveness, and pricing. Pricing is discussed on the basis of the value of the computing performed.

Singer, Neil, Herschel Kanter, and Arnold Moore, "Prices and the Allocation of Computer Time," AFIPS Conference Proceedings, 1968 Fall Joint Computer Conference, pp. 493-498 (1968).

An excellent article on pricing philosophy. Develops an argument for pricing in comparison to existing allocating mechanisms. Good discussion on some different variations to priority rules for user access.

Smidt, Seymour, "Flexible Pricing of Computer Services," Management Science, Vol. 14, No. 10, pp. B-581-B-600 (June 1968a).

Presents a market mechanism for allocating computer time at universities. Users specify either the desired completion time for their jobs or the maximum rate they will pay for service. Requires knowing economic supply and demand schedules. Provides a microeconomic basis for the approach. Discusses some priority considerations.

________, "The Use of Hard and Soft Money Budgets, and Prices to Limit Demand for Centralized Computer Facility," AFIPS Conference Proceedings, 1968 Fall Joint Computer Conference, pp. 499-509 (1968b).

Provides several quantitative models of theoretical economic pricing situations. Presupposes user utility functions. Addresses the longer range microeconomics under varying sets of assumptions. Offers some discussion of the general pricing situation.

Sutherland, I. E., "A Futures Market in Computer Time," Communications of the ACM, Vol. 11, No. 6, pp. 449-451 (June 1968).

This paper describes an auction (bidding) system used at Harvard University. Is based on the principle that the "currency" assigned to a project is not consumed.

Warner, C. Dudley, "Monitoring: A Key to Cost Efficiency," Datamation, Vol. 17, No. 1, pp. 40-49 (January 1, 1971).

A good paper on the use of hardware to measure the utilization of physical computing resources.

Williamson, Oliver E., "Peak-load Pricing and Optimal Capacity under Indivisibility Constraints," American Economic Review, Vol. 56, pp. 810-827 (1966).

Provides a theoretical microeconomic consideration of pricing capacity under indivisibility. Presupposes the existence of supply and demand schedules.

Woodson, Thomas T., Introduction to Engineering Design, New York: McGraw-Hill Book Company, (1966).

This text provides an excellent framework within which to perform any engineering design task. Discusses the formulation of objective (criterion) functions, goal analysis, and other practical design topics.

Wright, M. G., Discounted Cash Flow, London: McGraw-Hill Publishing Company Limited, (1967).

Some excellent practical discussion on fund allocation in relation to the corporate strategy and selection of projects.

Zangwill, Willard I., Nonlinear Programming, A Unified Approach, Engle Cliffs, N. J.: Prentice-Hall, Inc., (1969).

Relates basic quantitative concepts to practical usage in operations research, economics, and general business applications. An excellent text with an extensive bibliography; good combination reference for the Malinvaud text.

APPENDIXES

# APPENDIX A

## FORTRAN Computer Code to Display the Allocation of Computing Capacity

```
      IMPLICIT INTEGER (A-Z)
      COMMON /SET1/ORGCD(1500),PROJ(1500),BADGE(1500),PROG(1500),
     1        PROC(1500),DATE(1500),STRT(1500),ELAP(1500),TAPEM(1500),
     2        CDIN(1500),CDOUT(1500),PLTT(1500),MEM(1500),PAGES(1500)
      COMMON /SET2/KEY(4000),LOC(4000),P0,P1,P2,P3,P5,P6,P7,PX
      COMMON /SET3/KODE(9),TM(9),NX(9),JP(9),LWC,NEN,JST,NCEL
      COMMON /SET4/PLOT(4000),NP
      READ(5,1)IYR,MO,IDA,INT
    1 FORMAT(3(I2),I5)
      ITEMP=IYR*10000 + MO*100 + IDA
      CALL XTRAC(ITEMP,ITREC,ILREC)
      CALL ORGTM(INT,ILREC)
      CALL PCMAC(INT,ITEMP)
      WRITE(6,2)ILREC
    2 FORMAT(1H1,10X,15HNO. OF JOBS WAS,I5)
      END
```

```
C SUBROUTINE XTRAC EXTRACTS PHASE II ACCOUNTING DATA AND BUILDS          WFS10000
C   CORRESPONDING FORTRAN A-FORMAT (FIELDATA) AND I-FORMAT DATA ITEM     WFS10010
C   ARRAYS.  THIS SUBROUTINE USES THE MREAD SUBROUTINE TO PERFORM        WFS10020
C   NON-STANDARD FORWARD ODD PARITY READS OF 600 WORD RECORDS (30 WORDS WFS10030
C   PER LOGICAL RECORD, BLOCKED 20).  THE CONVERSION OF NON-STANDARD     WFS10040
C   INPUT BUFFER FORMAT TO STANDARD FORTRAN FORMAT IS DONE WITH THE FLD WFS10050
C   SUBROUTINE.  READING CONTROLLED BY THIS SUBROUTINE CONTINUES TILL    WFS10060
C   END OF FILE IS REACHED.  A RECORD COUNT, I, AND A LOGICAL RECORD     WFS10070
C   COUNT, NREC ARE RETURNED TO THE CALLING PROGRAM.  UNRECOVERABLE READWFS10080
C   ERRORS ARE REFLECTED IN PROG(NREC) AS 80008, RECORDS FOR WHICH THE  WFS10090
C   MREAD RETURN CODE, J, WAS NEITHER0, 1, 2, OR 3 IS REFLECTED IN      WFS10100
C   PROG(NREC) AS 70007, AND IRREGULAR LENGTH RECORDS ARE REFLECTED IN  WFS10110
C   PROG(NREC) AS 60006.  THE TIME BASE FOR DATA EXTRACTED IS ALWAYS ONEWFS10120
C   DAY SELECTED BY IDATE EQUAL YR*10000+MON*100+DAY PASSED FROM THE    WFS10130
C   CALLING PROGRAM.  LOGICAL RECORD COUNT, NREC, WILL BE USED IN THE   WFS10140
C   CALLING PROGRAM TO INDEX THE FORTRAN DATA ARRAYS.                   WFS10150
      SUBROUTINE XTRAC(IDATE,I,NREC)                                    WFS10200
      IMPLICIT INTEGER (A-Z)                                            WFS10210
      DIMENSION IREC(600)                                               WFS10240
      COMMON /SET1/ORGCD(1500),PROJ(1500),BADGE(1500),PROG(1500),       WFS10250
     1        PROC(1500),DATE(1500),STRT(1500),ELAP(1500),TAPEM(1500),  WFS10260
     2        CDIN(1500),CDOUT(1500),PLTT(1500),MEM(1500),PAGES(1500)   WFS10270
      LU=1                                                              WFS10300
      NREC=0                                                            WFS10310
      M=1                                                               WFS10320
      I=0                                                               WFS10330
      IBLOCK=600                                                        WFS10370
    6 CALL MREAD(LU,M,IREC,IBLOCK,J,L)                                  WFS10380
    5 IF(J.EQ.1)GO TO 5                                                 WFS10400
      IF(J.EQ.2)GO TO 9                                                 WFS10410
      I=I+1                                                             WFS10415
      IF(J.EQ.0)GO TO 7                                                 WFS10420
      IF(J.EQ.3)GO TO 12                                                WFS10430
      WRITE(6,10)I                                                      WFS10440
   10 FORMAT(10X,45HMREAD CODE J NOT EQUAL 0,1,2, OR 3 FOR RECORD,I5)   WFS10450
      READS=1                                                           WFS10460
      GO TO 81                                                          WFS10470
    9 WRITE(6,11)LU,I                                                   WFS10480
   11 FORMAT(10X,19HEND OF FILE ON UNIT,I2,19H** NO. OF RECORDS**,I5)   WFS10490
      RETURN                                                            WFS10500
   12 WRITE(6,13)I                                                      WFS10510
   13 FORMAT(10X,38HREAD TERM - UNRECOV READ ERR ON RECORD,I5)          WFS10520
      READS=2                                                           WFS10530
      GO TO 81                                                          WFS10540
    7 READS=0                                                           WFS10550
   81 IF(L.EQ.IBLOCK)GO TO 8                                            WFS10560
      WRITE(6,21)I,L                                                    WFS10570
   21 FORMAT(10X,8HBLOCK NO,I5,14H NOT 600---WAS,I5)                    WFS10580
      READS=3                                                           WFS10590
    8 IX=1                                                              WFS10600
      DO 20 IBM=1,20                                                    WFS10630
                  YR=FLD(0,7,IREC(IX+23))                               WFS10639
                  MON=FLD(7,5,IREC(IX+23))                              WFS10640
                  DAY=FLD(12,6,IREC(IX+23))                             WFS10650
                  KDATE=YR*10000 + MON*100 + DAY                        WFS10660
                  IF(KDATE.NE.IDATE)GO TO 25                            WFS10670
                  NREC=NREC + 1                                         WFS10674
              DATE(NREC)=KDATE                                          WFS10680
              ORGCD(NREC)=FLD(0,36,IREC(IX+24))                         WFS10690
```

```
      PROJ(NREC)=FLD(0,36,IREC(IX+26))                                  WFS10700
      BADGE(NREC)=FLD(0,36,IREC(IX+18))                                 WFS10710
      PROG(NREC)=FLD(0,36,IREC(IX+25))                                  WFS10720
      PROC(NREC)=FLD(18,6,IREC(IX+1))                                   WFS10730
      STRT(NREC)=FLD(18,18,IREC(IX+23))                                 WFS10740
      ELAP(NREC)=FLD(18,18,IREC(IX+22))                                 WFS10750
      TAPEM(NREC)=FLD(18,18,IREC(IX+20))                                WFS10760
      CDIN(NREC)=FLD(0,18,IREC(IX+19))                                  WFS10770
      CDOUT(NREC)=FLD(18,18,IREC(IX+19))                                WFS10780
      PLTT(NREC)=FLD(0,12,IREC(IX+28))                                  WFS10920
                 TEMP1=FLD(0,18,IREC(IX+27))                            WFS10930
                 TEMP2=FLD(18,18,IREC(IX+27))                           WFS10940
      MEM(NREC)=  TEMP1  +  TEMP2                                       WFS10950
      PAGES(NREC)=FLD(18,18,IREC(IX+21))                                WFS10960
      IF(READS.EQ.1)PROG(NREC)=70007                                    WFS10970
      IF(READS.EQ.2)PROG(NREC)=80008                                    WFS10980
      IF(READS.EQ.3)PROG(NREC)=60006                                    WFS10990
      IF(NREC.GE.1500)RETURN                                            WFS11021
25    IX=IX+30                                                          WFS11030
20 CONTINUE                                                             WFS11040
   GO TO 6                                                              WFS11050
   END                                                                  WFS11999
```

```
      SUBROUTINE ORGTM(INT,NREC)                                          WFS12500
      IMPLICIT INTEGER (A-Z)                                              WFS12510
      DIMENSION ETIM(4000),NMAC(4000)                                     WFS12520
      DIMENSION BUF1(21),BUF2(21)
      COMMON /SET1/ORGCD(1500),PROJ(1500),BADGE(1500),PROG(1500),         WFS12540
     1         PROC(1500),DATE(1500),STRT(1500),FLAP(1500),TAPEM(1500),   WFS12550
     2         CDIN(1500),CDOUT(1500),PLTT(1500),MEM(1500),PAGES(1500)    WFS12560
      COMMON /SET2/KEY(4000),LOC(4000),P0,P1,P2,P3,P5,P6,P7,PX            WFS12580
      COMMON /SET3/KODE(9),TM(9),NX(9),JP(9),LWC,NEN,JST,NCEL             WFS12590
      COMMON /SET4/PLOT(4000),NP                                          WFS12600
      P0=0                                                                WFS12690
      P1=0                                                                WFS12700
      P2=0                                                                WFS12710
      P3=0                                                                WFS12720
      P5=0                                                                WFS12730
      P6=0                                                                WFS12740
      P7=0                                                                WFS12750
      PX=0                                                                WFS12760
      DO 10 I=1,9                                                         WFS12770
           KODE(I)=0                                                      WFS12779
           JP(I)=0                                                        WFS12780
 10        NX(I)=0                                                        WFS12790
      NEN=0                                                               WFS12800
      LWC=0                                                               WFS12810
      JST=0                                                               WFS12820
                            NCEL=9                                        WFS12824
         IZER=1                                                           WFS12830
         IONE=1                                                           WFS12840
         ITWO=1                                                           WFS12850
         ITRE=1                                                           WFS12860
         IFIV=1                                                           WFS12870
         ISIX=1                                                           WFS12880
         ISEV=1                                                           WFS12882
         IATE=1                                                           WFS12883
      DO 30 I=1,NREC                                                      WFS12900
           IF(PROC(I).EQ.48)GO TO 31                                      WFS12910
           IF(PROC(I).EQ.49)GO TO 32                                      WFS12920
           IF(PROC(I).EQ.50)GO TO 33                                      WFS12930
           IF(PROC(I).EQ.51)GO TO 34                                      WFS12940
           IF(PROC(I).EQ.53)GO TO 35                                      WFS12950
           IF(PROC(I).EQ.54)GO TO 36                                      WFS12960
           IF(PROC(I).EQ.55)GO TO 37                                      WFS12970
      IF(PROC(I).LT.48)GO TO 199                                          WFS12980
      IF(PROC(I).GT.55)GO TO 199                                          WFS12990
      WRITE(6,201)PROC(I),DATE(I),STRT(I)                                 WFS13000
  201 FORMAT(10X,3H901,3X,A6,1X,I6,1X,I6)                                 WFS13010
      GO TO 203                                                           WFS13020
  199 WRITE(6,202)PROC(I),DATE(I),STRT(I)                                 WFS13030
  202 FORMAT(10X,3H900,3X,A6,1X,I6,1X,I6)                                 WFS13040
  203          P=8                                                        WFS13050
               PX=PX+1                                                    WFS13060
               GO TO 38                                                   WFS13070
 31        P=1                                                            WFS13080
               P0=P0+1                                                    WFS13090
               GO TO 38                                                   WFS13100
 32            P=2                                                        WFS13110
               P1=P1+1                                                    WFS13120
               GO TO 38                                                   WFS13130
 33            P=3                                                        WFS13140
```

```
            P2=P2+1                                                     WFS13150
            GO TO 38                                                    WFS13160
 34      P=4                                                            WFS13170
         P3=P3+1                                                        WFS13180
         GO TO 38                                                       WFS13190
 35      P=5                                                            WFS13200
         P5=P5+1                                                        WFS13210
         GO TO 38                                                       WFS13220
 36      P=6                                                            WFS13230
         P6=P6+1                                                        WFS13240
         GO TO 38                                                       WFS13250
 37      P=7                                                            WFS13260
         P7=P7+1                                                        WFS13270
 38      KEY(I)=P*100000                                                WFS13280
         KEY(I)=KEY(I) + STRT(I)                                        WFS13290
 30 LOC(I)=I                                                            WFS13300
    Z1=P0+P1                                                            WFS13310
    Z2=Z1+P2                                                            WFS13320
    Z3=Z2+P3                                                            WFS13330
    Z4=Z3+P5                                                            WFS13340
    Z5=Z4+P6                                                            WFS13350
    Z6=Z5+P7                                                            WFS13360
    Z7=Z6+PX                                                            WFS13370
    IF(Z7.EQ.NREC)GO TO 204                                             WFS13380
    WRITE(6,106)                                                        WFS13390
106 FORMAT(10X,3H902)                                                   WFS13400
204 IHIT=0                                                              WFS13450
    M=NREC-1                                                            WFS13460
    LIM=NREC                                                            WFS13470
    DO 20 J=1,M                                                         WFS13480
       IF(IHIT.EQ.1)GO TO 21                                            WFS13490
       IHIT=1                                                           WFS13500
       LIM=LIM-1                                                        WFS13510
     DO 20 I=1,LIM                                                      WFS13520
        IF(KEY(I+1).GE.KEY(I))GO TO 20                                  WFS13530
        TEMP=KEY(I+1)                                                   WFS13540
        KEY(I+1)=KEY(I)                                                 WFS13550
        KEY(I)=TEMP                                                     WFS13560
        TEMP=LOC(I+1)                                                   WFS13570
        LOC(I+1)=LOC(I)                                                 WFS13580
        LOC(I)=TEMP                                                     WFS13590
        IHIT=0                                                          WFS13600
 20 CONTINUE                                                            WFS13610
 21 IF(P0.LT.1)GO TO 50                                                 WFS13650
       K=LOC(1)                                                         WFS13660
       CALL SCHED(101,STRT(K))                                          WFS13670
 50 IF(P1.LT.1)GO TO 51                                                 WFS13680
       K=LOC(P0+1)                                                      WFS13690
       CALL SCHED(102,STRT(K))                                          WFS13700
 51 IF(P2.LT.1)GO TO 52                                                 WFS13710
       K=LOC(Z1+1)                                                      WFS13720
       CALL SCHED(103,STRT(K))                                          WFS13730
 52 IF(P3.LT.1)GO TO 53                                                 WFS13740
       K=LOC(Z2+1)                                                      WFS13750
       CALL SCHED(104,STRT(K))                                          WFS13760
 53 IF(P5.LT.1)GO TO 54                                                 WFS13770
       K=LOC(Z3+1)                                                      WFS13780
       CALL SCHED(105,STRT(K))                                          WFS13790
 54 IF(P6.LT.1)GO TO 55                                                 WFS13800
```

```
      K=LOC(Z4+1)                                                       WFS13810
      CALL SCHED(106,STRT(K))                                           WFS13820
   55 IF(P7.LT.1)GO TO 56                                               WFS13830
      K=LOC(Z5+1)                                                       WFS13840
      CALL SCHED(107,STRT(K))                                           WFS13850
   56 CALL SCHED(300,86400)                                             WFS13860
        ETIM(1)=0                                                       WFS13870
        NMAC(1)=0                                                       WFS13880
        MAC=0                                                           WFS13882
        L=1                                                             WFS13883
      WRITE(6,301)                                                      WFS13884
  301 FORMAT(10X,9HPRE-SCHED)                                           WFS13885
      WRITE(6,718)P0,P1,P2,P3,P5,P6,P7,PX,NRFC
  718 FORMAT(5X,9(I6))
      WRITE(6,738)Z1,Z2,Z3,Z4,Z5,Z6,Z7
  738 FORMAT(1H ,7(I6))
      WRITE(6,680)
  680 FORMAT(1H1,11HPROCESSOR 0///)
      PRES=0
      TPROC=P0
      KK=0
      DO 700 II=1,8
              DO 701 JJ=1,TPROC
                   INS=LOC(PRES+JJ)
                   KK=KK +  1
                   BUF1(KK)=STRT(INS)
                   BUF2(KK)=STRT(INS) +  ELAP(INS)
                 IF(KK.LT.21)GO TO 701
                      WRITE(6,702)BUF1
  702                 FORMAT(1H ,21(I6))
                      WRITE(6,703)BUF2
  703                 FORMAT(1H ,21(I6)//)
                      KK=0
  701         CONTINUE
      IF(KK.EQ.0)GO TO 993
         WRITE(6,702)(BUF1(MM),MM=1,KK)
         WRITE(6,703)(BUF2(MM),MM=1,KK)
      KK=0
  993    IF(II.NE.1)GO TO 704
         TPROC=P1
         PRES=P0
         WRITE(6,681)
  681    FORMAT(1H1,11HPROCESSOR 1///)
  704    IF(II.NE.2)GO TO 705
         TPROC=P2
         PRES=Z1
         WRITE(6,682)
  682    FORMAT(1H1,11HPROCESSOR 2///)
  705    IF(II.NE.3)GO TO 706
         TPROC=P3
         PRES=Z2
         WRITE(6,683)
  683    FORMAT(1H1,11HPROCESSOR 3///)
  706    IF(II.NE.4)GO TO 707
         TPROC=P5
         PRES=Z3
         WRITE(6,684)
  684    FORMAT(1H1,11HPROCESSOR 5///)
  707    IF(II.NE.5)GO TO 708
```

```
          TPROC=P6
          PRES=Z4
          WRITE(6,685)
685       FORMAT(1H1,11HPROCESSOR 6///)
708     IF(II.NE.6)GO TO 709
          TPROC=P7
          PRES=Z5
          WRITE(6,686)
686       FORMAT(1H1,11HPROCESSOR 7///)
709     IF(II.NE.7)GO TO 700
      IF(PX.EQ.0)GO TO 100
          TPROC=PX
          PRES=Z6
          WRITE(6,687)
687       FORMAT(1H1,11HPROCESSOR X///)
700   CONTINUE
100   L=L+1                                                             WFS13890
      CALL RMV(KOD,ITM)                                                 WFS13910
      NMAC(L)=MAC                                                       WFS13920
      ETIM(L)=ITM                                                       WFS13930
                  IF(ETIM(L).GE.ETIM(L-1))GO TO 750
                  WRITE(6,717)ETIM(L),ETIM(L-1),L,KOD,MAC
717               FORMAT(20X,5(I6))
750       IF(MAC.LT.8)GO TO 780
               WRITE(6,719)MAC,ITM,L
719            FORMAT(15X,3(I6))
780   IF(KOD.GT.200)GO TO 79
          MAC=MAC+1                                                     WFS13950
          IND=KOD-100                                                   WFS13960
          GO TO (70,71,72,73,74,75,76,77),IND                           WFS13970
 70      INS=IZER                                                       WFS13980
         IZER=IZER+1                                                    WFS13990
         GO TO 78                                                       WFS14000
 71      INS=P0+IONE                                                    WFS14010
         IONE=IONE+1                                                    WFS14020
         GO TO 78                                                       WFS14030
 72      INS=Z1+ITWO                                                    WFS14040
         ITWO=ITWO+1                                                    WFS14050
         GO TO 78                                                       WFS14060
 73      INS=Z2+ITRE                                                    WFS14070
         ITRE=ITRE+1                                                    WFS14080
         GO TO 78                                                       WFS14090
 74      INS=Z3+IFIV                                                    WFS14100
         IFIV=IFIV+1                                                    WFS14110
         GO TO 78                                                       WFS14120
 75      INS=Z4+ISIX                                                    WFS14130
         ISIX=ISIX+1                                                    WFS14140
         GO TO 78                                                       WFS14150
 76      INS=Z5+ISEV                                                    WFS14160
         ISEV=ISEV+1                                                    WFS14170
         GO TO 78                                                       WFS14180
 77      INS=Z6+IATE                                                    WFS14190
         IATE=IATE+1                                                    WFS14200
 78   INX=LOC(INS)                                                      WFS14210
      TIME=ITM+ELAP(INX)                                                WFS14220
                  IF(TIME.GE.ITM)GO TO 533
                  WRITE(6,716)IND,TIME,ITM,L,MAC,ELAP(INX)
716               FORMAT(40X,3H666,6(I6))
                  TIME=ITM
```

```
535 CALL SCHED(200+IND,TIME)                                         WFS14230
    GO TO 100                                                        WFS14240
 79 IF(KOD.GT.299)GO TO 80                                           WFS14250
        MAC=MAC-1                                                    WFS14260
    IND=KOD-200                                                      WFS14270
    GO TO  (81,82,83,84,85,86,87,88),IND                             WFS14280
 81    IF(IZER.GT.P0)GO TO 100                                       WFS14290
       INS=IZER                                                      WFS14291
       GO TO 89                                                      WFS14300
 82    IF(IONE.GT.P1)GO TO 100                                       WFS14310
       INS=P0  +  IONE                                               WFS14311
       GO TO 89                                                      WFS14320
 83    IF(ITWO.GT.P2)GO TO 100                                       WFS14330
       INS=Z1+ITWO                                                   WFS14331
       GO TO 89                                                      WFS14340
 84    IF(ITRE.GT.P3)GO TO 100                                       WFS14350
       INS=Z2+ITRE                                                   WFS14351
       GO TO 89                                                      WFS14360
 85    IF(IFIV.GT.P5)GO TO 100                                       WFS14370
       INS=Z3+IFIV                                                   WFS14371
       GO TO 89                                                      WFS14380
 86    IF(ISIX.GT.P6)GO TO 100                                       WFS14390
       INS=Z4+ISIX                                                   WFS14391
       GO TO 89                                                      WFS14400
 87    IF(ISEV.GT.P7)GO TO 100                                       WFS14410
       INS=Z5+ISEV                                                   WFS14411
       GO TO 89                                                      WFS14420
 88    IF(IATE.GT.PX)GO TO 100                                       WFS14430
       INS=Z6+IATE                                                   WFS14431
 89  INX=LOC(INS)                                                    WFS14440
     TIME=STRT(INX)                                                  WFS14450
                 IF(TIME.GE.ITM)GO TO 692
                 WRITE(6,715)IND,TIME,ITM,L,MAC
715              FORMAT(50X,3H777,5(I6))
              TIME=ITM
692  CALL SCHED(100+IND,TIME)
    GO TO 100                                                        WFS14470
 80 WRITE(6,150)L                                                    WFS14500
150 FORMAT(10X,27HNO OF TRANSITION POINTS WAS,I5)                    WFS14510
    WRITE(6,854)IZER,IONE,ITWO,ITRE,IFIV,ISIX,ISEV,IATE
854 FORMAT(1H ,3H555,8(I6))
    IS=INT                                                           WFS14590
       K=0                                                           WFS14600
    GRO=7*INT
       CUM=0                                                         WFS14610
      DO 99 N=2,L                                                    WFS14620
          IF(ETIM(N).GT.IS)GO TO 170                                 WFS14630
          IF(ETIM(N).LT.IS)GO TO 171                                 WFS14640
601       K   =   K   +   1                                          WFS14645
           PLOT(K)=CUM+  (ETIM(N) - ETIM(N-1)) * NMAC(N)             WFS14650
    IF(PLOT(K).LE.GRO)GO TO 812
        WRITE(6,723)PLOT(K),N,ETIM(N),ETIM(N-1),NMAC(N),NMAC(N-1),IS
723     FORMAT(1H ,3H851,7(I6))
812       CUM=0
          IS=IS+INT                                                  WFS14680
          GO TO 99                                                   WFS14690
170        K = K  +  1                                               WFS14695
          PLOT(K)=CUM+(IS - ETIM(N-1))   *  NMAC(N)                  WFS14700
    IF(PLOT(K).LE.GRO)GO TO 600
```

```
      WRITE(6,721)PLOT(K),N,ETIM(N),ETIM(N-1),NMAC(N),NMAC(N-1),IS
721   FORMAT(1H ,3H849,7(I6))
600     IDEL=ETIM(N)   -   IS                                        WFS14702
        IF(IDEL.NE.INT)GO TO 602                                     WFS14706
              IS = IS  +  INT                                        WFS14707
              CUM=0                                                  WFS14709
              GO TO 601                                              WFS14712
602     IF(IDEL.GT.INT)GO TO 603                                     WFS14715
                CUM=(ETIM(N)   -  IS)   *   NMAC(N)                  WFS14718
                IS   =   IS   +   INT                                WFS14720
                GO TO 99                                             WFS14725
603     K    =    K    +    1                                        WFS14728
        PLOT(K)=NMAC(N)*INT                                          WFS14731
  IF(PLOT(K).LE.GRO)GO TO 813
      WRITE(6,722)PLOT(K),N,ETIM(N),ETIM(N-1),NMAC(N),NMAC(N-1),IS
722   FORMAT(1H ,3H850,7(I6))
813     IS   =   IS   +   INT
        GO TO 600                                                    WFS14738
171     CUM=CUM+  (ETIM(N)   -   FTIM(N-1))   *   NMAC(N)            WFS14750
 99 CONTINUE                                                         WFS14760
    NP=K                                                             WFS14770
    RETURN                                                           WFS14780
    END                                                              WFS14850
```

```
      SUBROUTINE PCMAC(INC,IDAT)                                        WFS1500
      INTEGER PLOT,BUF,AST                                              WFS1501
      DIMENSION BUF(17),AST(7),L(13),ISUB(2),LINE(14)                   WFS1503
      COMMON /SET4/PLOT(4000),NP                                        WFS1504
      EQUIVALENCE (BUF(4),LINE(1))                                      WFS1505
      DATA (AST(I),I=1,7)/6H      ,6H*     ,6H**    ,6H***   ,6H****  , WFS1506
     16H***** ,6H******                                                 WFS1507
      DATA (L(I),I=1,13)/0,83,167,250,333,417,500,583,667,750,833,917,  WFS1508
     11000/BUF(1),BUF(2),BUF(3)/6H      ,6H      ,6H     I/             WFS1509
      R1=IDAT                                                           WFS1511
      R2=R1/10000.                                                      WFS1512
      IY=R2                                                             WFS1513
      R3=IY                                                             WFS1514
      R4=R2-R3                                                          WFS1515
      R5=R4*100.                                                        WFS1516
      IM=R5                                                             WFS1517
      R6=IM                                                             WFS1518
      R7=R5-R6                                                          WFS1519
      R8=R7*100.  + .1                                                  WFS1519
      ID=R8                                                             WFS1519
      WRITE(6,10)IM,ID,IY,INC                                           WFS1521
   10 FORMAT(1H1,17X,51HCUMULATIVE UNIPROCESSOR ACTIVITY PLOT  ----   DATWFS1522
     1E  ,I2,1H-,I2,1H-,I2,18H  TIME INCREMENT  ,I5,8H SECONDS///)      WFS1523
      WRITE(6,11)                                                       WFS1524
   11 FORMAT(18X,1H0,11X,1H1,11X,1H2,11X,1H3,11X,1H4,11X,1H5,11X,1H6,11XWFS1525
     11H7)                                                              WFS1526
      WRITE(6,12)                                                       WFS1527
   12 FORMAT(18X,86H----------------------------------------------------WFS1528
     1----------------------------------)                               WFS1529
      DO 30 N=1,NP                                                      WFS1533
      R1=PLOT(N)                                                        WFS1534
      R2=INC                                                            WFS1535
      R3=R1/R2                                                          WFS1536
      I4=R3                                                             WFS1537
      R4=I4                                                             WFS1538
      R5=R3-R4                                                          WFS1539
      I5=R5*1000                                                        WFS1539
      IF(I4.NE.7)GO TO 350                                              WFS1540
  355         J=0                                                       WFS1540
              I4=6                                                      WFS1540
              ISUB(1)=AST(7)                                            WFS1540
              ISUB(2)=AST(7)                                            WFS1540
              GO TO 40                                                  WFS1540
  350 IF(I4.LT.7)GO TO 353                                              WFS1540
              WRITE(6,351)PLOT(N),N,I4                                  WFS1540
  351         FORMAT(10X,3H909,I7,I7,I7)                                WFS1540
                   GO TO 355                                            WFS1541
  353 DO 6 I=1,13                                                       WFS1541
           IF(I5.GT.L(I))GO TO 6                                        WFS1542
           GO TO 5                                                      WFS1543
    6 CONTINUE                                                          WFS1544
    5 IF(I.LE.7)GO TO 7                                                 WFS1545
          I=I-6                                                         WFS1545
          ISUB(1)=AST(7)                                                WFS1546
          ISUB(2)=AST(I)                                                WFS1547
          GO TO 8                                                       WFS1548
    7     ISUB(2)=AST(1)                                                WFS1550
          ISUB(1)=AST(I)                                                WFS1551
    8 J=6-14                                                            WFS1552
```

```
40    K=0                                          WFS15530
   IF(I4.EQ.0)GO TO 20                             WFS15540
   DO 15 I=1,I4                                    WFS15550
        K    =    K    +    1                      WFS15559
        LINE(K)=AST(7)                             WFS15560
        K    =    K    +    1                      WFS15579
15      LINE(K)=AST(7)                             WFS15581
20 K    =    K    +    1                           WFS15590
   LINE(K)=ISUB(1)                                 WFS15600
   K    =    K    +    1                           WFS15610
   LINE(K)=ISUB(2)                                 WFS15620
   IF(J.EQ.0)GO TO 21                              WFS15630
   DO 19 I=1,J                                     WFS15640
            K    =    K    +    1                  WFS15650
        LINE(K)=AST(1)                             WFS15660
            K    =    K    +    1                  WFS15670
19      LINE(K)=AST(1)                             WFS15680
21 WRITE(6,25)BUF                                  WFS15690
25 FORMAT(1H ,17(A6))                              WFS15700
30 CONTINUE                                        WFS15710
   RETURN                                          WFS15810
   END                                             WFS15990
```

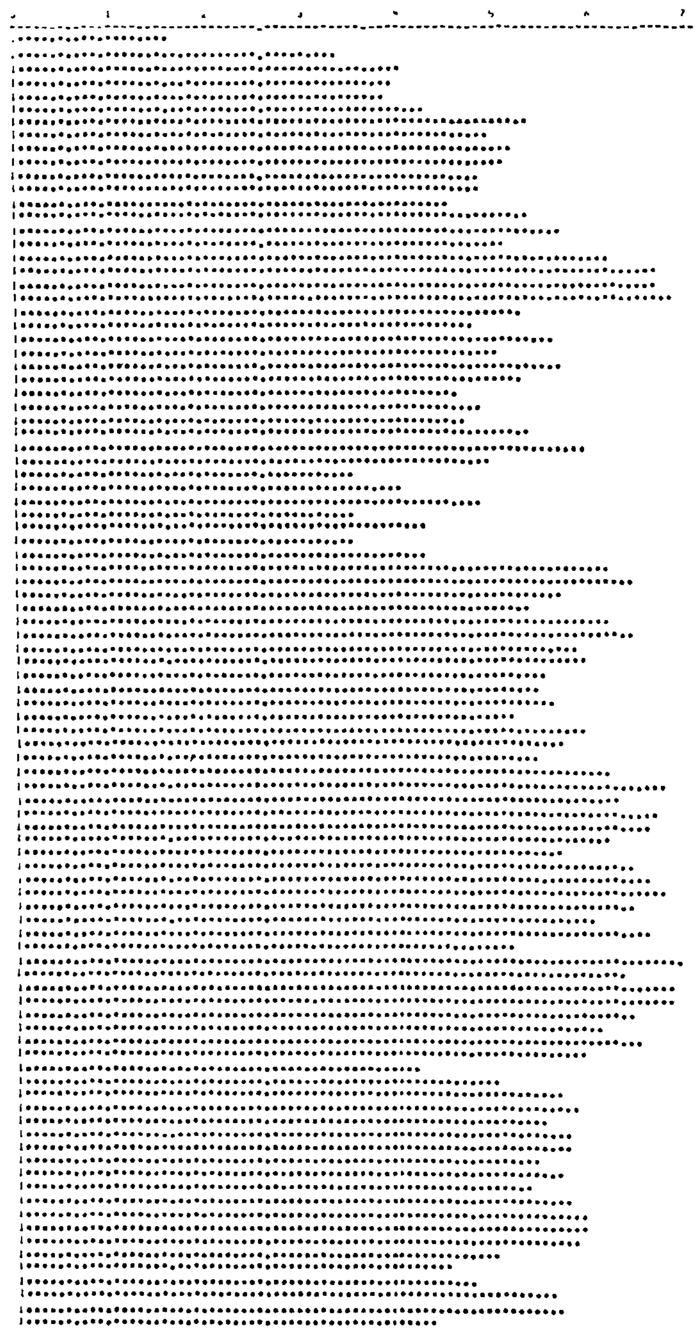
```
      SUBROUTINE SCHED(INCOD,TMIN)                                      WFS16000
      IMPLICIT INTEGER (A-Z)                                            WFS16010
      COMMON /SET3/KODE(9),TM(9),NX(9),JP(9),LWC,NEN,JST,NCFL           WFS16030
      NTRY=1                                                            WFS16040
      J=LWC+1                                                           WFS16050
    9 IF(J.LE.NCEL)GO TO 10                                             WFS16060
         J=1                                                            WFS16070
   10 IF(KODE(J).EQ.0)GO TO 11                                          WFS16080
         IF(NTRY.LT.NCEL)GO TO 6                                        WFS16090
            WRITE(6,100)                                                WFS16100
  100       FORMAT(10X,3H800)                                           WFS16110
            RETURN                                                      WFS16120
    6 NTRY=NTRY+1                                                       WFS16130
      J=J+1                                                             WFS16140
      GO TO 9                                                           WFS16150
   11 TM(J)=TMIN                                                        WFS16160
      KODE(J)=INCOD                                                     WFS16170
      IF(NEN.EQ.0)GO TO 12                                              WFS16180
      IF(TMIN.GE.TM(NEN))GO TO 13                                       WFS16190
      IF (JP(NEN).EQ.0)GO TO 18                                         WFS16200
      K=JP(NEN)                                                         WFS16210
   15 IF(TMIN.GE.TM(K))GO TO 21                                         WFS16220
      IF(K.EQ.JST)GO TO 16                                              WFS16230
      K=JP(K)                                                           WFS16240
      GO TO 15                                                          WFS16250
   16 NX(J)=K                                                           WFS16260
      JP(K)=J                                                           WFS16270
      GO TO 20                                                          WFS16280
   12 JST=J                                                             WFS16290
   17 NEN=J                                                             WFS16300
   19 LWC=J                                                             WFS16310
      RETURN                                                            WFS16320
   13 NX(NEN)=J                                                         WFS16330
      JP(J)=NEN                                                         WFS16340
      GO TO 17                                                          WFS16350
   18 JP(NEN)=J                                                         WFS16360
      NX(J)=NEN                                                         WFS16370
   20 JST=J                                                             WFS16380
      GO TO 19                                                          WFS16390
   21 JP(J)=K                                                           WFS16400
      L=NX(K)                                                           WFS16410
      JP(L)=J                                                           WFS16420
      NX(J)=NX(K)                                                       WFS16430
      NX(K)=J                                                           WFS16440
      LWC=J                                                             WFS16450
      RETURN                                                            WFS16460
      END                                                               WFS16500
```

```
      SUBROUTINE RMV(NCODE,TIME)                                          WFS1680
      IMPLICIT INTEGER (A-Z)                                              WFS1681
      COMMON /SET3/KODE(9),TM(9),NX(9),JP(9),LWC,NEN,JST,NCEL             WFS1682
      IF(JST.NE.0)GO TO 2                                                 WFS1683
              WRITE(6,1)                                                  WFS1683
    1         FORMAT(10X,3H801)                                           WFS1683
      RETURN                                                              WFS1683
    2 NCODE=KODE(JST)                                                     WFS1684
      TIME=TM(JST)                                                        WFS1685
      JX=JST                                                              WFS1686
      JST=NX(JST)                                                         WFS1687
      JP(JST)=0                                                           WFS1688
      NX(JX)=0                                                            WFS1689
      KODE(JX)=0                                                          WFS1690
      LWC=JX-1                                                            WFS1690
      RETURN                                                              WFS1691
      END                                                                 WFS1695
```

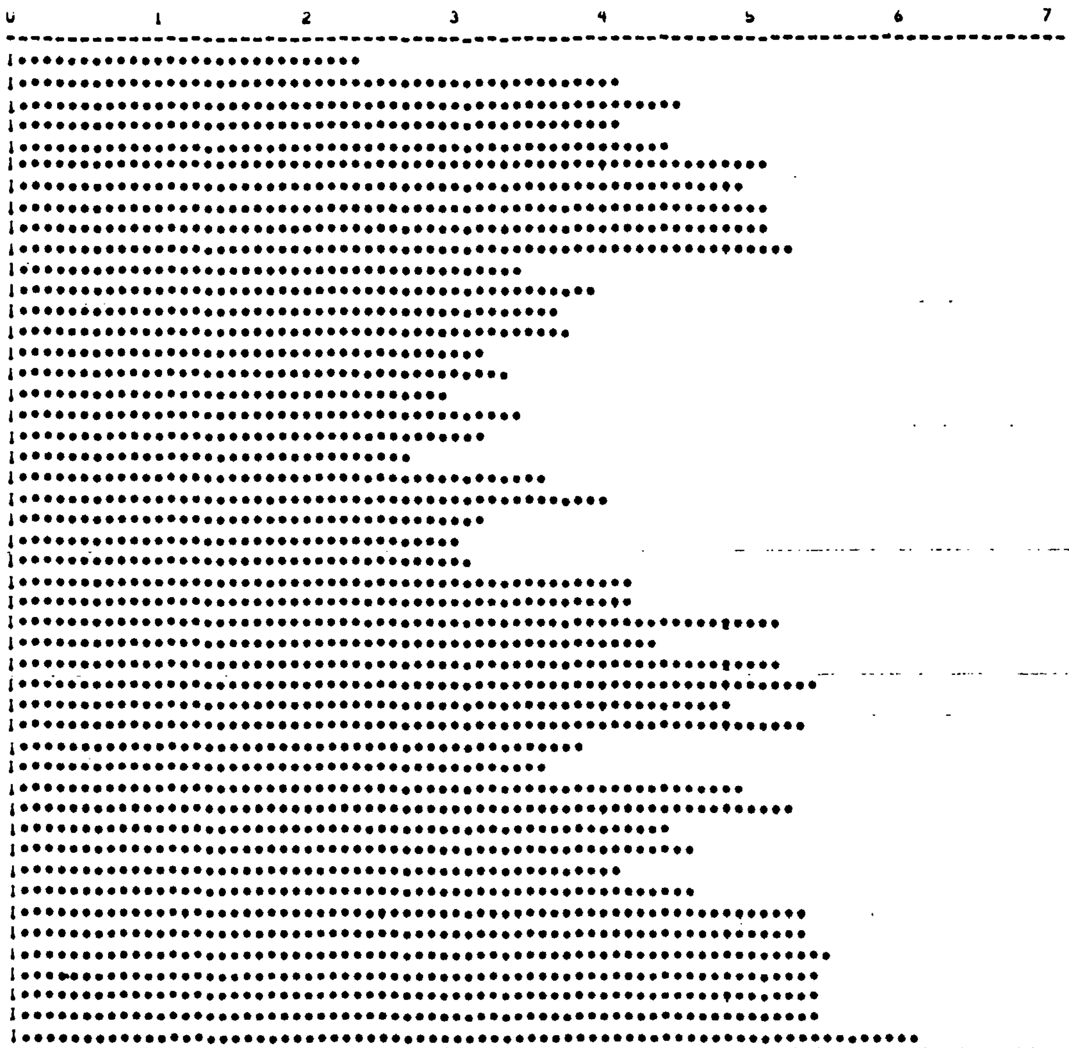
# APPENDIX B

## Example Results Displaying the Allocation of Computing Capacity

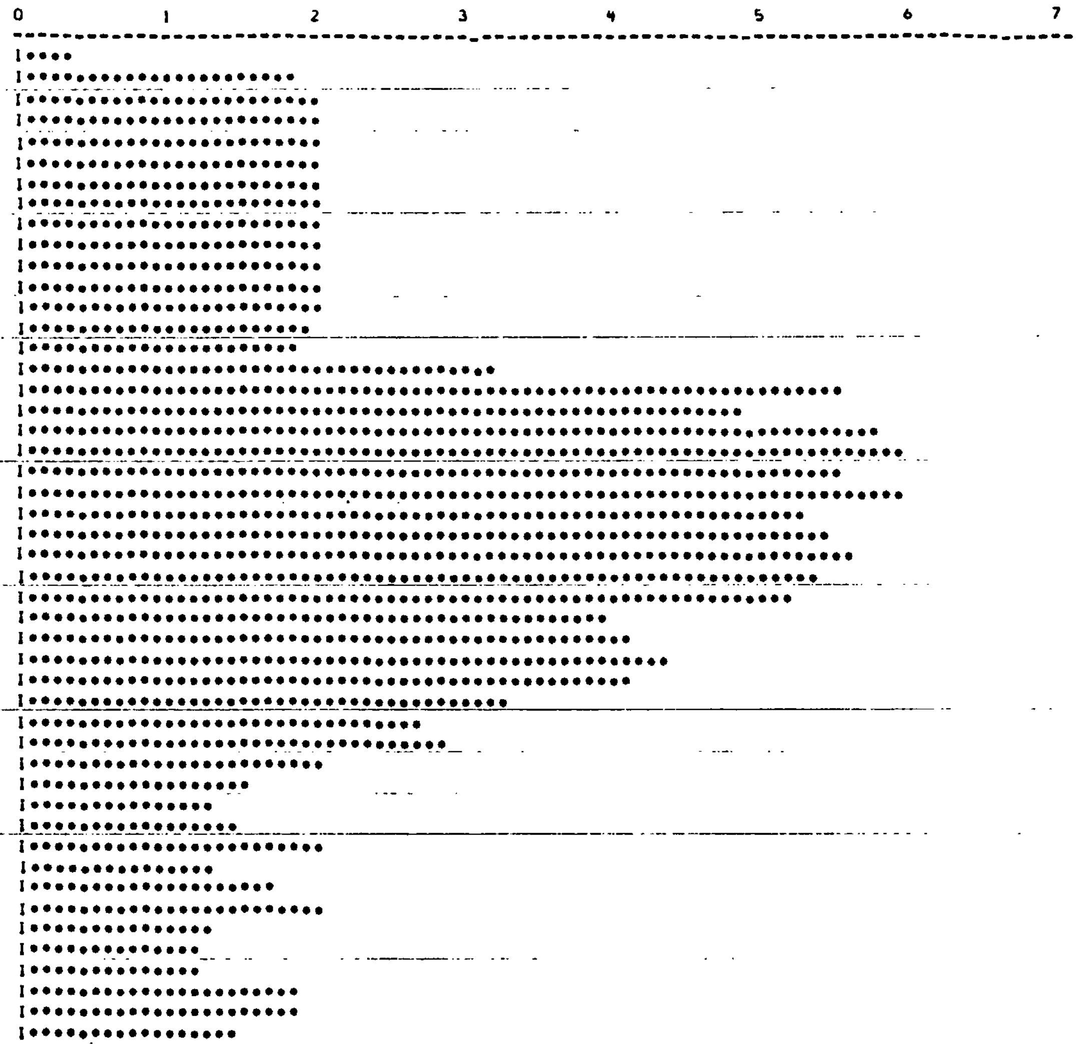
CUMULATIVE UNIPROCESSOR ACTIVITY PLOT ---- DATE 5-20-75 TIME INCREMENT 900 SECONDS



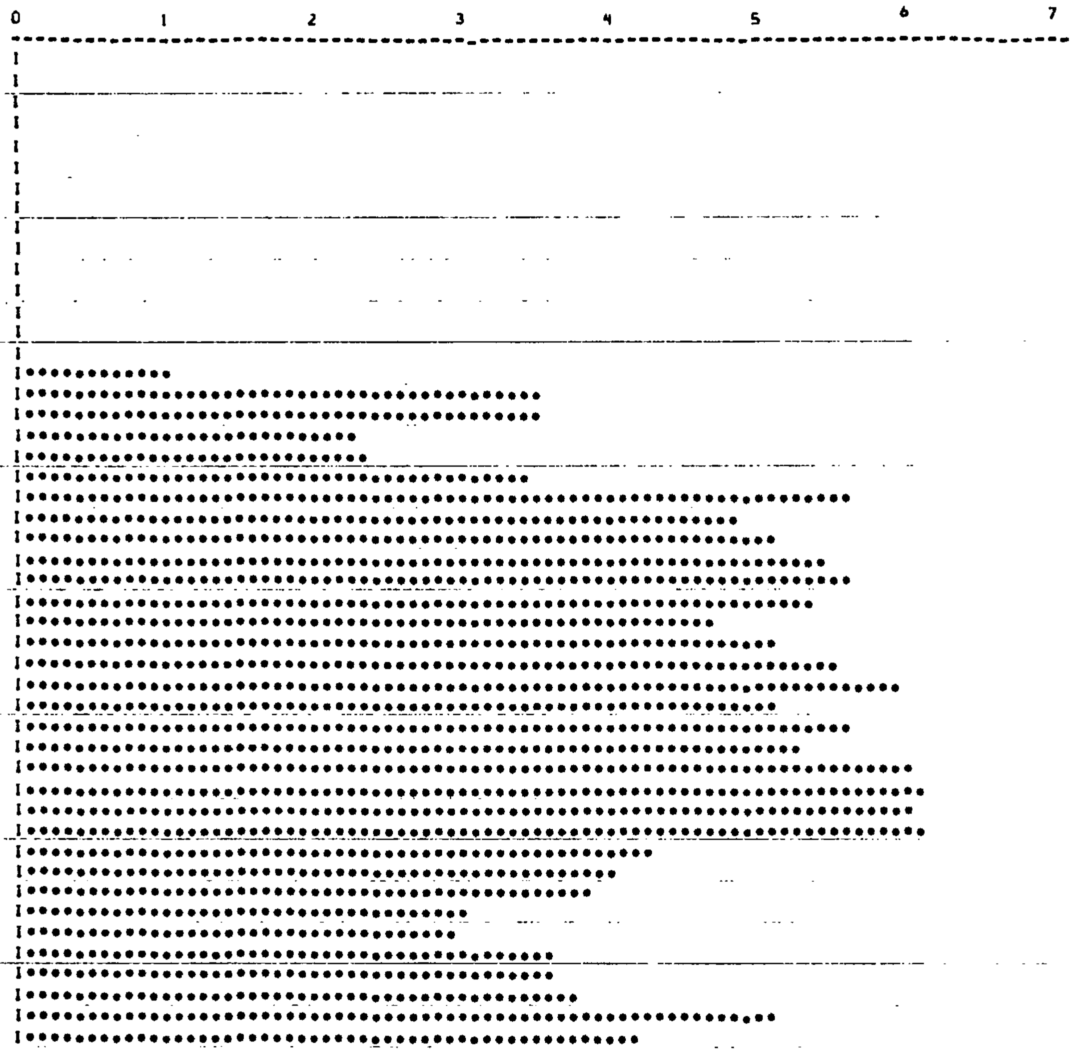
CUMULATIVE UNIPROCESSOR ACTIVITY PLOT ---- DATE 4-14-70 TIME INCREMENT 1800 SECONDS



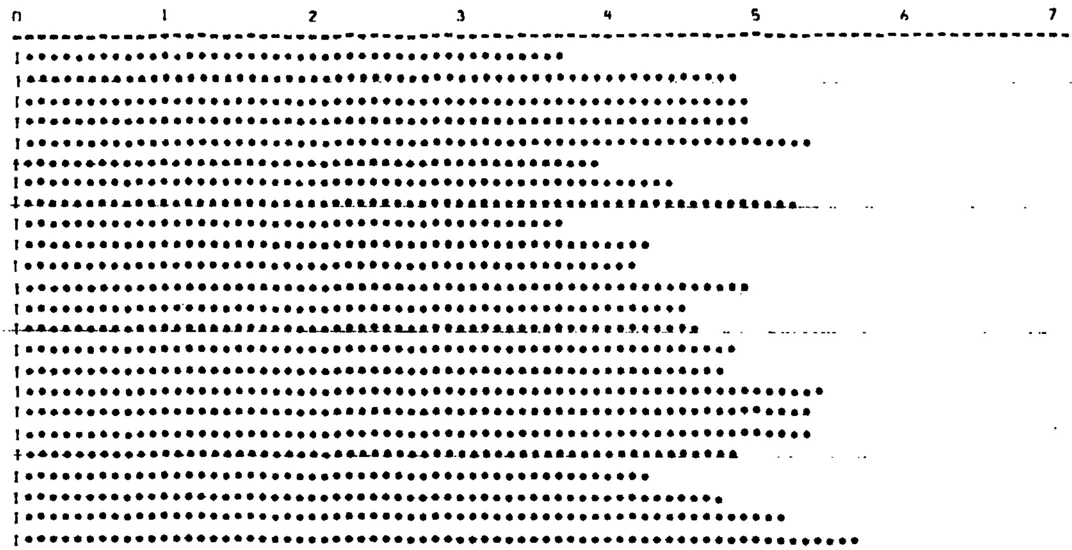
CUMULATIVE UNIPROCESSOR ACTIVITY PLOT ---- DATE 6-14-70 TIME INCREMENT 1800 SECONDS



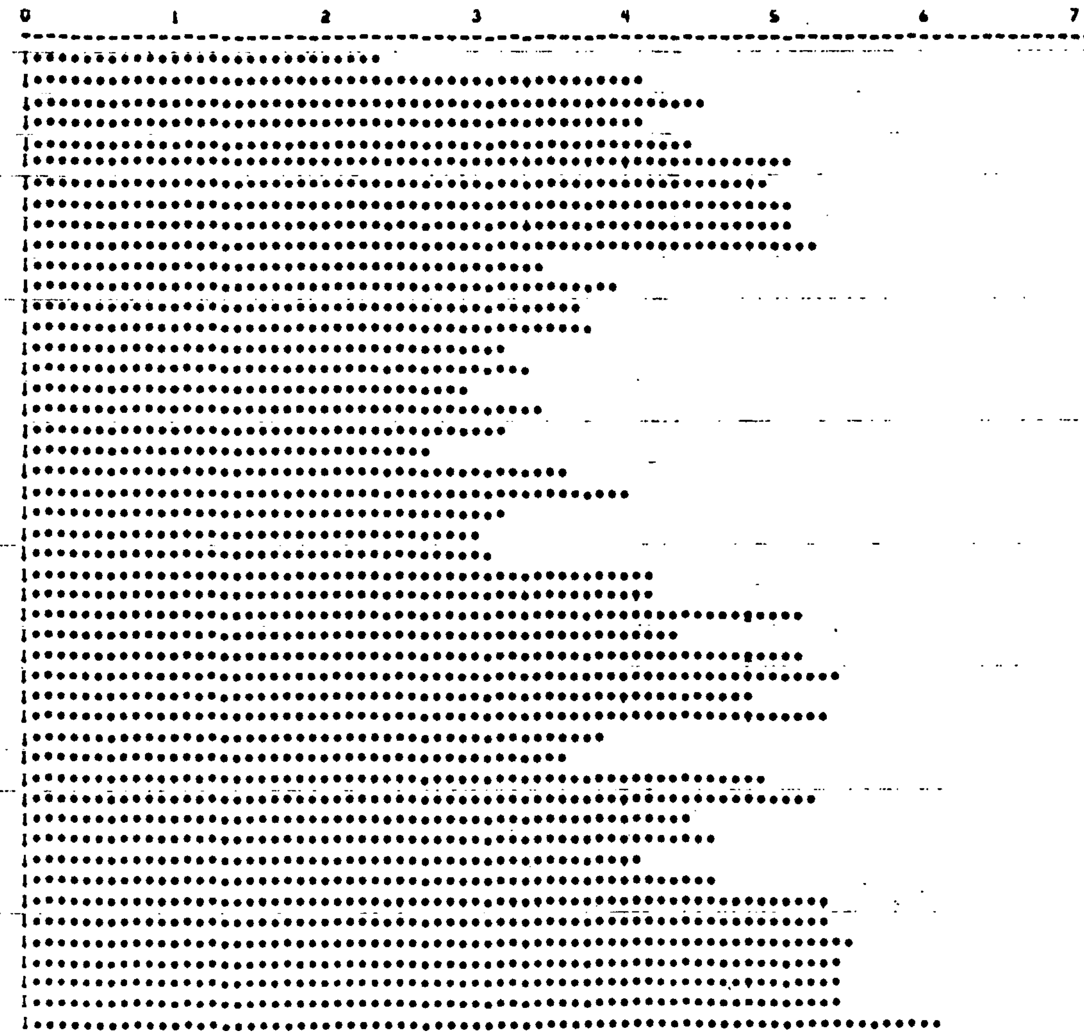
CUMULATIVE UNIPROCESSOR ACTIVITY PLOT ---- DATE 6- 8-70 TIME INCREMENT 1800 SECONDS



CUMULATIVE UNIPROCESSOR ACTIVITY PLOT ---- DATE 6-17-70 TIME INCREMENT 3400 SECON



CUMULATIVE UNIPROCESSOR ACTIVITY PLOT --A-- DATE 4-14-70 TIME INCREMENT 1800 SECONDS



CUMULATIVE UNIPROCESSOR ACTIVITY PLOT ---- DATE 5-20-73 TIME INCREMENT 900 SECONDS