SLAC-162
UC-32

MODELING THE TIME DEPENDENT BEHAVIOR OF COMPUTER SYSTEMS*

JOHN ARTHUR PFEFFERLE

STANFORD LINEAR ACCELERATOR CENTER
STANFORD UNIVERSITY
Stanford, California 94305

PREPARED FOR THE U.S. ATOMIC ENERGY COMMISSION UNDER CONTRACT NO. AT(04-3)-515

June 1973

Printed in the United States of America. Available from National Information Service, U. S. Department of Commerce, 5285 Port Royal Road, Springfield, Virginia 22151.
Price: $3.00; Microfiche $0.95.

---

*Engineer Thesis

ABSTRACT

This study develops a procedure for modeling computer systems and for analyzing their performance. Here, computer system performance is measured by the actual or potential utilization of its components, irrespective of the effect on any particular job. The utilization of a component is related to the queue length at that component. A knowledge of the time varying behavior of the queue length is useful in explaining time averaged utilization measures.

A computer program based on a queuing model was written to simulate the behavior of computer systems at the device level. The program is flexible enough to allow parametric variation of such things as the data channel structure, the scheduling methods used, and the service time distributions in force.

The computer program also keeps a running time average of the utilization factor and mean and standard deviation of the queue length at each component of the simulated system. In addition, the lengths of the queues are sampled at equally spaced time intervals during the simulation. From the resulting time series the program constructs two measures of the time varying activity at each component: the histogram and the power spectrum.

A series of nineteen experiments was run with the computer program. The experiments tested FCFS, SPTF, LPTF, RR, and preemptive SPTF scheduling and exponential, hyperexponential, hypoexponential, and constant service time distributions. The conclusions drawn from these experiments show the usefulness of time varying measures of activity and some specific phenomena associated with certain computer configurations or scheduling methods.

* Low frequency oscillatory behavior, associated with service distributions having large variance, results in poor performance unless jobs with long service times are given low priority.

* SPTF and other disciplines which favor shorter jobs tend to increase system performance. Such increases are associated with less variance in the queue length and less low frequency power in the spectrum.

* Constant, or low variance, service distributions give rise to a set of peaks in the spectra at the harmonics of the frequency corresponding to the mean service time. The amount of power in these peaks depends on the scheduling methods in force.

* Components connected on the data path between other components can act as low pass filters. Activity in the low frequency ranges may be passed between components.

PREFACE

Most studies of computer system activity have sought to characterize only the overall, or time averaged, activity of the system. The utilization factor or job queue size is used as the criterion for proclaiming one system more efficient than another. This procedure does not necessarily yield insight as to why one system is better.

In this work we emphasize measures of the time varying activity within a system. We follow the approach suggested by G. Fishman and P. Kiviat (1965) for applying statistical analysis to time series generated by computer simulation. In addition to computing utilization and average queue length, we sample the time series of the queue lengths. We use this sample to construct a histogram and to perform a spectral analysis.

The queuing model used in this study is flexible enough to represent many computer architectures or operating system designs available today. The model includes several scheduling disciplines with a preemptive version of each. It also features various types of service time distributions reflecting the different performance characteristics in different parts of a computer system.

In this report we demonstrate how such a queuing model, with a procedure for analyzing it, can be used to compare the effects of service time and choice of scheduling method on system activity. The procedures developed in this study are important tools to be used by the systems designer. The use of these procedures can aid the choice of system components and the scheduling of their use.

The author wishes to thank Professor Forest Baskett III for his invaluable advice and direction during the preparation of this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

This paper deals with one aspect of the problem of utilization of resources in a computer system. Typically, one gauges the efficiency of a computer system by time averaged measurements of activity in various components. These measurements are fine for summarizing computer utilization but they do not give any insight as to the dynamic behavior of the system. It would seem that a method for studying the time varying activity within a system could suggest possible ways to improve the efficiency.

The computer systems discussed in this report have a set of components at which all processing takes place. These components may be thought of as central processors, input/output devices, or even software modules. Each component has an associated queue for holding jobs which are waiting for processing. Queue length is used as the principal measure of activity within the system.

Chapter 2 describes the model used in this study. Some justification is made for using the computer simulation approach to this problem. The computer program which is proposed operates in two passes. The first pass is the simulation during which a sample time series of queue lengths is formed. Time averaged values for the means and standard deviations of the queue lengths and for the utilization factors are computed during this pass. The second pass processes the time series to arrive at two descriptions of the activity: the histogram and the power spectrum.

The histogram represents an approximation to the probability distribution for queue length. It provides a way of estimating the average queue length. It also can indicate, to some extent, the degree of fluctuation in activity. Since the queue length is a step function (of time), two widely separated peaks in the histogram demonstrate a significant amount of variability in the queue size.

The power spectrum gives more detail of the time varying nature of the queue length. It specifies the extent to which each frequency contributes to the total variance. The variance measures the average deviation from the mean. A large variance indicates considerable fluctuation in the activity level at that component, which can mean greater idle time and more frequent occurrences of congestion.

The principal reason for using the model is to note any correlations among the utilization factor, mean and standard deviation of the queue length, and the shapes of the histogram and power spectrum. The correlations, if they exist, can provide insight necessary to explain system performance.

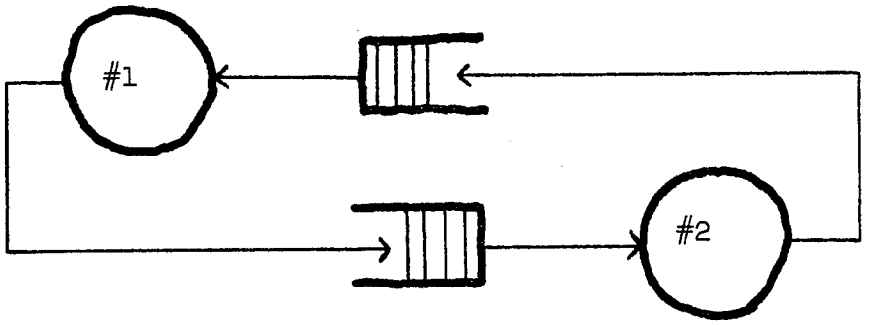
Chapter 3 presents the basic theoretical background on which the construction of our model is based. This includes terminology and basic results from queuing theory and the theory of stochastic processes, as well as a description of procedures used in computing the spectrum of a sampled time series.

Chapter 4 describes the computer program designed for this study, including details of the internal operation and advice to the user on how to use the program.

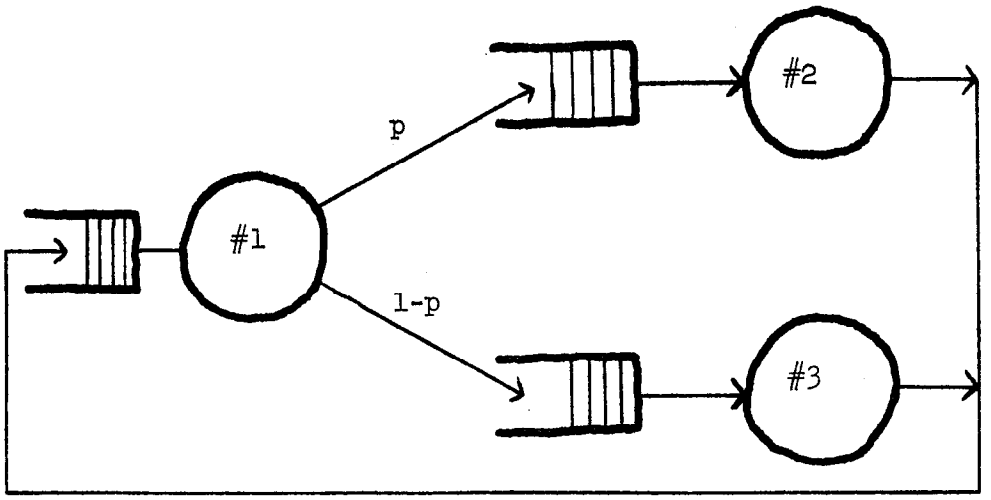
Chapter 5 presents the results of using the computer program to analyze several small systems. These are not intended to represent entire computer systems but rather certain critical subsystems. The structure of each example follows one of three formats which are illustrated in Figure 1. Structures 1 and 2 are closed systems in that they are initialized with a set of jobs and jobs may neither enter nor leave. Such models are realistic when used to study the behavior within some subsystem of a larger system and where there is some dominant process controlling the number of jobs in that subsystem. Structure 1 consists of two components "feeding" each other. This can be viewed as a simple CPU with a disk storage device attached. The analogy is valid as long as there is some process such as a scheduler restricting the number of jobs in this subsystem. Structure 2 has a CPU component "feeding" two disk components. Structure 3 has the same structure as structure 2 except that it is an open subsystem with jobs entering and leaving the subsystem at the CPU. Using these three basic structures, we assign various combinations of service time distributions and scheduling disciplines and comment on the statistical results.

Chapter 6 summarizes the experimental results of Chapter 5, states some specific conclusions characterizing system performance, discusses the value of this type of investigation, and suggests areas of further study.

Structure 1



Structure 2



Structure 3

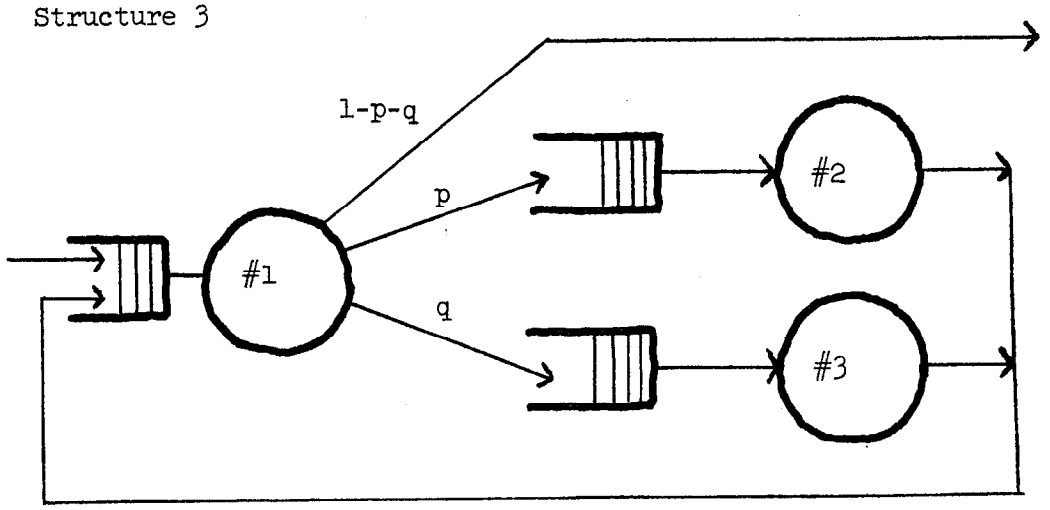


FIGURE 1. Basic Structures of Systems Studies

CHAPTER 2

GENERAL DESCRIPTION OF THE MODEL

The first problem one encounters in an investigation of this sort is the formulation of the model. The model will either be mathematical or of the simulation type. Naylor, et al., (1966) have presented a rather strong case for using simulation in the queuing model with which this paper deals. An outline of the argument follows.

A mathematical model consists of a set of equations characterizing the behavior of some process. If these equations describe the precise operation of the process, the model may be said to be deterministic. If they describe some probabilistic behavior, the model is called statistical.

A simulation model involves the use of a computer program which mimics the operation of the actual process to the level of detail required. A deterministic simulation model is supposed to reproduce exactly the important events in the flow of the process. A stochastic simulation attempts to represent the statistical properties of a process which must be viewed as random.

This investigation is concerned with activity at the device level. For instance, a disk access or an uninterrupted execution of a central processor program would represent a quantum of activity. Now in a real computer system the actual lengths of these quanta are job dependent. In order to construct a deterministic model, one would have to model the computer system in sufficient detail to determine the precise length of any processing quantum. It is theoretically possible to model such detailed characteristics as the access times and rotation rates of disks

and the settling time of the logic circuitry. However, such a model is useful only for studying one particular system. This paper is concerned with obtaining general procedures and results which then may be applied to specific systems. This makes the statistical approach the only practical one.

The model should be able to handle systems with up to about twenty components with a rather arbitrary set of communication links. Further, it must be flexible enough to allow one to change the scheduling methods or service time distributions without a major reorganization of the model. This type of flexibility is well beyond the present capabilities of mathematical analysis which eliminates mathematical modeling as a viable alternative.

The one remaining method, which is the one adopted for this study, is to construct a simulator, using Monte Carlo techniques, capable of duplicating the statistical properties of our computer system. This is possible because a simulation model requires one to construct only the significant modules of the system and describe how they act and interact. For a computer system this simply requires a knowledge of its architecture and performance characteristics.

Now, on a gross scale, the components of most computer systems can be described as processors having certain operating characteristics. For the purpose of creating a simulation model at this level of detail, it is sufficient to specify a scheduling method and a service time distribution for each processor. The means of communication among components can be described by a set of paths along which jobs are routed between components. Setting up the simulator for a particular system,

then, requires the selection of a set of components with associated scheduling methods and service distributions, and a set of paths representing the routing structure.

This essentially is the reasoning wich led to the construction of a computer program to aid in the analysis of computer system activity.

CHAPTER 3

THEORETICAL BACKGROUND

This chapter discusses the theoretical framework upon which the model is built. The section on queuing theory gives the basic definitions and methods used to construct models of the systems with which this study deals. The section on stochastic processes states fundamental results needed to develop a procedure for studying time series. The section on spectral analysis of real time series presents various methods and techniques for estimating the power spectrum from a set of sampled data.

A. Queuing Models

Included among the many applications of queuing theory are production lines, machine shops, telephone networks, and computer systems. Probably because of this, several terminologies have been used to describe the subject. For example, Conway, et al., (1967) speak of a set of "jobs" on which "operations" are performed by a set of "machines." Morse (1958), on the other hand, deals with "units" which are "serviced" at "facilities." In this report we use an amalgamation of the nomenclature which seems suitable for describing computer systems.

There is a set of items, called jobs, which arrive at the system for servicing by a set of items, called service centers. There are one or more servers at each service center. These servers are indistinguishable in the sense that a job which has arrived at a particular center can be serviced by any server and the service time is the same for each server. Each service center has a queue for holding jobs when all servers at the center are busy. The queue can be finite or infinite in

length and may or may not allow jobs to leave the queue without being serviced (reneging).

After a job has entered the system, it is serviced sequentially by some subset of the service centers before leaving the system. In many cases, such as with a production line, the sequence of service centers visited by a job is easily specified in advance. However, in a model of a computer system it is not necessarily clear to the researcher where, among several possible centers, a particular job is to be routed. For this reason, a probabilistic routing method of the following form can be used. A job, i, at a particular service center, j, is assigned a probability $P_{jk}(i)$ of being routed to service center k, where k indexes the set of service centers. We must have $P_{jk}(i)$ non-negative for all j, k, and i and the sum over k of $P_{jk}(i)$ must be 1 for all j and i. Choosing $P_{jk}(i) = 1$ for some j, k, and i gives the deterministic routing case.

Similarly, the time which a job spends being serviced at a center could be determined by the researcher. However, in most cases this time can be characterized only approximately by a probability distribution covering all jobs processed at that center.

Similar, too, is the means of characterizing the interarrival times of jobs into the system. Again, a probability distribution is generally used to describe this aspect of the model.

The following are some of the probability distributions commonly used in the simulation of real systems.

1. The uniform distribution on an interval (a,b) has density function

$$f(x) = \begin{cases} 1/(b-a) & a < x < b \\ 0 & \text{otherwise} \end{cases}$$

which gives a mean value $(b+a)/2$ and a variance $(b-a)^2/12$. The

uniform distribution is valuable in implementing a stochastic routing scheme in a queuing model. A uniform variate can be used to determine the destination service center of a job. Most importantly, the uniform distribution on (0,1) is the basis for computing variates of many other distributions.

2. The exponential distribution with parameter a has density function

$$f(x) = ae^{-ax} \qquad a > 0,\ x \geq 0$$

with mean $1/a$ and variance $(1/a)^2$. The characteristic function is $a/(a-is)$. This distribution is characterized by generally small variates with "occasional" large ones. One of the simplest queuing models, the M/M/1 system, consists of a single input channel with exponential interarrival times and a single server with exponential service times.

3. The hyperexponential distribution has two parameters, a and p, and represents the weighted sum of two exponential processes. A variate of this distribution is obtained by taking, with probability p, a variate from an exponential distribution with parameter 2pa, or with probability (1-p), a variate from an exponential distribution with parameter 2(1-p)a. The density function

$$f(x) = 2ap^2 e^{-apx} + 2a(1-p)^2 e^{-2a(1-p)x}$$

give a mean of $1/a$ and variance of $(1/(2p(1-p))-1)/a^2$. This distribution has exponential characteristics but with greater variance as controlled by p.

4. The hypoexponential distribution with parameters a and n represents the sum of n exponential distributions, each having mean na. A variate is obtained by summing n such independent exponential variates. The mean is, then, $1/a$ and the variance is $(1/a)^2/n$. The characteristic function is $(a/(a-is))^n$. This distribution has exponential characteristics but with smaller variance as controlled by n. As n approaches infinity, random variables from this distribution approach the constant $1/a$.

When a service center finishes the processing of a job, it looks at its queue to see if there are any jobs waiting. The procedure used to select the job to be serviced among several candidates in the queue is called a scheduling discipline. The scheduling method may be based on any one of a number of attributes of jobs in the system. The choice could be based on time of arrival, amount of processing, due-date of the job, or even some randomly assigned number. For each attribute there are usually several criteria that can be used to select a job for processing, based on its value of that attribute. Conway, et al., (1967) present a thorough account of these methods. The scheduling methods used in this report are summarized in the following paragraphs.

Most computer systems allow jobs to be classified according to their relative importance or urgency. Jobs are placed in one of a finite number of classes based on a priority value attached to the job when it enters the system. The selection discipline is then specified by two rules which determine which of several non-empty classes and which of several jobs in that class should be chosen. The first rule, generally, is to pick the class with the largest associated priority value. Several examples of the second rule are now discussed. First-come, first-served (FCFS) means the selection of the job with the earliest time of arrival. It is the type used at butcher shops where one is required to "take a number." Shortest-processing-time-first (SPTF) means the selection of the job which will take the shortest time to service. Use of this method requires either prior knowledge of the service times or a reasonable estimate of these times. Longest-processing-time-first (LPTF) has the same nature as SPTF except the maximum,

rather than the minimum, service time is used as the selection criterion.

Scheduling methods may be further classified as to whether they are preemptive. A preemptive discipline allows a job to be interrupted during processing and returned to the queue while another job is serviced. An example is round-robin (RR) scheduling which is one preemptive version of FCFS. RR takes jobs on a FCFS basis but interrupts any unfinished job after a period of time called the time slice. An interrupted job is put at the end of the queue with its service time diminished by the size of the time slice. The next job on the queue is then given a time slice during which it may or nay not finish servicing at that center. Any scheduling discipline has a preemptive version which simply involves rescheduling whenever a new job arrives at the service center. Such a job may, if it satisfies the selection criterion, interrupt a job being serviced. The preemptive methods used in this study are work conserving in that preempted jobs resume where they left off and the preemption itself requires no overhead processing.

B. Stochastic Processes

The function $q(t)$ defined on an interval $(0,T)$ and representing the queue size at a particular service center can be considered, for purposes of statistical analysis, to be one possible instance of a stochastic process. A stochastic process is a collection of random variables $X(t)$ defined on an index set $(t \mid t \text{ in } (0,T))$. The reader is referred to Parzen (1962) or Karlin (1969) for general information about stochastic processes and to Jenkins and Watts (1968) for a development of spectral analysis. The definitions now presented are from these sources.

A stochastic process is called strictly stationary if the joint probability distributions of $(X(t_1),X(t_2),\ldots,X(t_n))$ and $(X(t_1+h), X(t_2+h),\ldots,X(t_n+h))$ are the same for all t_i in the index set and for all $h > 0$. In particular, a strictly stationary process implies that $X(s)$ and $X(t)$ have the same probability distribution for all s,t in the index set. The process is time invariant in the sense that $(X(s)-X(t))$ depends only on the value of $(t-s)$, not on s or t explicitly.

A stochastic process is called covariance, or second order, stationary if it has finite second moments and if $\mathrm{Cov}(X(s),X(t))$ depends only on $(t-s)$. For a covariance stationary process, the autocovariance function is given by $C(u)$, the covariance at "lag" u, which is $E((X(t)-\mu)\ (X(t+u)-\mu))$ where μ is the mean of $(X(s))$. The variance of $(X(s))$ is given by $C(0)$. The normalized function $C(u)/C(0)$ is called the autocorrelation function and essentially measures the dependence of a process on its past. It is shown by Blackman and Tukey (1958) that

$$C(u) = \lim_{T \to \infty} (1/T) \int_{-T/2}^{T/2} (X(t)-\mu)\ (X(t+u)-\mu)dt$$

can be reduced to the form

$$C(u) = \int_{-\infty}^{\infty} P(f)\ e^{i2\pi fu}\ du$$

where

$$P(f) = \lim_{T \to \infty} (1/T) \left| \int_{-T/2}^{T/2} (X(t)-\mu)\ e^{-i2\pi ft}\ dt \right|^2$$

Hence, $C(u)$ and $P(f)$ are Fourier transform pairs and

$$P(f) = \int_{\infty}^{\infty} C(u)\ e^{-i2\pi fu}\ du.$$

$P(f)$ is called the power spectrum. If $C(u)$ is replaced by the auto-

correlation function in the above equation, then P(f) is usually called the spectral density or power spectral density. The power spectrum yields the same information, in the frequency domain, as the autocovariance function, in the time domain. Setting u = 0 in the reduced form for C(u), one obtains the relation

$$\int_{-\infty}^{+\infty} P(f)df = C(0)$$

As pointed out by Fishman and Kiviat (1956), P(f)/C(0) has the properties of a probability density. Moreover, P(f)df represents the contribution to the total variance by frequencies in a band of width df around frequency f.

A highly autocorrelated process is one for which C(u) is relatively large for large u. That is, the state of the process is significantly dependent on the state u time units previous. Such a process has a rapidly decreasing power spectrum since the lower frequencies predominate.

C. Spectral Analysis of Real Time Series

The practical problem in the spectral analysis of real time series is to estimate the autocovariance function or power spectrum from a series of equispaced samples of some real or simulated process.

Two approaches to estimating the power spectrum are suggested by the equations in the previous section. Webb (1970) calls these approaches the "indirect" and "direct" methods.

The "indirect" method involves getting an estimate of the autocovariance function C(u) and Fourier transforming it to arrive at the estimate of the spectral density P(f). The number of lags, or values

of u, is usually chosen to be near N/10 where N is the number of sample points. For each lag, u, one computes the mean lag product

$$R(u) = (1/N) \sum X(t)X(t+u)$$

to get the sample autocovariance function. The sample spectral estimate at frequency f is then

$$S(f) = (1/2\,\pi) \sum R(u)\, e^{-i2\pi fu}.$$

Now if the number of lags, m, is proportional to N, then the number of operations required to get the function R is proportional to N^2. Similarly, the number of operations to get S from R is proportional to N^2. Hence, the total computation time is proportional to N^2.

The "direct" method uses the definition of P(f) to arrive at a spectral estimate by forming the modulus of the Fourier transform of the entire sampled data set. The number of operations required by the straightforward transform method is proportional to N^2. However, by using the Fast Fourier Transform Algorithm (FFT) as described by Singleton (1967), among others, one can reduce this magnitude to ($N \log_2 N$) when N is a power of two.

A certain amount of foresight is needed in choosing the length of sampling interval, Δt, and the number of samples, N.

It is a fundamental result of communication theory that, from a time series with spacing Δt, meaningful information can be obtained for frequencies up to $N_y = 1/(2\,\Delta t)$ which is called the Nyquist, or folding, frequency. If the data contains frequencies above N_y, these frequencies will be folded back into the interval $(0,N_y)$ making the resulting spectrum difficult to interpret. This effect is called aliasing. The re-

searcher either must know in advance that no frequencies above N_y are present in the data, or must remove those frequencies with a digital filter.

With Δt fixed, the number of samples, N, determines the length of the sampling period $T = N\,\Delta t$. For the direct method of computing the spectrum by using the FFT algorithm, either N must be a power of two or zeroes must be added up to the next power of two. The frequency resolution is $\Delta f = 1/T$, which is also the value of the lowest frequency discernible. If one is really interested in frequencies near $(1/T)$, one must choose a much larger sampling period in order to afford those frequencies better representation in the time series.

It is shown in Jenkins and Watts (1968) that, for a finite length record, the mean of the sample spectrum estimator is the Fourier transform of $c(u)w(u)$ where

$$w(u) = \begin{cases} 1 - |u|/T & , \quad |u| \leq T \\ 0 & , \quad |u| > T \end{cases}$$

An equivalent statement is that the mean of the sample spectrum estimator is the convolution product of $P(f)$ with $W(f)$ where $W(f)$ is the Fourier transform of $w(u)$:

$$W(f) = T\,(\sin((\pi)Tf)/(\pi)Tf)^2.$$

Hence, the mean of the spectral estimate corresponds to looking at the real spectrum through a filter with impulse response $W(f)$. The functions $w(u)$ and $W(f)$ are called the lag window and spectral window respectively.

The effect of this is that the estimate has an unduly large variance. This situation is usually remedied by applying another lag window

to the sampled data in order to reduce the variance. Unfortunately, this procedure leads to distortion or biasing of the estimate. The approach recommended by Jenkins and Watts (1968) is to choose a spectral window empirically by a process called window closing.

This process depends on the characteristic of a spectral window called its bandwidth. The bandwidth of a spectral window is the width of a rectangular window which yields the same variance for the spectral estimate as the given window. From this, one gets the fundamental relationship

$$(\text{VARIANCE})\ (\text{BANDWIDTH}) = \text{CONSTANT}.$$

Hence, a small bandwidth is associated with a large variance and a small bias.

Window closing involves computing a spectral estimate using a window with large bandwidth. This will give a smooth, stable estimate but one lacking in detail. The procedure is then to compute further estimates with smaller bandwidths until the desired degree of detail is achieved or instability appears. Jenkins and Watts even recommend presenting three spectral estimates: one reasonably stable and detailed, one slightly lacking in detail and one slightly unstable.

We now state a few of the results from Jenkins and Watts (1968) regarding the distribution of, and confidence intervals for, spectral estimates. It is shown that I/T represents the ratio of the variance in a smoothed estimator to that of the raw estimator, where

$$I = \int w(u)^2\, du.$$

Now, assuming that the spectrum is relatively smooth, the quantity $vS(f)/P(f)$ has a chi-square distribution with v degrees of freedom,

where $v = 2(T/I)$. Here $S(f)$ is the smoothed estimator. Further, the interval for $\log(P(f))$, corresponding to confidence value a, is $\log(S(f)) + \log(v/x_v(1-(a/2)))$, $\log(S(f)) + \log(v/x_v(a/2))$. The size of the confidence interval for $P(f)$ is independent of frequency when $S(f)$ is plotted on a logarithmic scale.

CHAPTER 4

COMPUTER PROGRAM

In this chapter we describe the actual implementation of the computer program used in this study. First we discuss what the program does. Then we present the detailed operation of the program. Finally, we give some notes on how to use the program.

A. Detailed Description of the Model

The computer program to be used in the study will simulate the gross behavior of computer systems. The structure of the model follows that of a job shop. A set of service centers represents the components of the system. These could be the CPU or CPU's, disk or tape units, interactive terminals, card readers, printers, etc. The service centers may not correspond precisely to actual hardware. Instead, they may represent certain software modules which perform various tasks. The service centers may have more than one server. This allows one to regard all of the terminals or all of the disk units in the system as a single center.

Jobs enter the system stochastically according to some probability distribution. After the jobs enter the system, they are routed from one service center to another. Each center uses a designated scheduling method. A job is assigned a stochastic service time at each service center upon arrival. Each service center has its own probability distribution defining the service times there. The method of routing jobs among centers is also probabilistic in nature. The routing paths or data links are described by a two-dimensional matrix (R_{ij}). The r_{ij}-th element represents the probability that a job at center i will be routed to center j. Note that the routing method does not depend on any characteristic of the

job to be routed. The more general approach discussed in Chapter 2 has not been implemented.

Throughout the simulation, whenever a change in state occurs at a particular service center, a set of time averaged statistics is updated. The random variables averaged include the queue sizes and number of servers busy. The latter represents the utilization factor at a service center.

To measure system activity we sample the length of the queue at each service center. N equally spaced samples are collected during the simulation. After the simulation is completed, the sample time series are processed to get the histogram and the power spectrum.

We have used the "direct" approach described by Webb (1970) for obtaining an estimate of the power spectrum. Since this involves the use of the FFT algorithm, we have chosen to make the sample size, N, a power of two. We use a cosine data window before transforming and smooth the spectral estimate with a moving average. Since we are most interested in the qualitative aspects of the spectrum, we have not implemented any sort of confidence tests. For further information about the "direct" approach to power spectral estimation, the reader is referred to Webb (1970).

The power spectrum is a meaningful measure only if the time series is sampled from a covariance stationary process. We are most interested in simulation times large enough that ineffficiencies in the computer system reflect significantly higher operating costs. This could be on the order of half an hour, an hour, or more. We are not particularly interested in simulation times of several minutes or less. It seems reasonable to assume that our systems are covariance stationary over periods of

an hour. We should not expect appreciable change in the distributions of activity levels over this period of time. However, over much shorter periods of time the effect of a particular job could invalidate the assumption. After a period of many hours slow trends could ruin stationarity. Another such condition is the dedication of the entire system to certain small groups of people at specific times.

B. Internal Operation of the Program

The program can be divided into two logical parts which execute sequentially. The first is the simulator which also samples the queue lengths periodically. The second uses this sampled data to compute the histogram and power spectrum. We discuss the structure of the simulator first.

The state of a job includes such data as its priority class, its arrival time at the current service center, its service time at, and its scheduled departure time from, the current service center.

The state of a service center may be completely characterized by the set of jobs at the center either being serviced or awaiting service.

The only activity within the system, aside from jobs entering or leaving the system, is the movement of jobs from one service center to another.

The previous statements are intended to suggest the following way of representing the data.

Each job is represented as a "data item" with values assigned to the attributes describing it. In the program these values are specific elements of arrays. For job J, JPC(J) is the job priority class, JATIME (J) is the arrival time at the current service center, etc.

We represent the set of jobs at a certain service center, I, by a queue of the data items for these jobs. Hence, for each job we form two new attributes, a left link and a right link with values JLLINK(J) and JRLINK(J) respectively. These links define the left and right neighbors of a job on a queue. We also define a data item with these link attributes for each service center in the system. The links for center I are SCTL(I) and SCHD(I) respectively and this data item is used as a header for the queue. The queue is then represented as a two-way, circularly linked list. In order to differentiate those jobs on the queue which are being serviced from those which are waiting, we put the active jobs at the head of the queue (pointed to by SCHD(I)). The inactive jobs are linked at the end. The queue is divided effectively into an active queue and an inactive queue which are linked together. For each header data item we define an attribute having the value SCHDIA(I) which is a link to the first job on the inactive queue for center I.

We have now specified completely a way of representing the instantaneous state of the system. We must now describe the dynamics of the system, i.e., how changes of state occur. We shall forget, for the moment, jobs entering or leaving the system.

The first question the simulator must answer is what to do next from a given state. The only thing that can cause a change of state in this system is the termination of a job at a service center. This will cause a change in the queue size and perhaps the number of busy servers at that center. Now each job J which can finish servicing at a center is in that center's active queue and has a value, JDEP(J), assigned as the scheduled departure time. Hence, the simulator can decide what to do next by

searching all of the active queues and finding the job with the earliest departure time. That job will trigger the next significant event. It is appropriate to call the aggregate of the active queues the event queue.

Now that the simulator has determined what will trigger the next event and when, it can take the appropriate action, which is now described. First a routine is called whose duty it is to sample the queue sizes every Δt time units. This routine stores the samples in a buffer for later statistical analysis. Processing of the actual event proceeds as follows. Assume job J finishes being serviced at center I. Then J can be removed from the active queue at center I. Where job J is sent is determined stochastically using an array ROUTE corresponding to the routing matrix (R_{ij}) mentioned in the previous section, except that ROUTE(I,-) is a cumulative distribution. A number N is obtained from a pseudo-random number generator (uniform on (0,1)). The service center to which job J is sent is the least number K for which N > ROUTE(I,K). We must have ROUTE(I,M) $\leq$ ROUTE(I,N) for M $\leq$ N and ROUTE(I,$\max_{sc}$) = 1 for all I. Note that we can set up a deterministic routing scheme by setting ROUTE(I,M) = 0 for M < K and ROUTE(I,M) = 1 for M $\geq$ K. This represents a fixed path from center I to center K.

Before we can actually move the job to the new service center, we must realize that the states of both the old and new centers are going to change. Hence, we must update, at this point, the time averages of the random variables we are measuring.

When the service center, K, to which job J is to be sent, has been determined, the data item for job J is linked to the inactive queue for center K. The new service center has a specific probability distribution

associated with its service times. A sample is taken from this distribution using the pseudo-random number generator and used as JSTIME(J), the time needed to service the job at this center. The current "clock time" is recorded as JATIME(J), the arrival time at the current center.

Now, since a job has left center I, there is certainly an idle server there. We should now search the inactive queue for any jobs awaiting service. Similarly, the inactive queue at center K is certainly non-empty at this moment. If there is an idle server, it can be put into use. So, it may be necessary at both the old and new centers to select a job to be serviced next. The way the job is chosen depends on the scheduling method in force at the service center. The selection could be based on the job's arrival time or assigned service time. Once a job is selected to be serviced, its data item is delinked from the inactive queue at that center and placed on the active queue. Since it is now also a member of the event queue, it must have a value assigned to the departure time attribute. Generally, this departure time, JDEP(J), is set to the sum of the current "clock time" and the service time JSTIME(J). When we have completed this processing for both the old and the new service centers, we have completed the action dictated by the event. We are now ready to select the next event in the event queue.

Thus far, we have neglected the problem of how jobs enter and leave the system. An attempt was made to relate these activities as closely as possible to the data structures just described. The following solutions were implemented.

Service center number one was arbitrarily designated a pseudo-service center or "doorman" who controls the entry of jobs into the

system. This service center has an active queue which has NJPC pseudo-jobs in it, where NJPC is the number of job priority classes allowed. Each job priority class may have associated with it a particular arrival time distribution. The departure time value for these pseudo-jobs is the time at which a new arrival of that priority class occurs. When an arrival occurs, a sample is made of the arrival time distribution and added to the current "clock time" to get the time of the next arrival in that class. A new arrival is processed by assigning an empty data item from the free storage list kept of such items. The priority class is assigned. The job is then routed to its first "real" service center using the routing methods previously described, where center one is the old center. Note that the queue of pseudo-jobs is part of the event queue and hence is searched each time the simulator looks for something to do.

If we are using N "real" service centers, we assign these centers numbers 2,3,...,N+1. We then designate another pseudo-service center with number N+2. This service center is the output port of the system. Jobs are routed here when they have been completely processed. Any job arriving here is dropped from the simulation. Its data item is returned to the free storage list. Hence, center N+2 has empty active and inactive queues.

We will now further elaborate on some particular topics discussed earlier.

We only consider Poisson arrivals for each of the job priority classes. That is, the interarrival times are exponentially distributed with different means for each class. We allow one to restrict the num-

ber of arrivals in a class during the simulation. If one limits this number, sets the mean arrival time relatively small, and never routes a job out of the system (to service center N+2), one can simulate a "closed" system in which a set of jobs circulates internally.

We considered four different types of service time distributions. These are the exponential, hyperexponential, hypoexponential, and constant distributions. All but the constant distribution require one or more calls to the pseudo-random number generator.

We have allowed four different types of scheduling disciplines: shortest (remaining) processing time, longest (remaining) processing time, first-come first-served, and round-robin. These methods decide which of a set of equal priority jobs is to be served first. Among a set of jobs in an inactive queue, a job with priority class greater than that of any other job will be selected first, regardless of the scheduling method in force at that center. We also have implemented preemptive scheduling which may be specified along with any of the basic disciplines already mentioned. The interpretation of preemptive FCFS or RR scheduling is that a newly arrived job can preempt one being serviced only if its job priority class is higher.

It should be pointed out that, for a job J, the value JSTIME(J) has different interpretations depending on where J is. When J is on an inactive queue, JSTIME(J) represents the remaining amount of time job J requires for servicing at that center. When J is on an active queue, JSTIME(J) represents the remaining amount of time job J will require for servicing at that center after its current servicing period is over. For non-preemptive SPTF, LPTF, or FCFS scheduling, the value of JSTIME(J),

when J is on the inactive queue, will be the assigned total service time at that center. When J is selected for servicing, JSTIME(J) will be set to 0 and JDEP(J), the departure time, will be set to "clock time" plus JSTIME(J). When a job J is selected for processing in a RR scheduled center, the remaining service time is compared to the assigned time slice at that center. The smaller of the two quantities is added to "clock time" to yield JDEP(J) and subtracted from JSTIME(J) to get the new JSTIME(J). RR scheduling, then, introduces the possibility that a job, J, may not be fully serviced when its event time occurs, i.e., when JDEP(J) equals "clock time." Hence, we must modify our method of processing an event when JSTIME(J) is greater than zero. Instead of routing the job to a new service center and assigning a new service time, we route it back to the inactive queue of the same center without changing its JSTIME(J) value.

Preemptive scheduling introduces the following additional processing. If a newly arrived job does not qualify to preempt a job being serviced, it is put on the inactive queue with its assigned service time. If it does qualify to preempt, it is processed like a regular job being selected for servicing, as just described, and put on the active queue. The job, J, which is preempted, however, has not completed so it must be returned to the inactive queue. In addition, its remaining service time, JSTIME(J), must be incremented by the difference between JDEP(J) and "clock time." This increment is the amount of additional service time the simulator thought J would have accrued before leaving the active queue.

This completes the description of the mechanics of the simulator. The second phase of the program consists of doing the statistical analysis and providing the output.

The task of computing and printing the mean and standard deviation of the queue lengths is done first.

If a print-out of the actual sampled data is desired, it is retrieved in "normal" order from the buffer and printed.

The sampled data is then retrieved in "even-odd" order as needed for doing a Fast Fourier Transform of a double array of real data. If desired, a histogram is computed at this point and a Calcomp plot produced.

The power spectrum of this data is now computed. First the sample mean is subtracted from the data. Then the spectral window is applied. A routine is then called which computes the FFT of the data and then the modulus of the transform to get the raw power spectrum. The raw power spectrum is then smoothed using a weighted moving average. The logarithm of the spectrum is then displayed on a Calcomp plot. The subroutines used for these computations are part of a package of routines for doing spectral analysis presented in the report by Webb (1970).

The above statistical procedure is applied to the data for each real service center which completes the processing of one set of inputs.

A listing of the computer program used in this study appears in the Appendix.

C. User Notes

In designing an experiment to be performed with this program, the user has two main responsibilities. The first is to ensure that the

abstract model reflects the basic nature of the real computer system. This means using a representative structure and choosing realistic parameters. Secondly, the user must select certain parameters which control the accuracy and validity of the statistical results and the efficiency and speed of the computer run.

The data is input to the program in blocks. Block 1 contains the following variables which give the basic dimensions of the simulation:

NSC - the number of service centers in the model, including the two pseudo-service centers for handling arrivals and departures;

NJPC - the number of job priority classes which are ordered in increasing priority from 1 to NJPC;

SEED - integer used to initiate the pseudo-random number generator;

NP2 - 2**NP2 is the number of samples of the queue lengths to take at equal intervals during the simulation;

NS - number of points used in the moving average applied to the power spectral estimate;

PT - the percent taper used in the window applied to the sampled time series;

LIMIT - time in arbitrary units to stop the simulation;

T2SAMP - time within the simulation at which sampling is to begin.

Block 2 contains the following data characterizing the job priority classes:

PCMAT(J),J=1,...,NJPC - the means of the exponential distributions of the interarrival times for the classes;

MJP(J),J=1,...,NJPC - the maximum number of jobs of a particular class to allow in the system at any given time.

Block 3 has all of the parameters needed to describe the behavior of the service centers:

SCNS(J),J=1,...,NSC - the number of servers at each service center;

SCSM(J),J=1,...,NSC - an integer number specifying the scheduling method in effect at each service center;

SCMAXQ(J),J=1,...,NSC - the maximum queue size to be recorded during sampling for each center (does not affect actual queue size, only expedites packing of the sampled data);

SCSD(J),J=1,...,NSC - an integer number specifying the type of service time distribution in effect at each center;

SCMST(J),J=1,...,NSC - mean service time for the distribution specified by SCSD(J);

SCD1(J),J=1,...,NSC - extra parameter needed to complete the description of the distribution specified by SCSD(J) (for the hyperexponential distribution SCD1(J) is the parameter p; for the hypoexponential distribution SCD1(J) is the number of component exponentials).

Block 4 contains only the routing matrix ROUTE(J,K),J=1,...,NSC, K=1,...,NSC which has already been described.

Block 5 contains no data but initiates execution of a particular case.

The running of multiple cases requires only stacking of those blocks of data which are to be changed and inserting a "block 5" for each case to be executed.

The amount of list output one obtains can be controlled by the variable ILIST which is assigned a value at compile time. The value 30 provides listings of all but the diagnostic print-out of each step of the simulation and of the intermediate results of the power spectral analysis.

CHAPTER 5

EXPERIMENTS WITH THE PROGRAM

A. Experimental Procedure

The previous sections of this report have laid the foundation for the study. We have described in detail the structure of the simulation model and the methods for analyzing data obtained from it. The whole procedure has been incorporated into a flexible computer program. In this section we demonstrate how this program is used to learn details about the behavior of systems described by the model.

Each example used in this section has one of the three basic structures shown in Figure 1. For purposes of exposition we will number the components, or service centers, starting at one, even though the computer program numbers from two. Structure one is a two service center loop. Jobs at service center one are routed to service center two and vice versa. Note that this is an example of the special deterministic case of the general stochastic routing method used in the program. Structures two and three are both three service center systems. Jobs at service center one may be sent to either center two or center three. Jobs at centers two or three always return to center one. Structures two and three, then, incorporate both deterministic and stochastic routing. Structures one and two are closed systems. Once initiated, jobs may neither enter nor leave the system. For these examples, we use the special feature of the program which cuts off arrivals to the system when there is a specified number of jobs in the system. If we then refrain from routing jobs out of the system, the closed loop is established. We have somewhat arbitrarily chosen the number of jobs to be

15 for examples discussed in this report. Experiments run with 30 jobs in the system indicated that the time varying behavior was not significantly different for those cases tested. Structure three is an open system with jobs arriving at and departing from service center one.

The procedure followed was to systematically vary, for each structure, parameters describing the scheduling methods used and the arrival and service time distributions in force. Table 1 lists the parameters for those cases selected for inclusion in this report. Table 2 lists values for the important time averaged variables computed by the program.

The computer program does not impose any specific definition for the unit of time. The experimenter is responsible for relating the arbitrary units of the program to real time units. For this study, the time unit is considered to be on the order of 1-10 seconds. Depending on the specific choice, the duration of the simulation ranges from less than half an hour to over an hour. This is approximately the range of times mentioned in Chapter 4 as being most interesting. Because of the range of interpretations for the time unit, we do not label explicitly quantities with units of time. These quantities are the means of the service and arrival time distributions, the time slices for RR scheduling, and the duration of the simulation. Similarly, we use the term "cycles" for the frequency units. One "cycle" should be considered 1/n cycles/second if there are n seconds in one time unit. Finally, the mean and standard deviation of the queue length obviously have units of "number of jobs." Hence, we shall omit the units when listing these quantities.

TABLE 1

SERVICE CENTER PARAMETERS FOR EXAMPLES STUDIED

Example	Structure	Center 1				Center 2				Center 3			
		Sched.	Serv. Distr.	Mean	Param.	Sched.	Serv. Distr.	Mean	Param.	Sched.	Serv. Distr.	Mean	Param
1	1	FCFS	exp	0.05		FCFS	exp	0.05					
2	1	SPTF	exp	0.05		SPTF	exp	0.05					
3	1	FCFS	hyper exp	0.05	0.05	FCFS	hyper exp	0.05	0.05				
4	1	SPTF	hyper exp	0.05	0.05	SPTF	hyper exp	0.05	0.05				
5	1	FCFS	hyper exp	0.2	0.05	FCFS	const	0.25					
6	1	LPTF	hyper exp	0.2	0.05	FCFS	const	0.25					
7	1	SPTF	hyper exp	0.2	0.05	FCFS	const	0.25					
8	1	RR 0.06	hyper exp	0.2	0.05	FCFS	const	0.25					
9	2	FCFS	exp	0.05		FCFS	const	0.350		FCFS	const	0.2	
10	2	FCFS	exp	0.05		RR 0.05	const	0.35		RR 0.05	const	0.2	
11	2	FCFS	exp	0.1		FCFS	const	0.35		FCFS	const	0.2	
12	2	FCFS	exp	0.1		RR 0.05	const	0.35		RR 0.05	const	0.2	
13	2	FCFS	exp	0.1		FCFS	hypo exp	0.35	100.0	FCFS	hypo exp	0.2	100.0

TABLE 1 - continued

Example	Structure	Center 1				Center 2				Center 3			
		Sched.	Serv. Distr.	Mean	Param.	Sched.	Serv. Distr.	Mean	Param.	Sched.	Serv. Distr.	Mean	Param.
14	2	LPTF	exp	0.1		RR 0.05	const	0.35		FCFS	const	0.2	
15	2	SPTF	exp	0.1		FCFS	const	0.35		FCFS	const	0.2	
16	3	FCFS	exp	0.1		FCFS	exp	0.3		FCFS	exp	0.2	
17	3	SPTF	exp	0.1		FCFS	exp	0.3		FCFS	exp	0.2	
18	3	SPTF PREEMP	exp	0.1		FCFS	exp	0.3		FCFS	exp	0.2	
19	3	SPTF PREEMP	exp	0.1		SPTF	exp	0.3		SPTF	exp	0.2	

TABLE 2

TIME AVERAGED OUTPUT VALUES FOR EXAMPLES STUDIED

Example	Center 1			Center 2			Center 3		
	Utiliz.	Mean Queue Length	St. Dev. Queue Length	Utiliz.	Mean Queue Length	St. Dev. Queue Length	Utiliz.	Mean Queue Length	St. Dev. Queue Length
1	0.93	7.64	4.63	0.93	7.35	4.63			
2	0.99	6.81	2.94	0.99	8.17	2.97			
3	0.78	7.93	5.97	0.75	7.06	5.97			
4	0.96	6.83	3.69	0.98	8.15	3.70			
5	0.74	5.06	5.34	0.90	9.94	5.34			
6	0.73	5.28	5.74	0.87	9.71	5.74			
7	0.75	3.60	4.20	0.94	11.40	4.21			
8	0.80	2.81	3.10	0.99	12.19	3.11			
9	0.37	0.47	0.83	0.99	10.47	3.53	0.90	4.06	3.47
10	0.37	0.56	0.96	0.98	8.85	3.98	0.91	5.59	3.93
11	0.74	1.78	1.84	0.99	9.15	3.53	0.91	4.07	3.22
12	0.73	2.43	2.70	0.96	7.49	4.24	0.90	5.08	3.83
13	0.74	1.71	1.93	0.98	8.15	3.97	0.93	5.14	3.74
14	0.70	2.81	3.44	0.96	8.04	4.22	0.86	4.14	3.75
15	0.76	1.55	1.30	0.99	8.15	3.63	0.93	5.30	3.52
16	0.55	1.23	1.60	0.45	0.76	1.12	0.52	1.03	1.49
17	0.54	1.08	1.43	0.38	0.66	1.08	0.49	0.90	1.28
18	0.51	0.86	1.12	0.38	0.58	0.94	0.46	0.81	1.14
19	0.47	0.76	1.06	0.35	0.50	0.82	0.43	0.72	1.05

To avoid aliasing, it was necessary to force a higher Nyquist frequency in some cases. We did this by lowering the simulation time which increased the sampling rate. The simulation times range from 256 to 512 time units.

We have used a process similar to the window closing procedure described in Chapter 3 to determine the smoothing parameters. A 0.10 tapered cosine data window and a 51 point moving average were used in all cases. Only one smoothed spectral estimate is presented for each time series.

B. Experimental Results - Structure One

The experiments having structure one illustrate the different types of output of the program and the striking variations in behavior possible even in very simple systems. The parameters to be specified for this structure are the scheduling method and service time distribution at each of the two service centers. This structure has a symmetry which simplifies the presentation of the output for these cases. These are closed systems with two service centers and 15 jobs. If there are n jobs at center one, there must be (15 - n) jobs at center two. Hence, the histograms for the two centers are mirror images of each other. Similarly, if a job leaves one service center, it must go to the other. That is, activity at the two centers always changes at the same instants in time. Hence, the periodic behavior of the two centers is similar and it is necessary to display the power spectrum of only one. The standard deviations of the two processes should be identical but, in fact, roundoff errors will cause slight differences.

In example 1 both service centers are identical using FCFS scheduling and an exponential service time distribution with mean 0.05. Figure 2 shows the outputs for this case. The histogram demonstrates a fairly uniform queue length distribution. In particular, the queue will be empty about one-fifteenth of the time which is reflected in the 0.93 utilization factor. The power spectrum shows that the low frequencies predominate with the trace falling off gradually up to about 4.0 cycles. The standard deviation is 4.63.

Example 2 is similar to example 1 with SPTF replacing FCFS as the scheduling method at both service centers. The effect of this change is significant as pointed out in Figure 3 by the histogram which has a triangular shape with the peak at 7-8 cycles. This system is rarely far out of balance and, when it is, the tendency is to restore balance, i.e., move back toward the peak. The standard deviation, 2.94-2.97, is lower than that of example 1. The queues are almost never empty so that the utilization factors are essentially 1.0. The power spectrum, Figure 3, shows that while the low frequencies still predominate, the spectrum falls off sharply, leveling off around 0.75 cycles.

The improvement of SPTF scheduling over FCFS can be explained in the following manner. The exponential service distribution is characterized by mostly small service times with an occasional service time much larger than the mean. FCFS scheduling will process a job with a large service time when its turn comes. This can cause the queue to lengthen because jobs are still arriving from the other center. SPTF scheduling will ignore these large service times until there are no more jobs with smaller service times. Thus, there will be many fewer in-

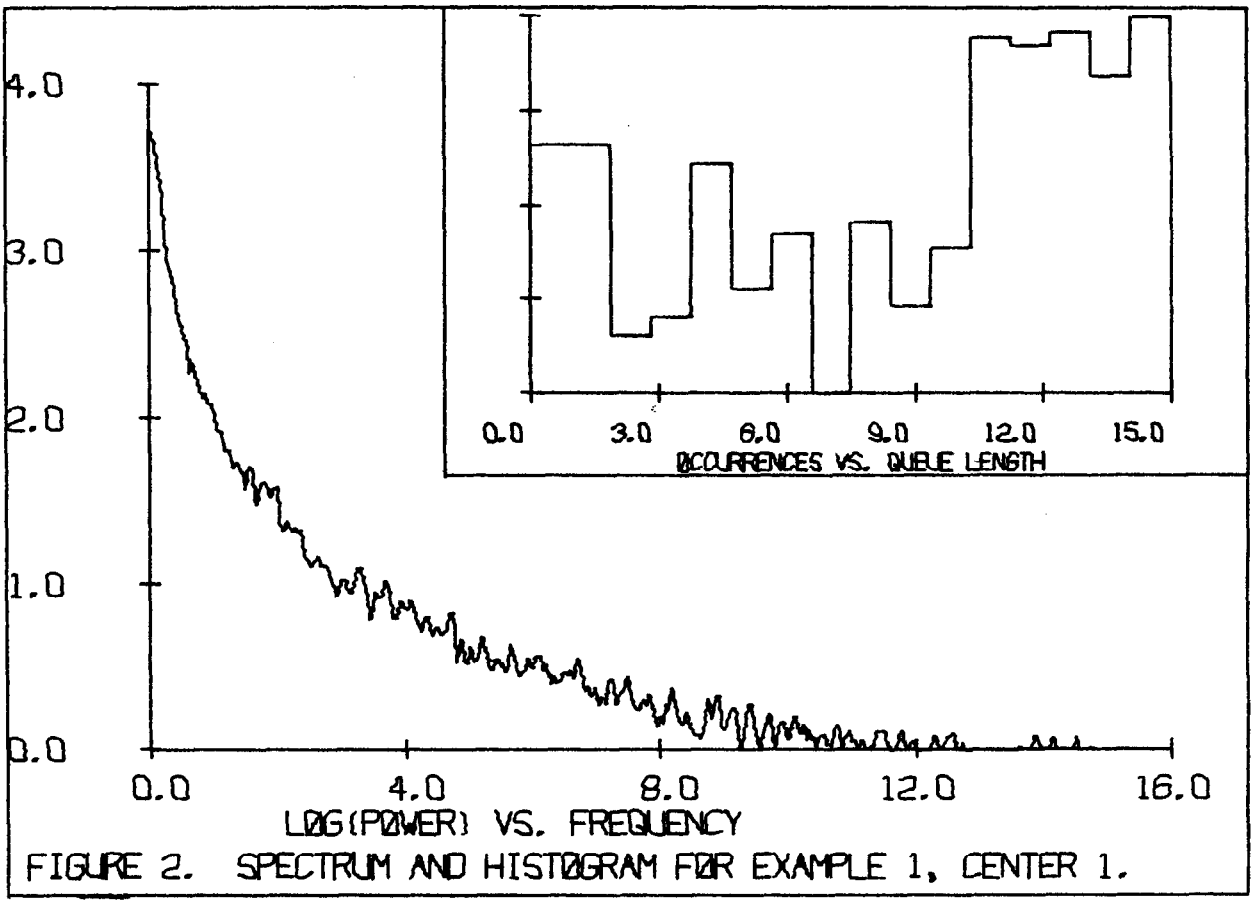


FIGURE 2. SPECTRUM AND HISTØGRAM FØR EXAMPLE 1, CENTER 1.

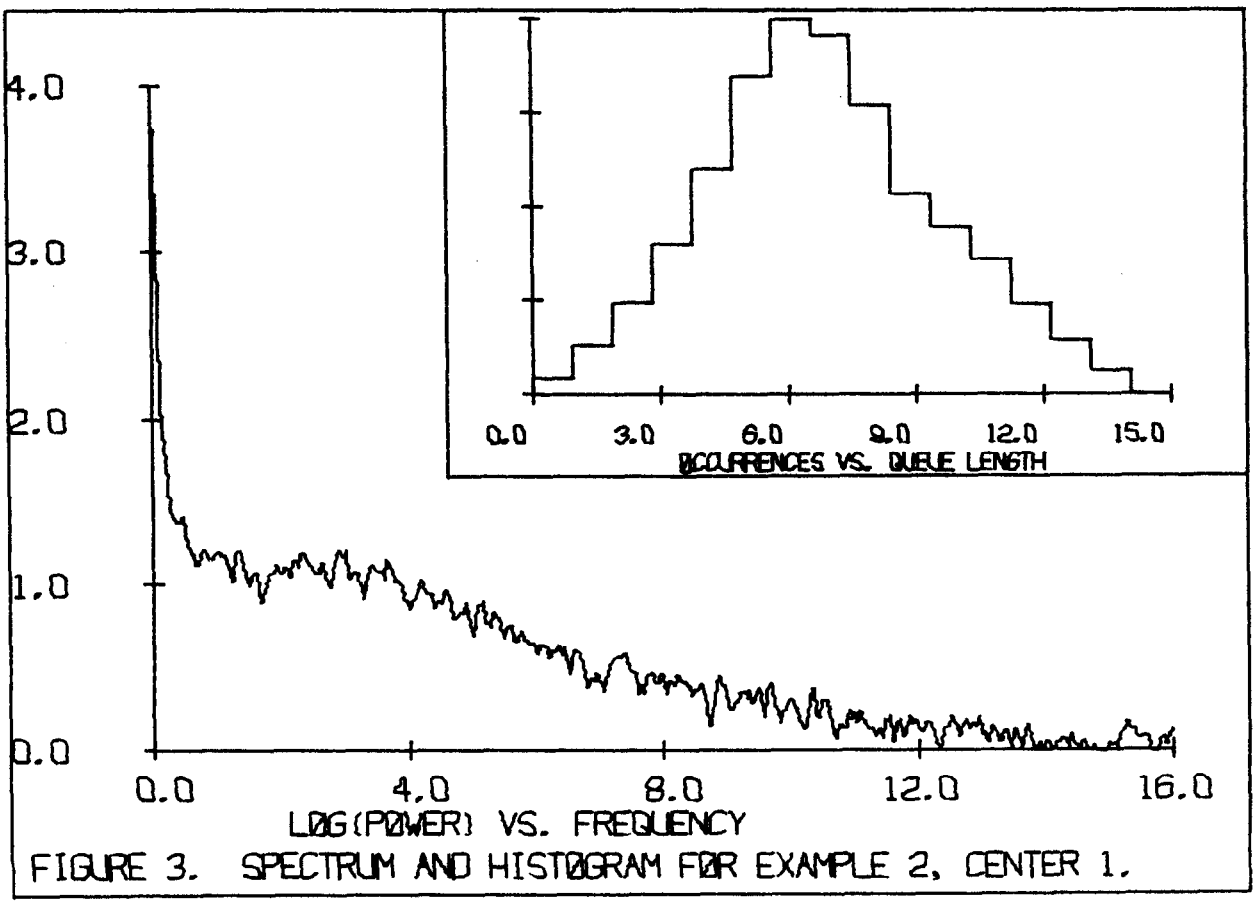


FIGURE 3. SPECTRUM AND HISTØGRAM FØR EXAMPLE 2, CENTER 1.

stances of congestion at one service center and idleness at the other. Less improvement might be expected with SPTF if the service distribution does not generate occasional large service times.

Example 3 is similar to example 1 with the exponential distribution replaced by a hyperexponential distribution with parameter 0.05 and mean 0.05. This distribution has a standard deviation which is about three times that of an exponential distribution with the same mean. This implies getting larger variates more frequently. The result is that the problem of congestion caused by processing jobs with larger service times is aggravated in this case. The histogram and power spectrum appear in Figure 4. The histogram is slightly concave with very steep gradients near queue lengths of 0 and 15. The queue is either full or empty about half of the time and the system tends to gravitate to one of these extremes. This is reflected by utilization factors of about 0.76 and the rather high standard deviation, 5.97. None of this behavior is particularly demonstrated by the power spectrum. It falls off slightly more slowly than that of example 1, but the shape is practically identical.

Example 4 substitutes SPTF scheduling for FCFS in example 3. The effect of this change is even more dramatic than it was in example 2. The utilization factor is about 0.97, 0.2 above that for FCFS and the standard deviation has diminished to 3.69. The histogram, Figure 5, still exhibits a triangular shape although the queue size reaches its extremes more often than in example 2. This can be explained by the greater abundance of larger service times, causing some problem with congestion even with SPTF scheduling. The power spectrum, Figure 5,

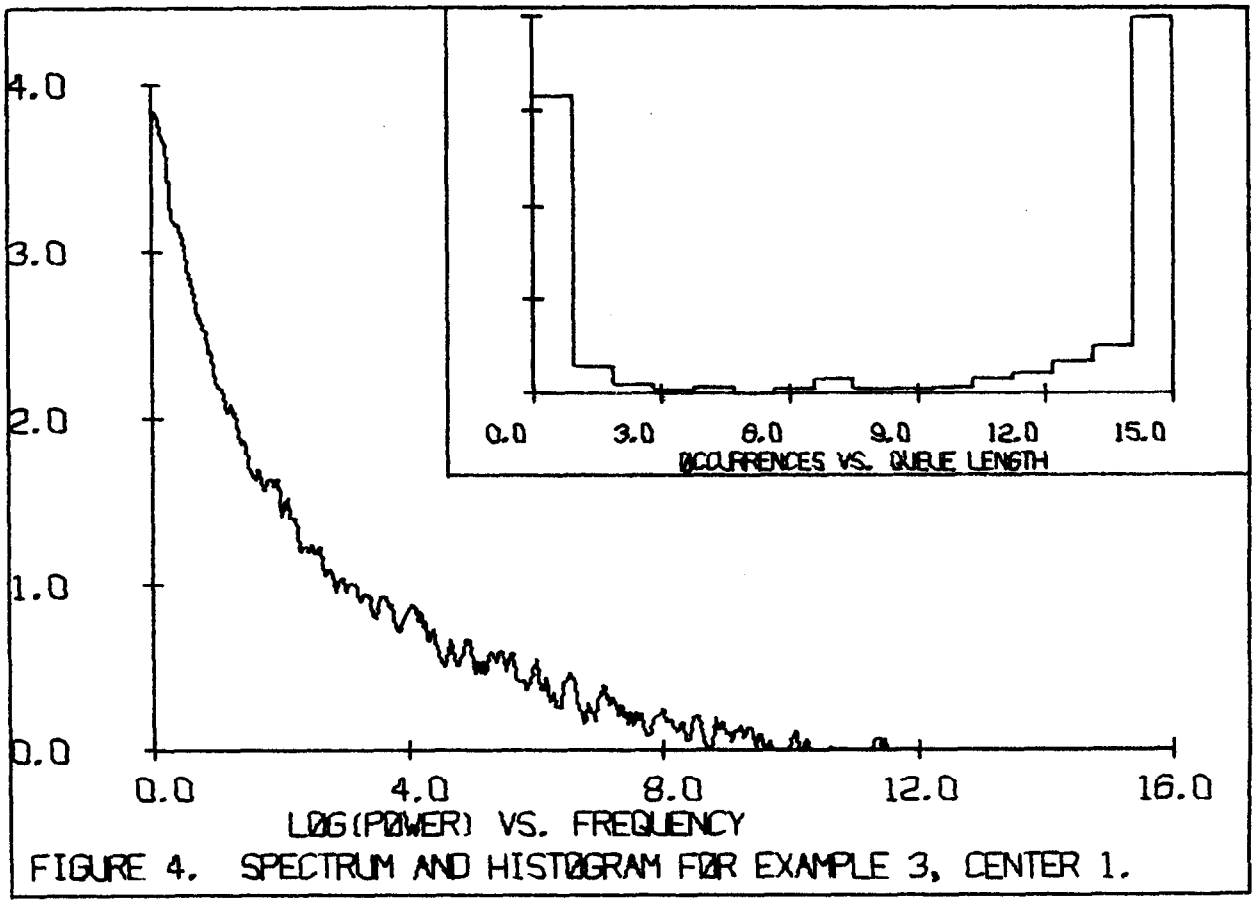


FIGURE 4. SPECTRUM AND HISTØGRAM FØR EXAMPLE 3, CENTER 1.

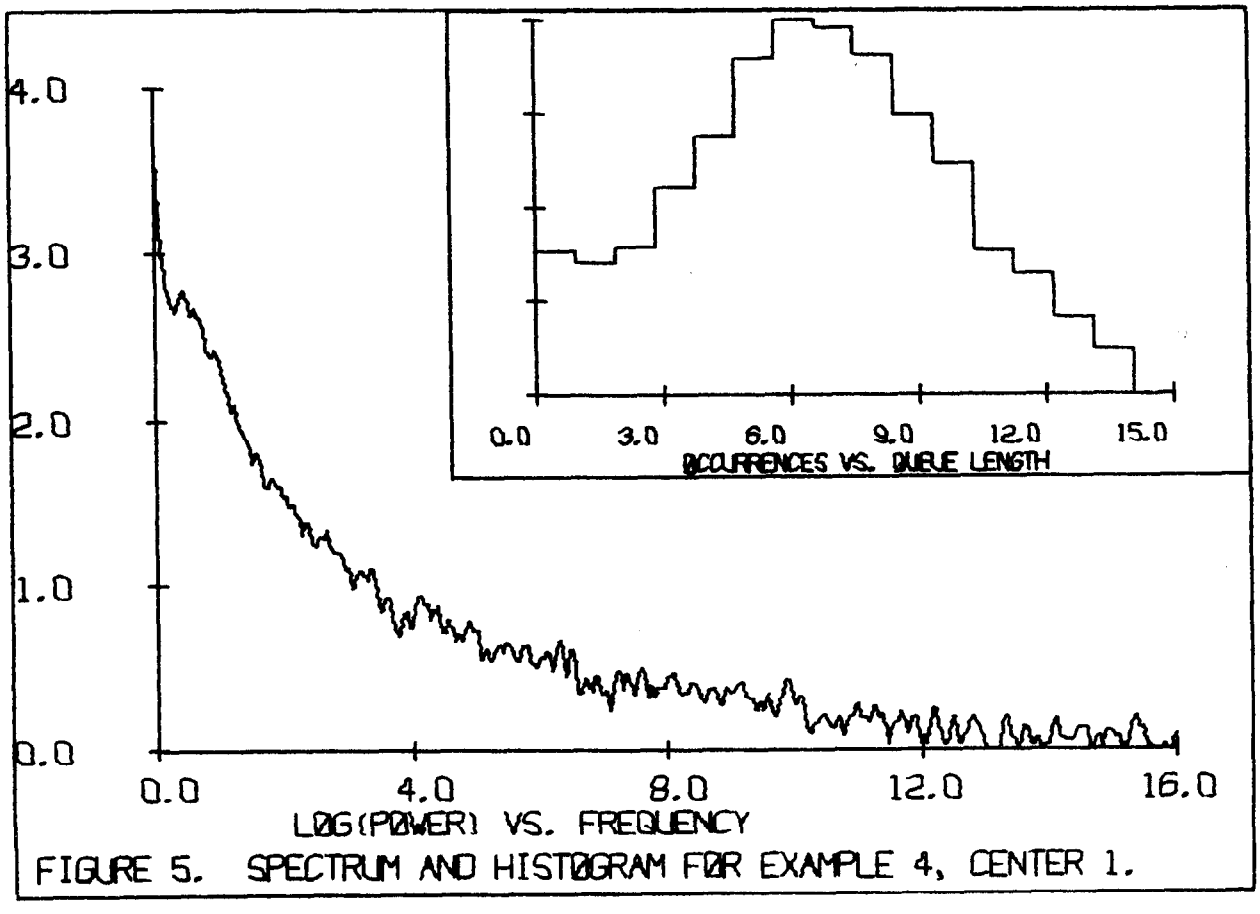


FIGURE 5. SPECTRUM AND HISTØGRAM FØR EXAMPLE 4, CENTER 1.

seems to resemble that of example 3 more than that of example 2, falling off almost linearly over a range of about 2 cycles. The hyperexponential distribution has caused a significant widening of the low frequency band, contributing to the variance of the process.

In the next series of experiments, we investigate the effects of scheduling when one cervice center has a constant service time. For examples 5 through 8 jobs arriving at service center two will be scheduled FCFS and assigned a service time of 0.25. Service center one is characterized by a hyperexponential distribution with mean 0.200 and parameter 0.05. These systems will not have balanced activity like the previous systems. The utilization factors of the two centers will be different. However, our reasons for presenting the histogram and power spectrum of only one service center are still valid.

FCFS at center one completes the specification of example 5. The outputs for this case are shown in Figure 6. The histogram for this case has the same U-shape as that of example 3. The large number of small queue sizes is simply the result of the differing means of the service distributions. The rather large number of queue sizes of 15 achieved indicates the presence of the previously mentioned congestion problem aggravated by the hyperexponential distribution. The power spectrum now has the peculiar characteristics caused by the constant service time distribution. The spectrum has a set of peaks of diminishing size centered at 4, 8, and 12 cycles. These are the fundamental and second and third harmonic frequencies corresponding to the service time of 0.25. The spectrum is still dominated by the low frequencies leveling off around 2.5-3.0 cycles. The utilization factors for the

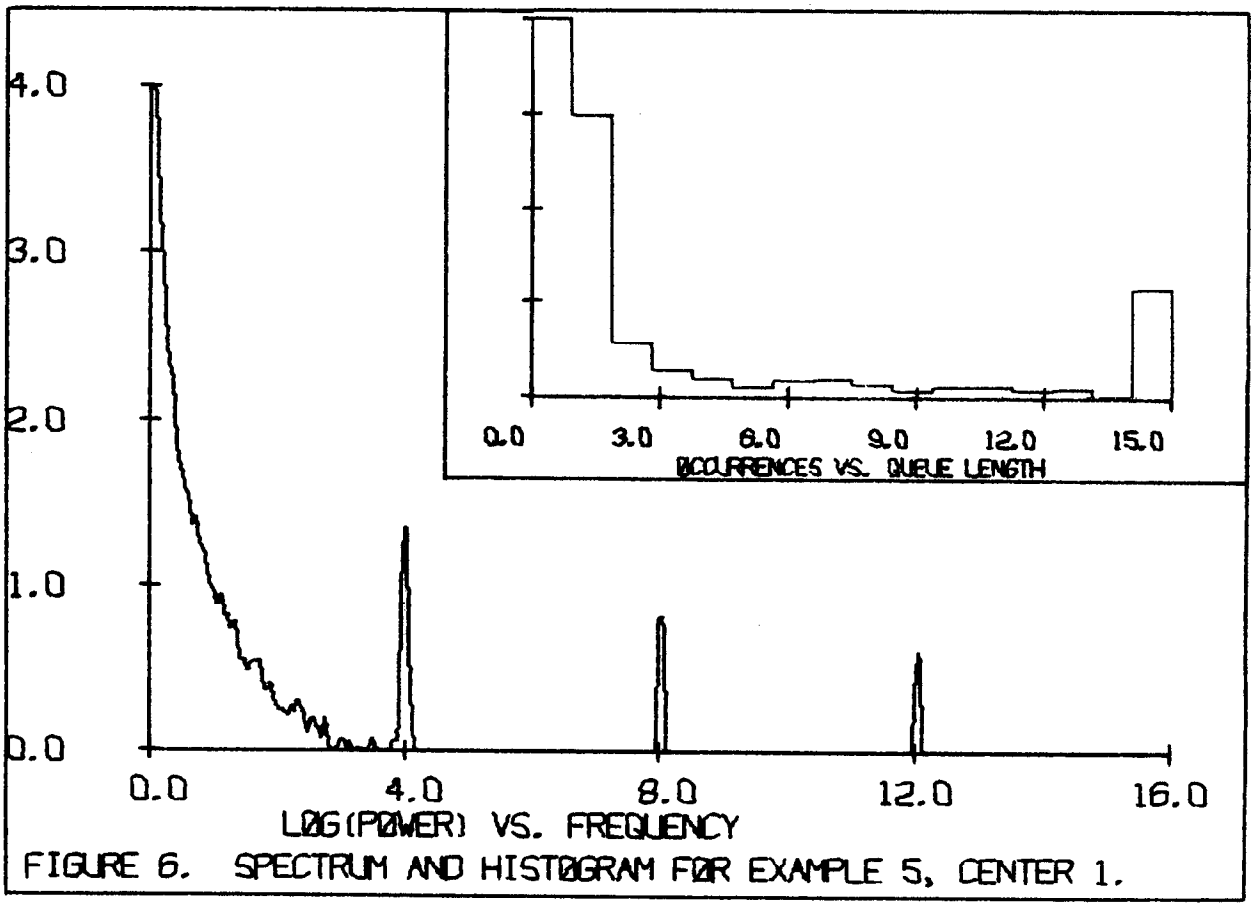


FIGURE 6. SPECTRUM AND HISTØGRAM FØR EXAMPLE 5, CENTER 1.

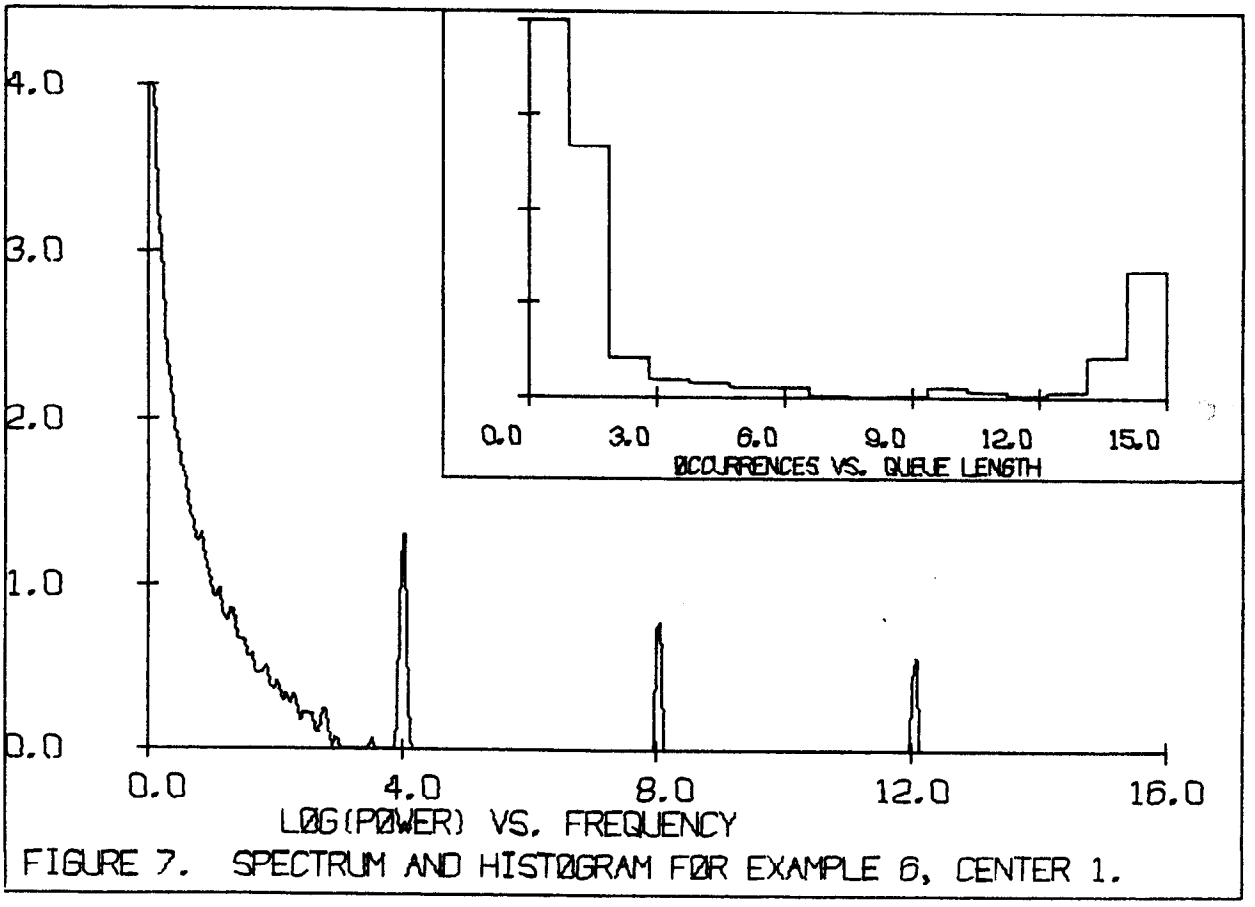


FIGURE 7. SPECTRUM AND HISTØGRAM FØR EXAMPLE 6, CENTER 1.

two service centers are 0.74 and 0.90 respectively. The standard deviation is 5.34.

Example 6 differs from example 5 only in that LPTF scheduling is used at center one. The outputs appear in Figure 7. The power spectrum is practically identical to that of example 5.. The histogram has the same shape with very steep sides. Queue sizes of 0, 1, 14, and 15 are more prevalent than was true in example 5. The larger queue sizes are the result of LPTF scheduling exaggerating the congestion caused by relatively large service times. It may strike one as somewhat surprising that the deleterious effect of LPTF was not more noticeable. The utilization factors at the two centers dropped only slightly to 0.73 and 0.87, while the standard deviation increased to 5.74.

Example 7 differs from example 5 in that SPTF scheduling is used at center one. The resulting power spectrum, Figure 8, has the same overall appearance as that of example 5. Some of the power seems to have been moved from the lowest frequencies to the rest of the spectrum. The histogram, Figure 8, now has a severely distorted U-shape. There is a much greater representation of the smaller queue sizes. The congesting effect of large service times at center one has not disappeared entirely but it is considerably less pronounced. These results indicate that SPTF scheduling at center one has reduced the effect of the hyperexponential distribution on the system and increased the effect of activity at the other service center. It should be noted that the beneficial effect of SPTF scheduling on system performance occurs at the other center in this case. The utilization factor of 0.75 at center one nearly equals that obtained with FCFS. However, the utilization at center two has been raised to 0.94. The standard deviation decreased to 4.20.

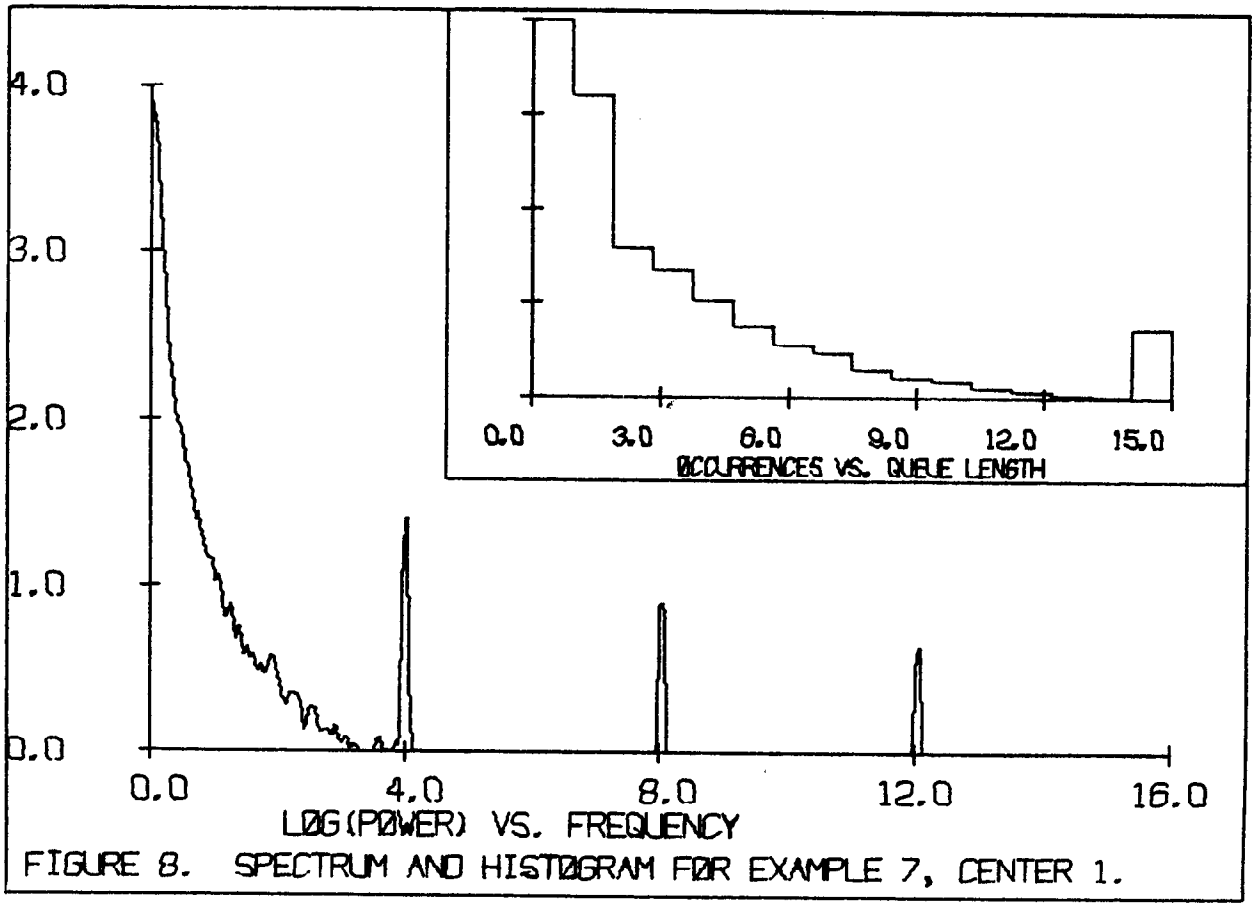


FIGURE 8. SPECTRUM AND HISTØGRAM FØR EXAMPLE 7, CENTER 1.

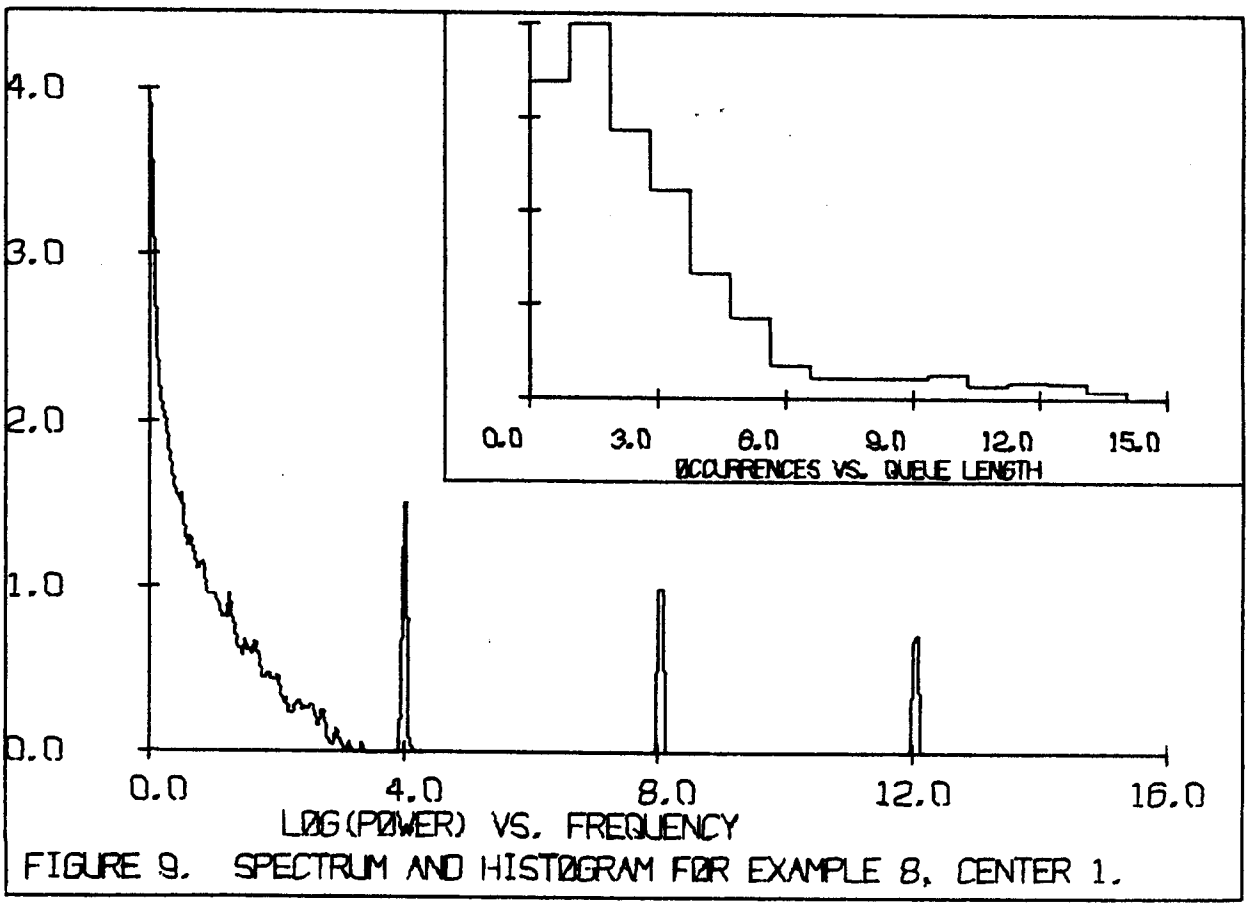


FIGURE 9. SPECTRUM AND HISTØGRAM FØR EXAMPLE 8, CENTER 1.

Example 8 provides an even more effective way of dealing with large service times than did SPIF scheduling. In this case we use RR scheduling with a time slice of 0.06, which will allow short jobs to be completely processed while parts of longer jobs are serviced. The outputs are shown in Figure 9. A look at the power spectrum reveals that power has been removed from the whole frequency range below 4.0 cycles as compared to examples 5-7. There is noticeably more power in the spikes generated by the constant service time distribution. The histogram no longer exhibits the U-shape it had in previous examples. It is almost a monotone decreasing function of queue size. The number of occurrences of queue sizes 14 or 15 is practically negligible. These results demonstrate that RR scheduling is an effective way of dealing with congestion at a service center. The utilization factors for the two centers are 0.80 and 0.99 respectively. The standard deviation is 3.10. In a sense this is near optimal. Since the ratio of the mean service times at the two centers is 0.8, if it were possible to keep center two busy all of the time, the expected utilization at center one would be 0.80.

C. Experimental Results - Structure Two

The experiments having structure two illustrate the somewhat more complicated behavior of a three component system. Jobs are routed from center one to center two with probability 0.375 and to center three with probability 0.625. Jobs always return to center one from either of centers two or three. The mean service times at centers two and three are 0.350 and 0.200 respectively. The remaining parameters to be specified for these examples are the scheduling method and type of service

distribution at each of the three service centers. Because this structure does not have the symmetry displayed by structure one, it is necessary to present sets of output for all three service centers.

Examples 9 through 12 all use FCFS scheduling and an exponential service time distribution at center one. Centers two and three have constant service times.

In example 9 the mean service time at center one is 0.050 and FCFS scheduling is used at centers two and three. The histograms and power spectra are shown in Figures 10 through 12. There are seldom more than two jobs at center one because of its relatively low mean service time. Centers two and three have nearly linear histograms with positive and negative slopes respectively. The power spectrum for center one shows relatively little power in the low frequencies compared with earlier examples. The interesting features are the high power spikes caused by the constant service times. There is measureable power up to the fourth harmonic corresponding to the 0.200 service time and up to the fifth harmonic corresponding to the 0.350 service time. The spectrum for center two exhibits quite a bit of power in the low frequency range and a set of spikes with the fundamental frequency corresponding to the to the 0.350 service time. There are no spikes corresponding to the 0.200 service time. The spectrum for center three is similar in that there is significant low frequency power and a set of spikes corresponding, in this case, to the 0.200 service time. However, there is also a single spike near 2.9 cycles representing the fundamental of the 0.350 constant service time. The principal reason for power leakage from center two to center three is that center one does not delay

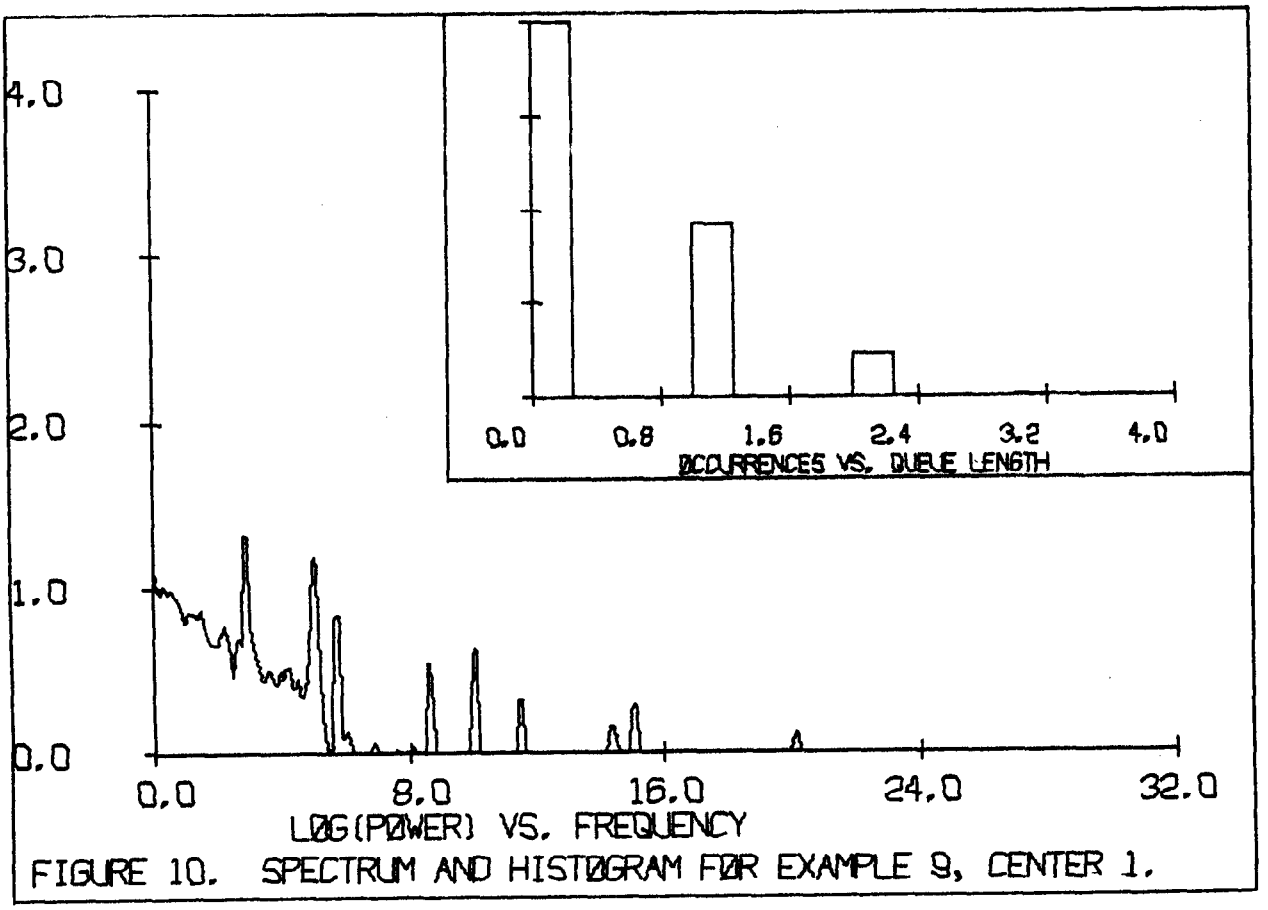


FIGURE 10. SPECTRUM AND HISTØGRAM FØR EXAMPLE 9, CENTER 1.

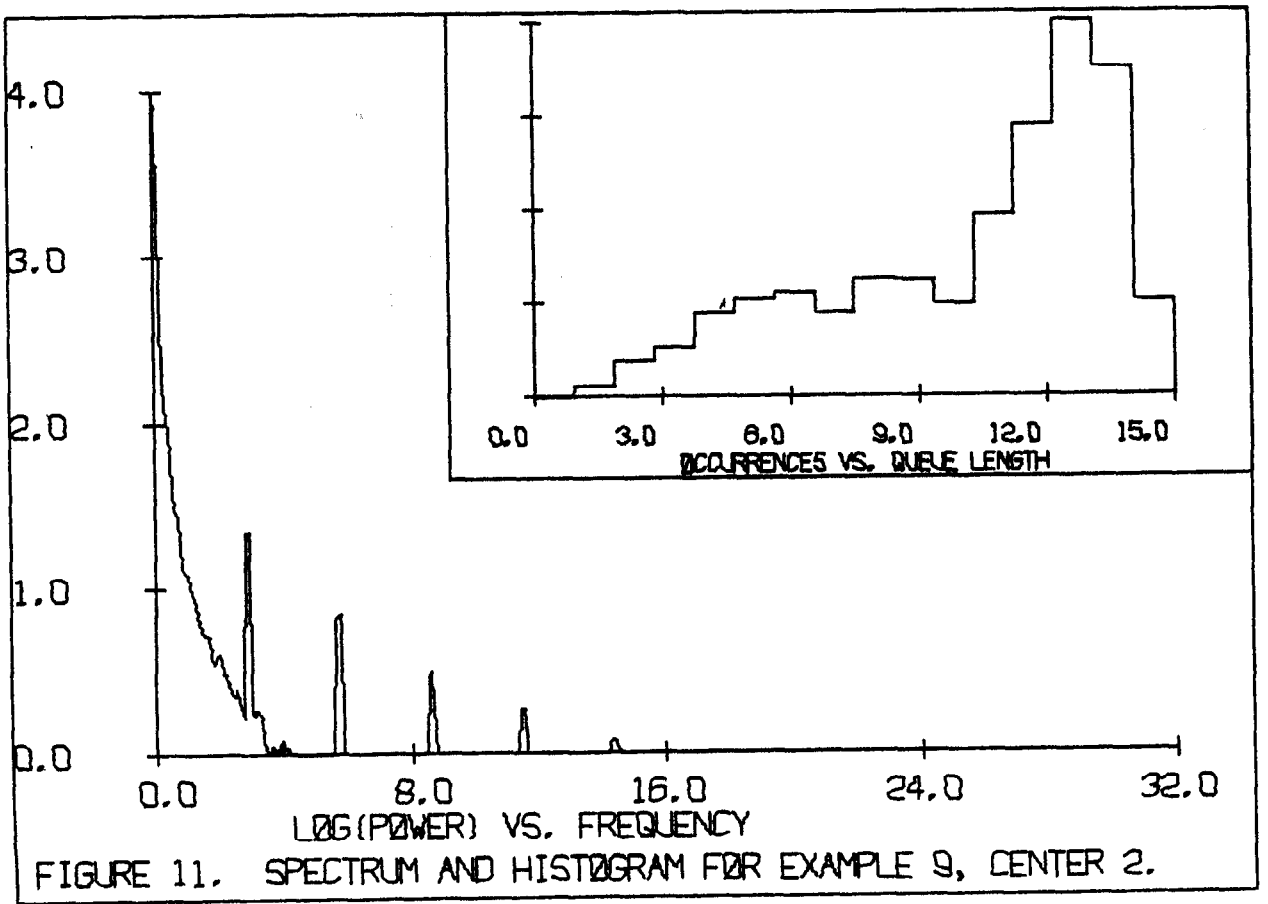


FIGURE 11. SPECTRUM AND HISTØGRAM FØR EXAMPLE 9, CENTER 2.

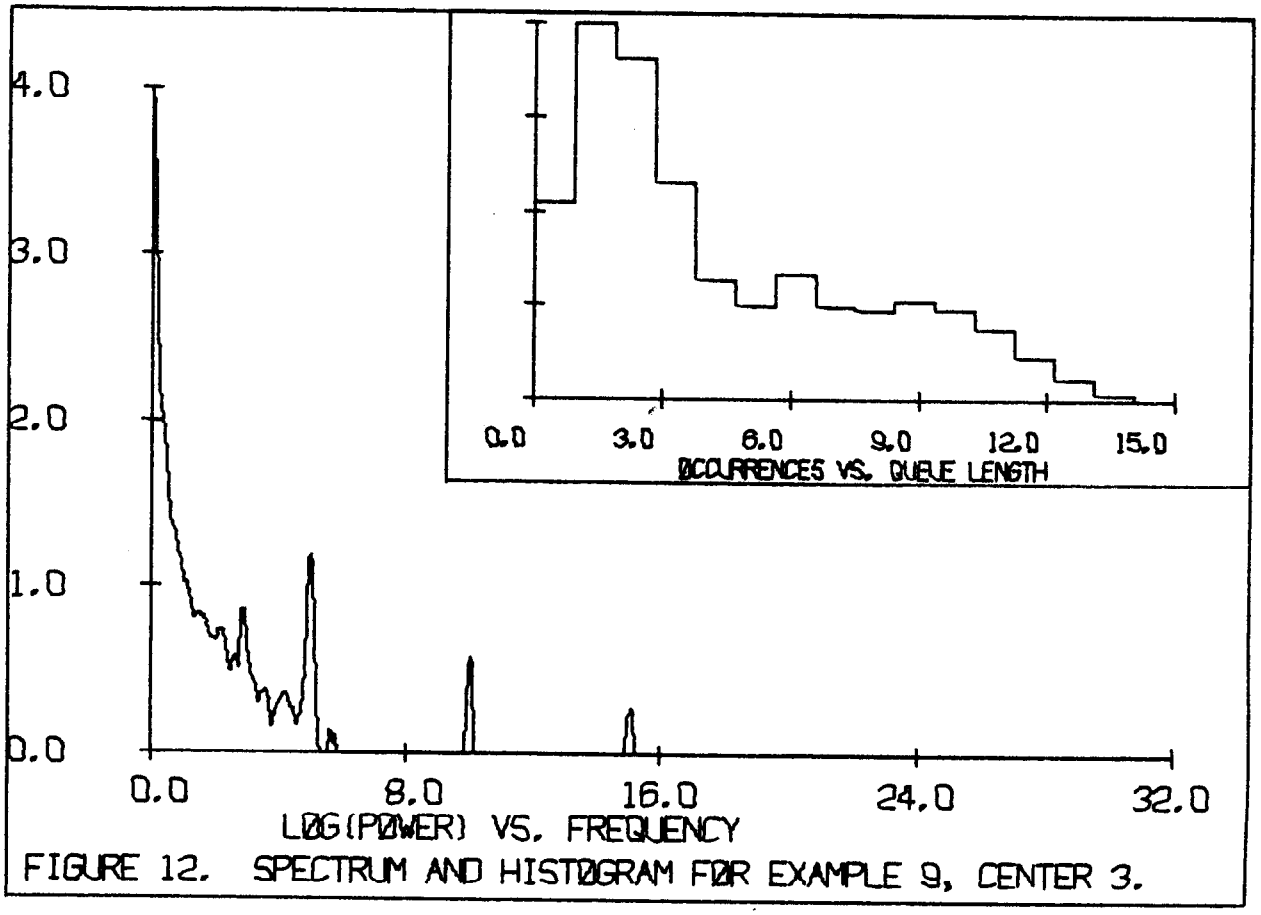


FIGURE 12. SPECTRUM AND HISTØGRAM FØR EXAMPLE 9, CENTER 3.

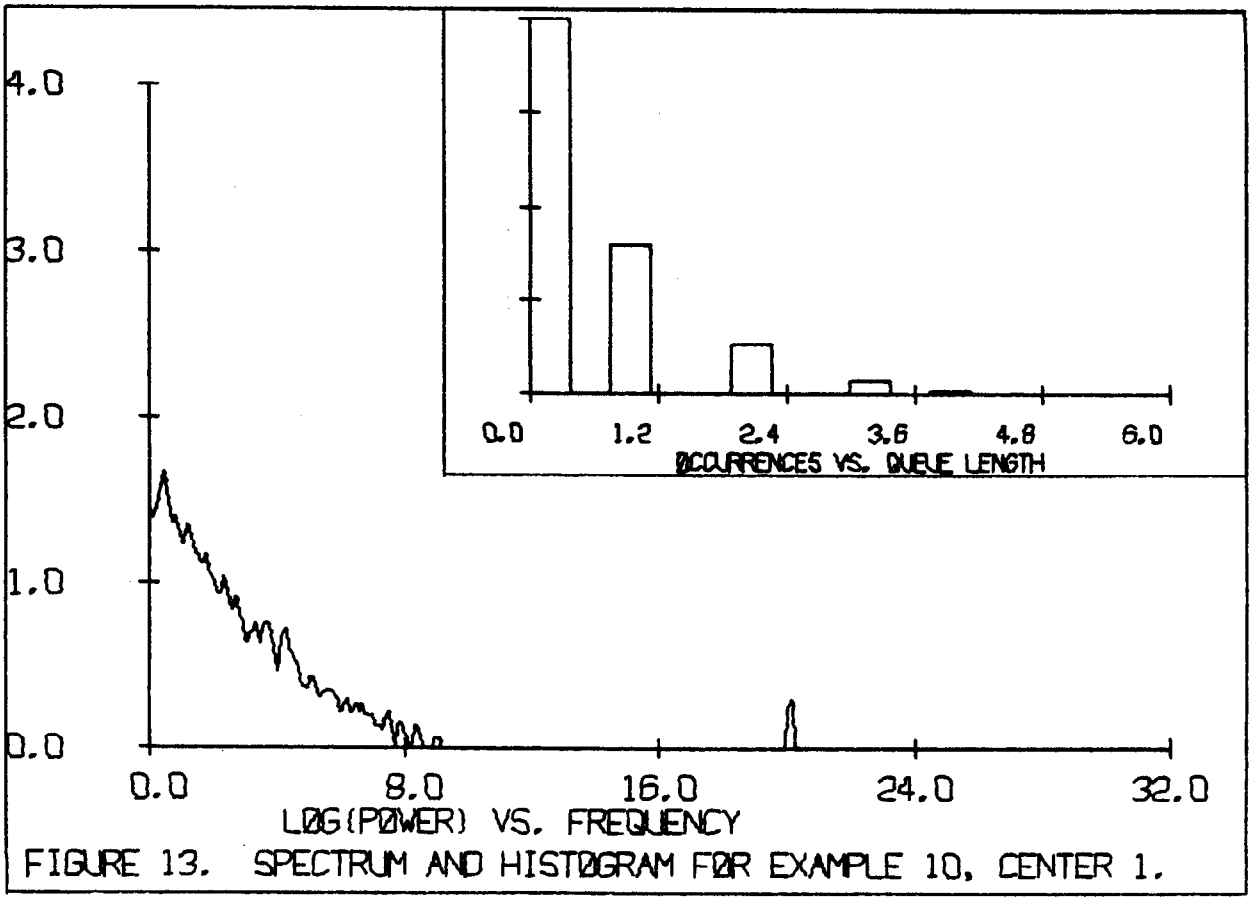


FIGURE 13. SPECTRUM AND HISTØGRAM FØR EXAMPLE 10, CENTER 1.

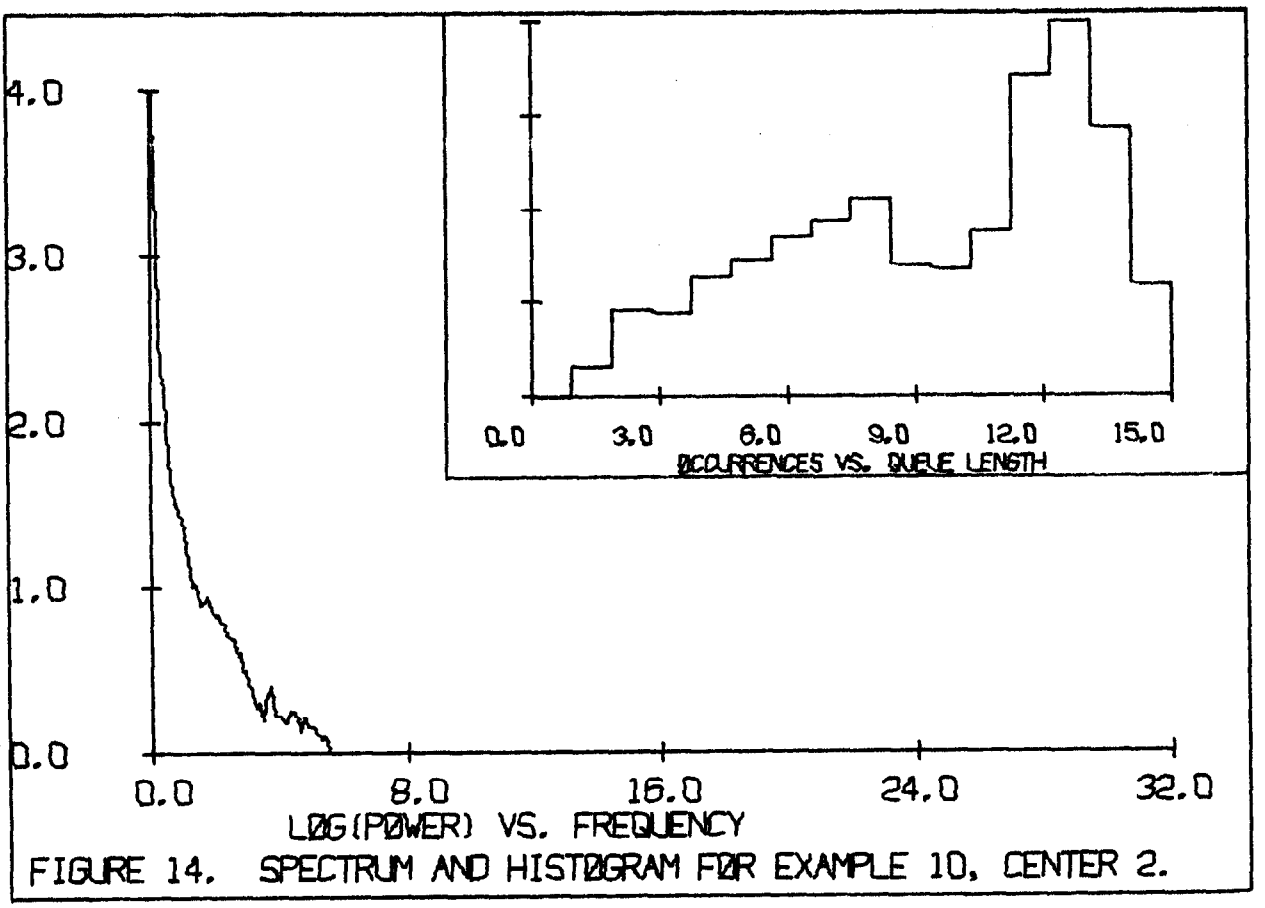


FIGURE 14. SPECTRUM AND HISTØGRAM FØR EXAMPLE 10, CENTER 2.

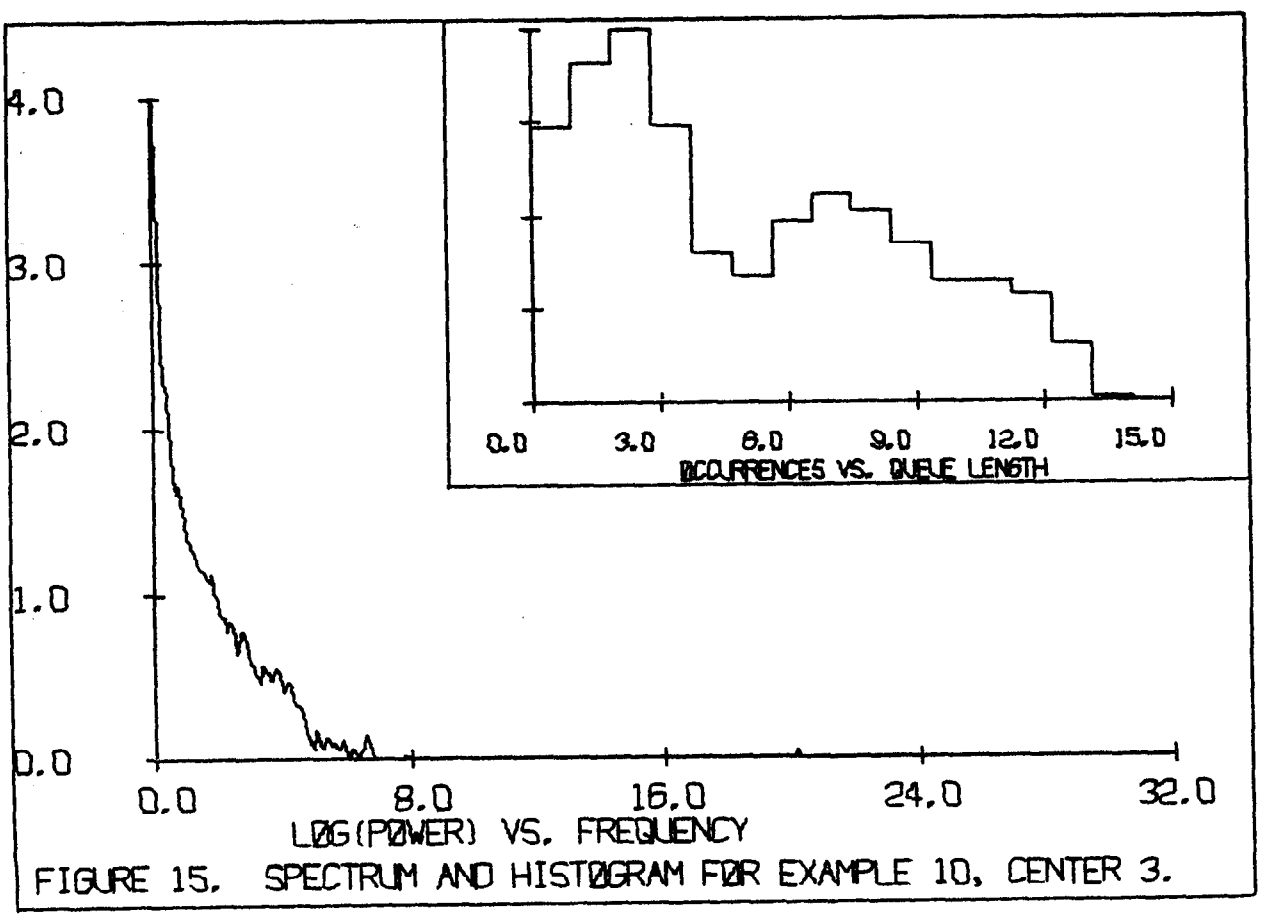


FIGURE 15. SPECTRUM AND HISTØGRAM FØR EXAMPLE 10, CENTER 3.

appreciably the flow of jobs in that direction. The mean service time at center one is small compared to the other centers, especially center two. Hence, a job passing from center two to center three experiences proportionately less delay than a job passing the other way. Center one, then, acts as a low pass filter allowing the low frequency (2.9 cycles) spike to pass from center two to center three, but not the higher frequency (5.0 cycles) spike the other way. Note that the second harmonic (5.8 cycles) did not show up in the spectrum for center three. The utilization factors are 0.37, 0.99, and 0.90 respectively. The standard deviations are 0.83, 3.53, and 3.47 respectively.

From the results of the two service center case we try RR scheduling to reduce the effect of the constant service times. Example 10 differs from example 9 in that RR scheduling with a 0.05 time slice is used at centers two and three. The histograms and power spectra are shown in Figures 13 through 15. Table 2 shows that the overall performance was hardly changed from that of example 9. The utilization factors are essentially identical and the standard deviations are slightly higher. However, the power spectra are noticeably different. All of the spikes from example 9 have disappeared. In their place there is a new peak at 20 cycles which is significant only in the spectrum for center one. Also, the amount of power in the lower frequencies has increased slightly for all centers. In this case, it appears that RR scheduling has transferred power from the relatively high frequency spikes into the lower frequencies. It has also created a spike at a frequency equal to the inverse of the time slice. If the service time were not an integral multiple of the time slice, we should

expect the spike to occur at a frequency corresponding to the service time remaining in the last time slice.

In the previous two examples the flow of jobs to centers two and three was not slowed particularly by center one. The next two examples duplicate examples 9 and 10, except that the mean service time at center one is changed to 0.100.

Example 11 uses FCFS scheduling at centers two and three. Figures 16 through 18 contain the histograms and spectra. A look at Table 2 shows that doubling the mean service time at center one has increased the utilization factor to 0.74 and the average queue length to 1.78 as compared with example 9. The mean queue length at center two has also increased at the expense of center three. The histograms reflect this shift of job load although the shapes are generally the same as in example 9. The power spectra all have the same spikes as in example 9 and more power in the lower frequencies, particularly in the case of service center one. In this example, as opposed to example 9, there is no appreciable power leakage in either direction between centers two and three. Increasing the mean of the exponential distribution at center one has effectively blocked the transmission of all high frequency spikes between these two components. In this example the increased power in the lower frequencies is not associated with performance degradation. The utilization of center one has doubled and that of center three has increased slightly because center one is kept busier.

Example 12 is similar to example 11 but uses RR scheduling with a time slice of 0.05 at centers two and three. Figures 19 through 21 show the resulting histograms and power spectra. Table 2 shows that

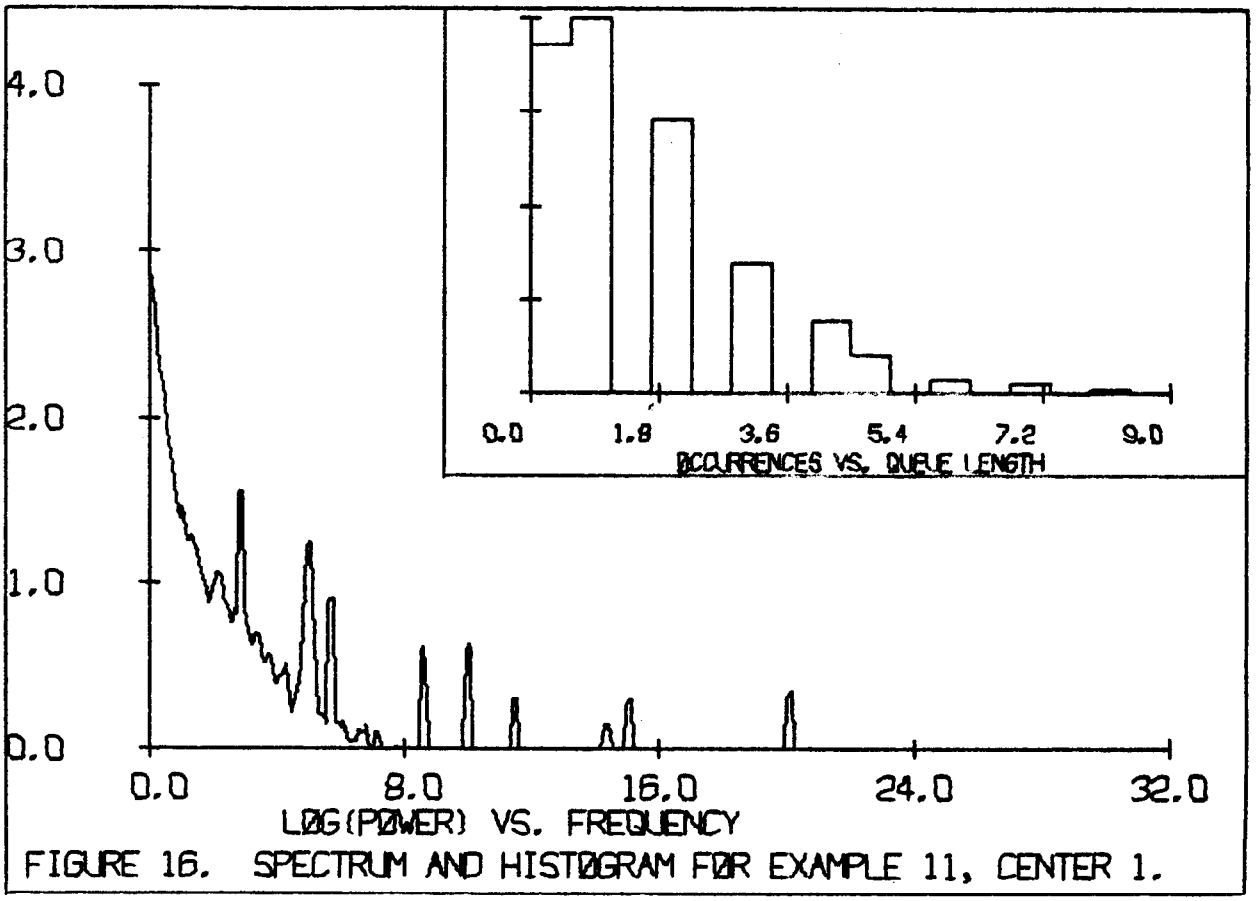


FIGURE 16. SPECTRUM AND HISTØGRAM FØR EXAMPLE 11, CENTER 1.

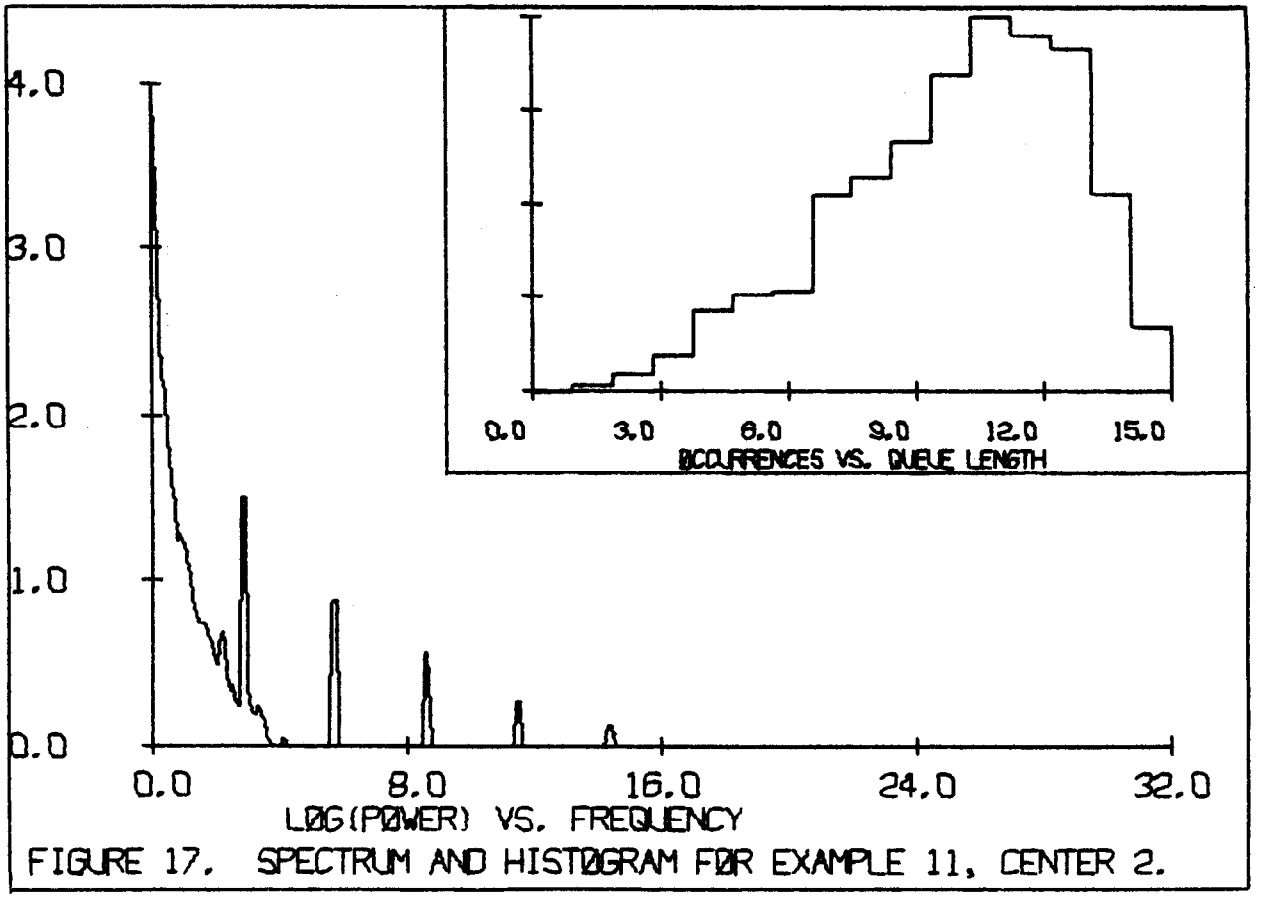


FIGURE 17. SPECTRUM AND HISTØGRAM FØR EXAMPLE 11, CENTER 2.

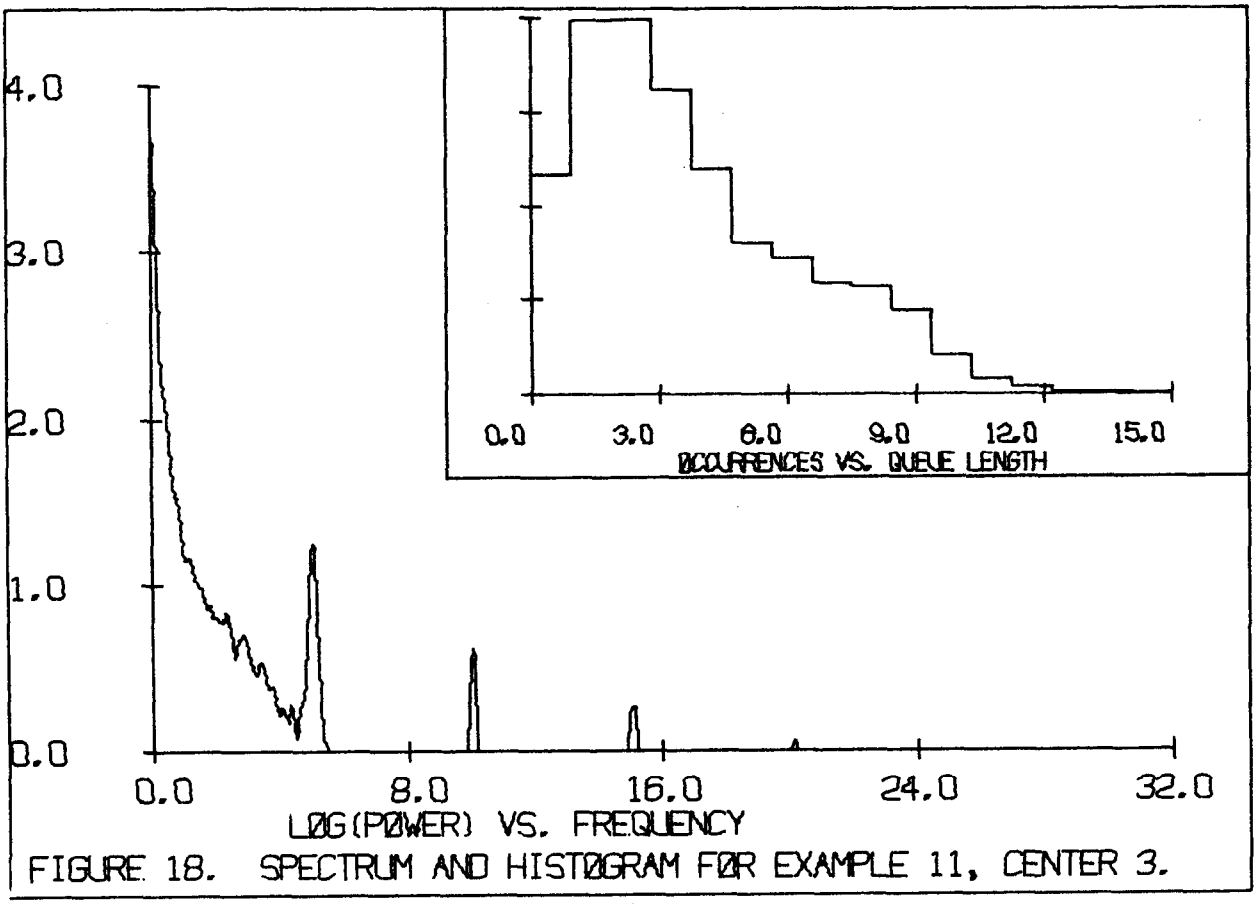


FIGURE 18. SPECTRUM AND HISTØGRAM FØR EXAMPLE 11, CENTER 3.

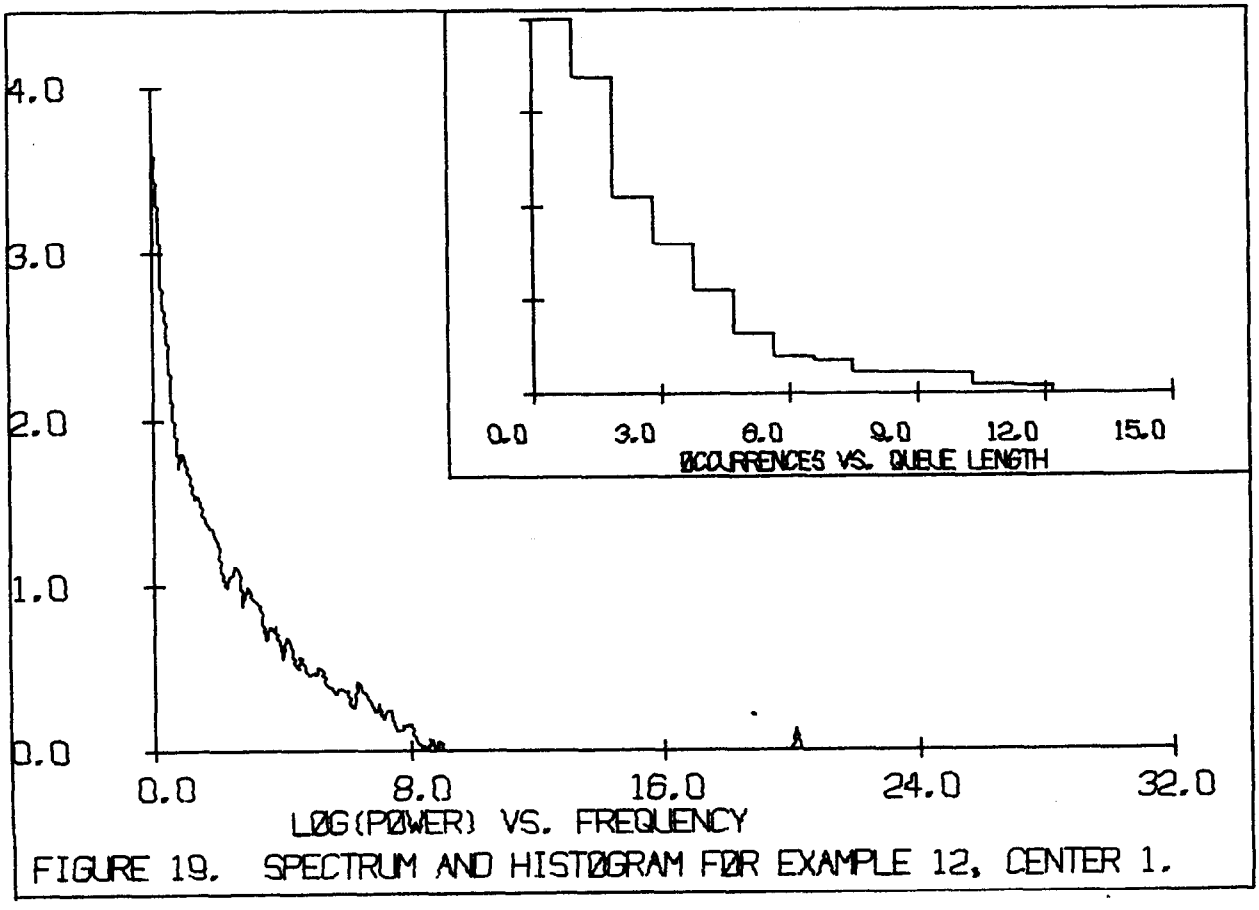


FIGURE 19. SPECTRUM AND HISTØGRAM FØR EXAMPLE 12, CENTER 1.

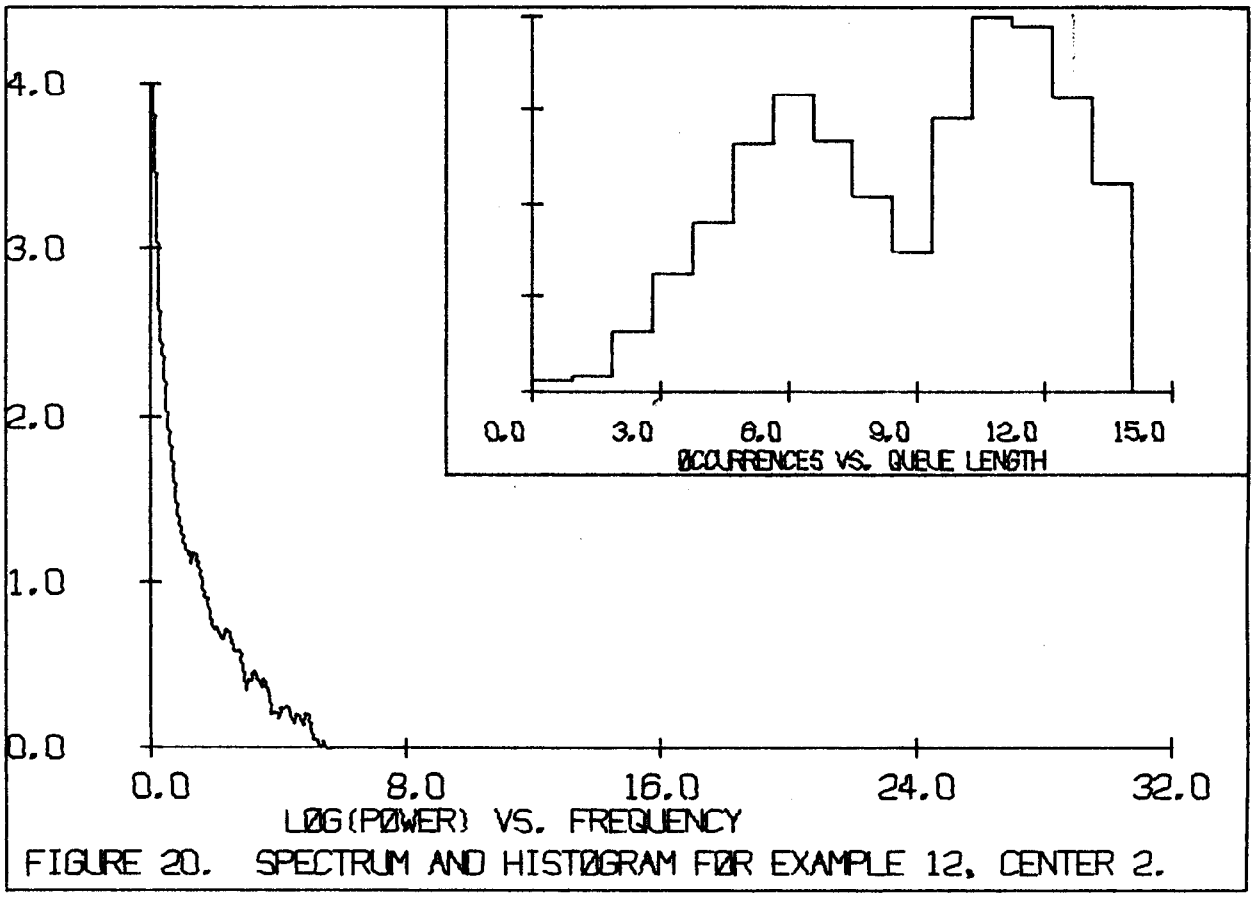


FIGURE 20. SPECTRUM AND HISTØGRAM FØR EXAMPLE 12, CENTER 2.

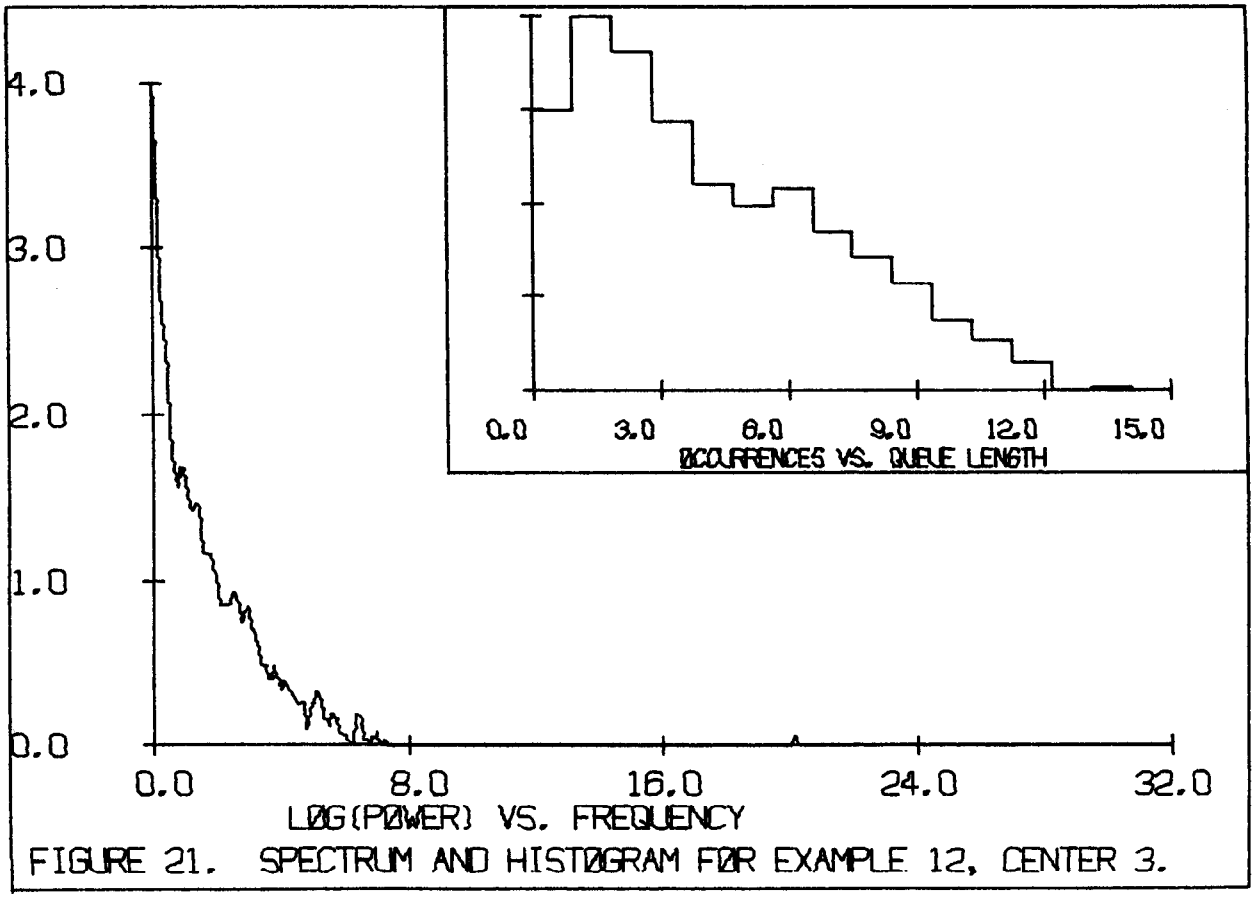


FIGURE 21. SPECTRUM AND HISTØGRAM FØR EXAMPLE 12, CENTER 3.

the utilization factors are 0.01 to 0.03 lower than in example 11, despite the fact that the mean queue lengths increased at centers one and three. We note that the standard deviations have increased in all cases. The histograms are not much different from those of example 11, reflecting only the shifts in mean queue lengths. The power spectra show a marked difference. As was the case in example 10, the spikes have completely disappeared, including the spike at 20 cycles caused by the time slice of 0.05. The processes are now dominated by 0-2 cycle fluctuations. The power in this range is significantly higher than that in example 11. As opposed to example 11, in this example increased low frequency power is associated with performance degradation.

In example 13 we investigate changes in system behavior when the service distribution at centers two and three is given some non-zero variance. This example is similar to example 11 except that the service distribution at centers two and three is hypoexponential with parameter 100.0 and mean as before. These distributions have standard deviations which are about one-tenth that of exponential distributions with the same means. The histograms and power spectra are shown in Figures 22 through 24. A comparison of values in Table 2 shows that the utilization factors are within 0.02 of the values in example 11. The mean queue lengths and standard deviations also have changed only slightly. A look at the histograms shows that they differ little in form from those in example 11. The spectra indicate that the power distribution in the lowest frequencies has not changed much. However, the spikes are much wider and shorter than those in example 11. Also, the harmonics have disappeared although the second harmonic (5.8 cycles) from center

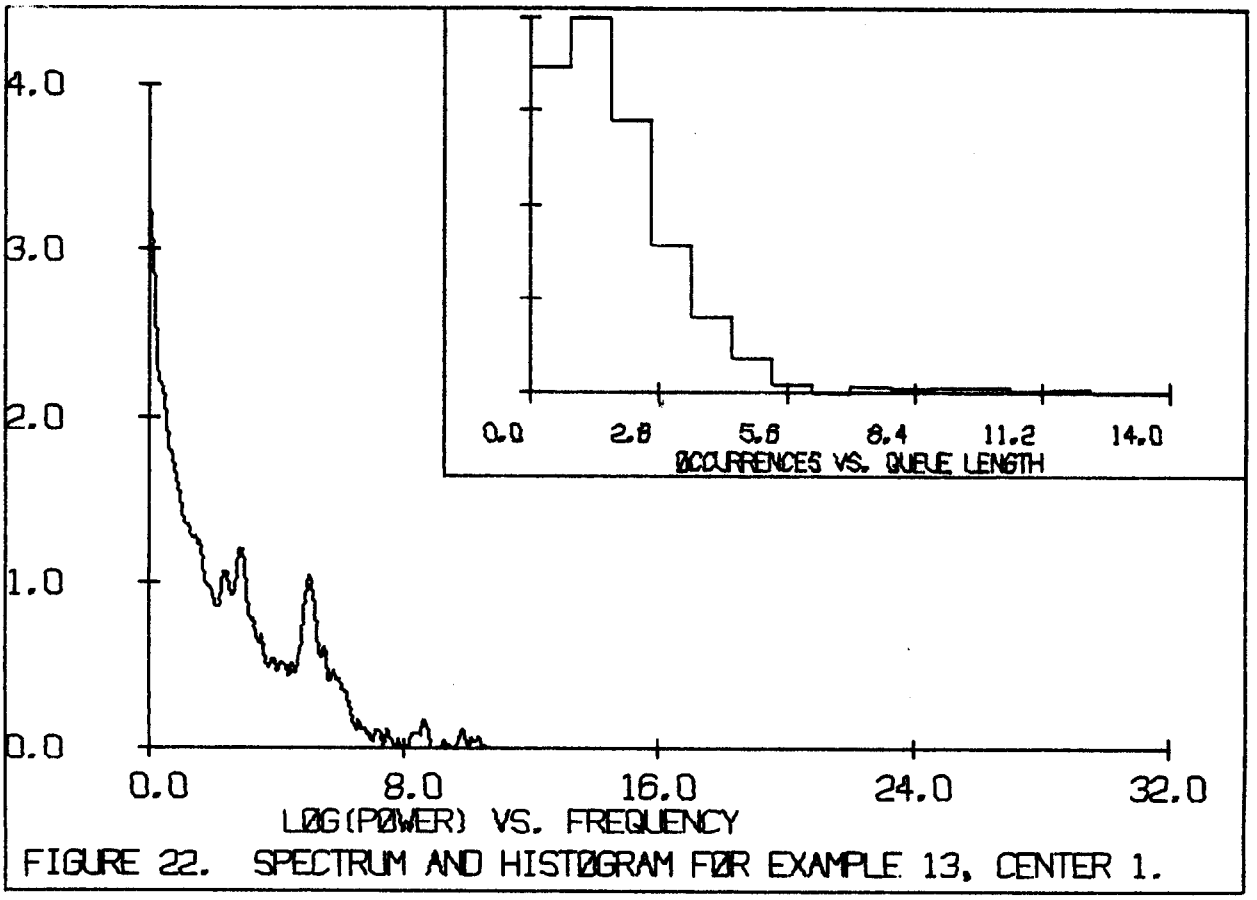


FIGURE 22. SPECTRUM AND HISTØGRAM FØR EXAMPLE 13, CENTER 1.

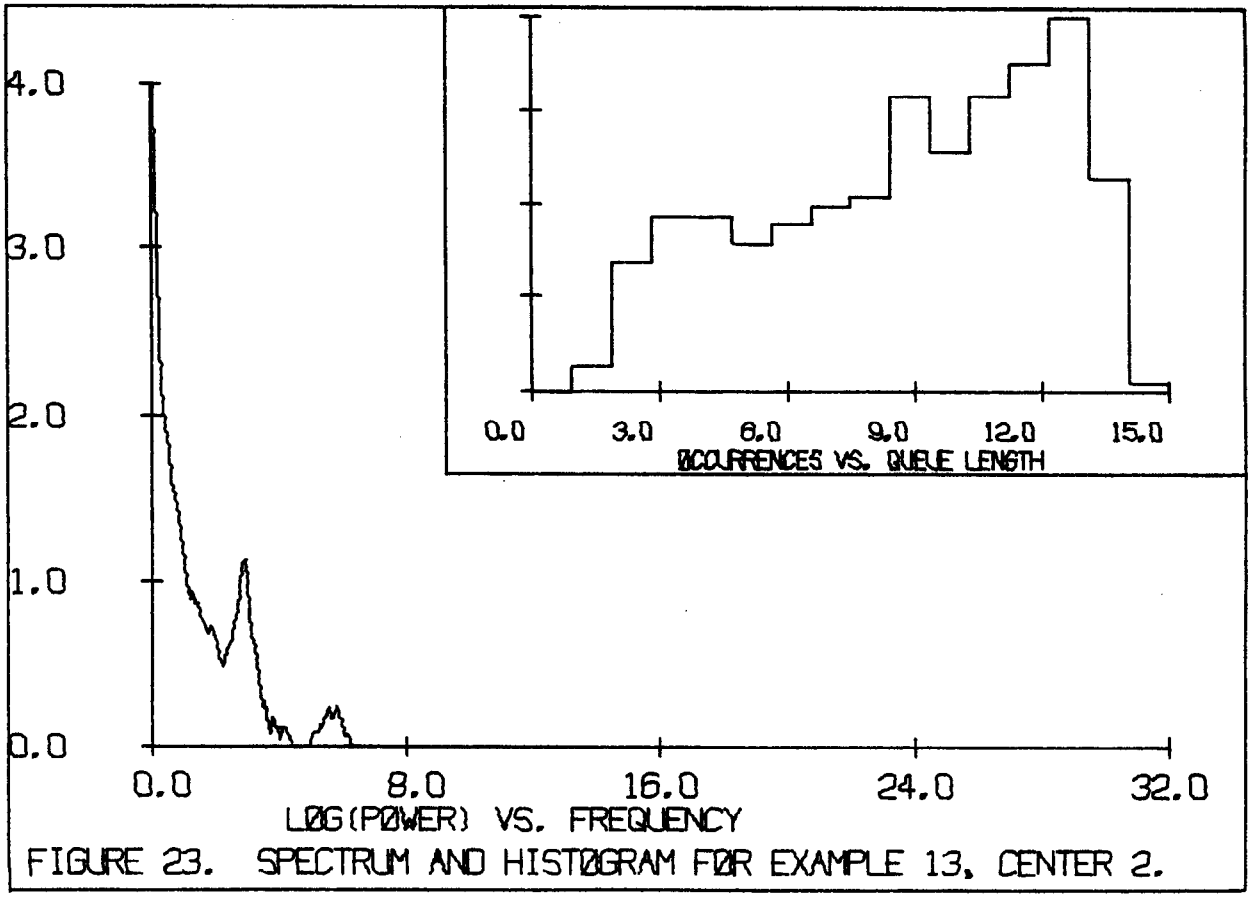


FIGURE 23. SPECTRUM AND HISTØGRAM FØR EXAMPLE 13, CENTER 2.

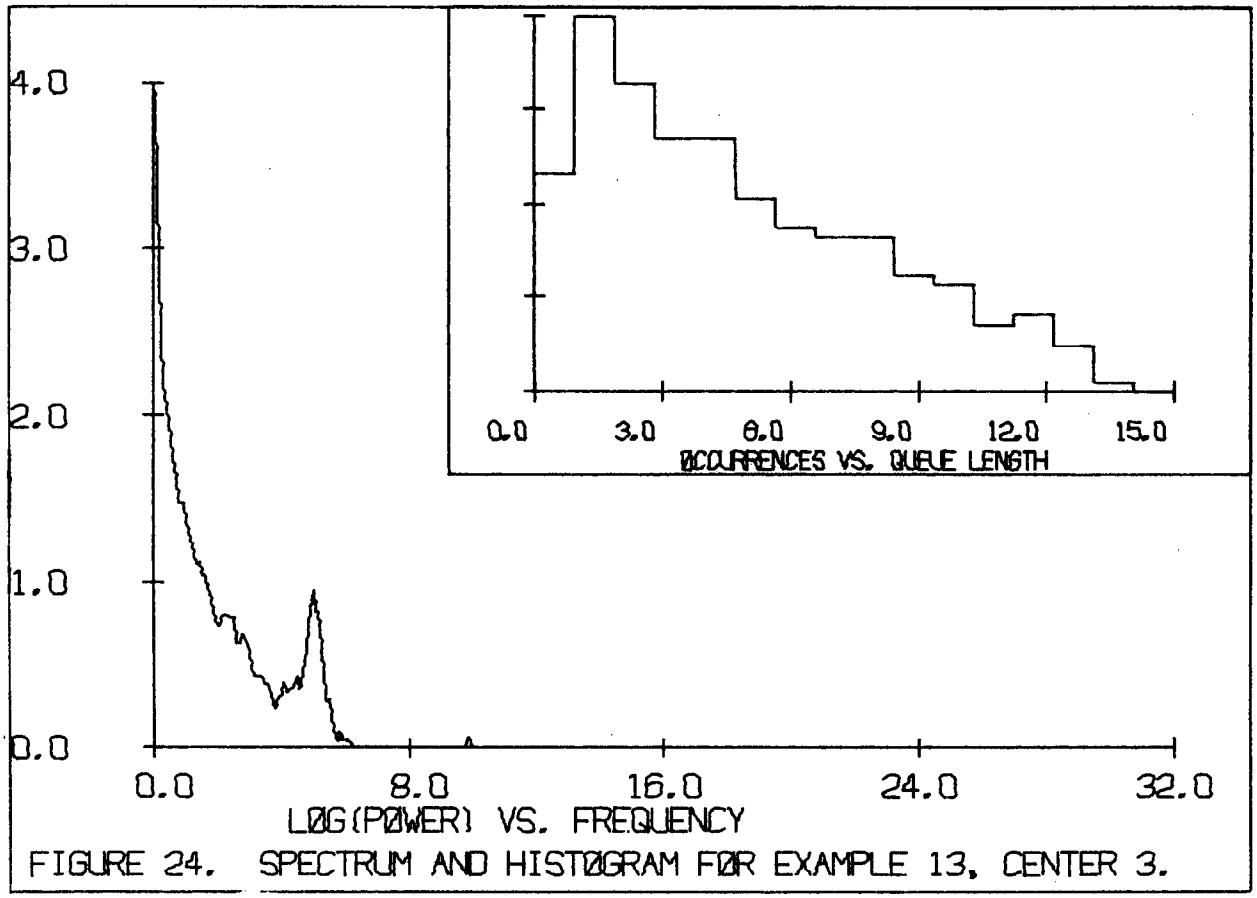


FIGURE 24. SPECTRUM AND HISTØGRAM FØR EXAMPLE 13, CENTER 3.

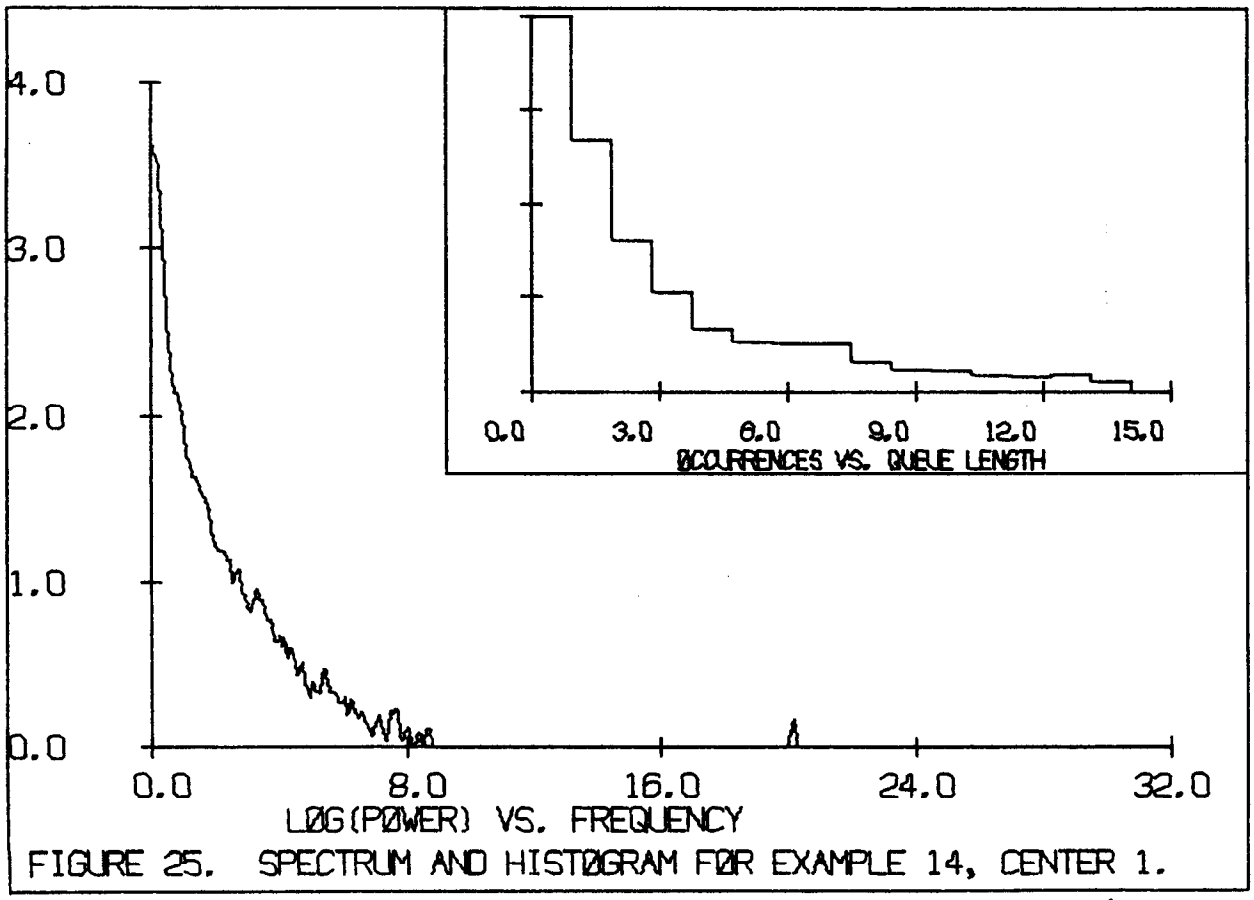


FIGURE 25. SPECTRUM AND HISTØGRAM FØR EXAMPLE 14, CENTER 1.

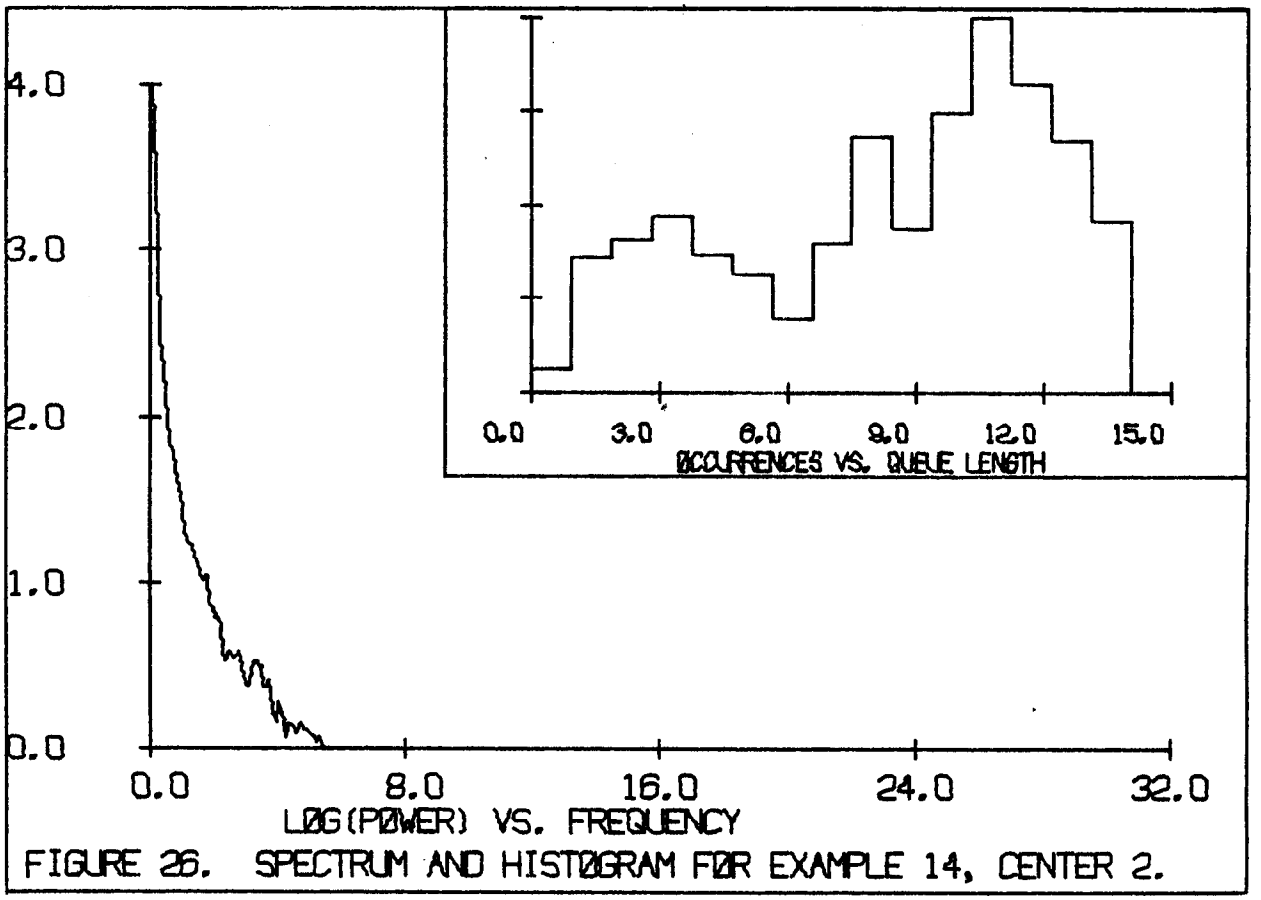


FIGURE 26. SPECTRUM AND HISTØGRAM FØR EXAMPLE 14, CENTER 2.

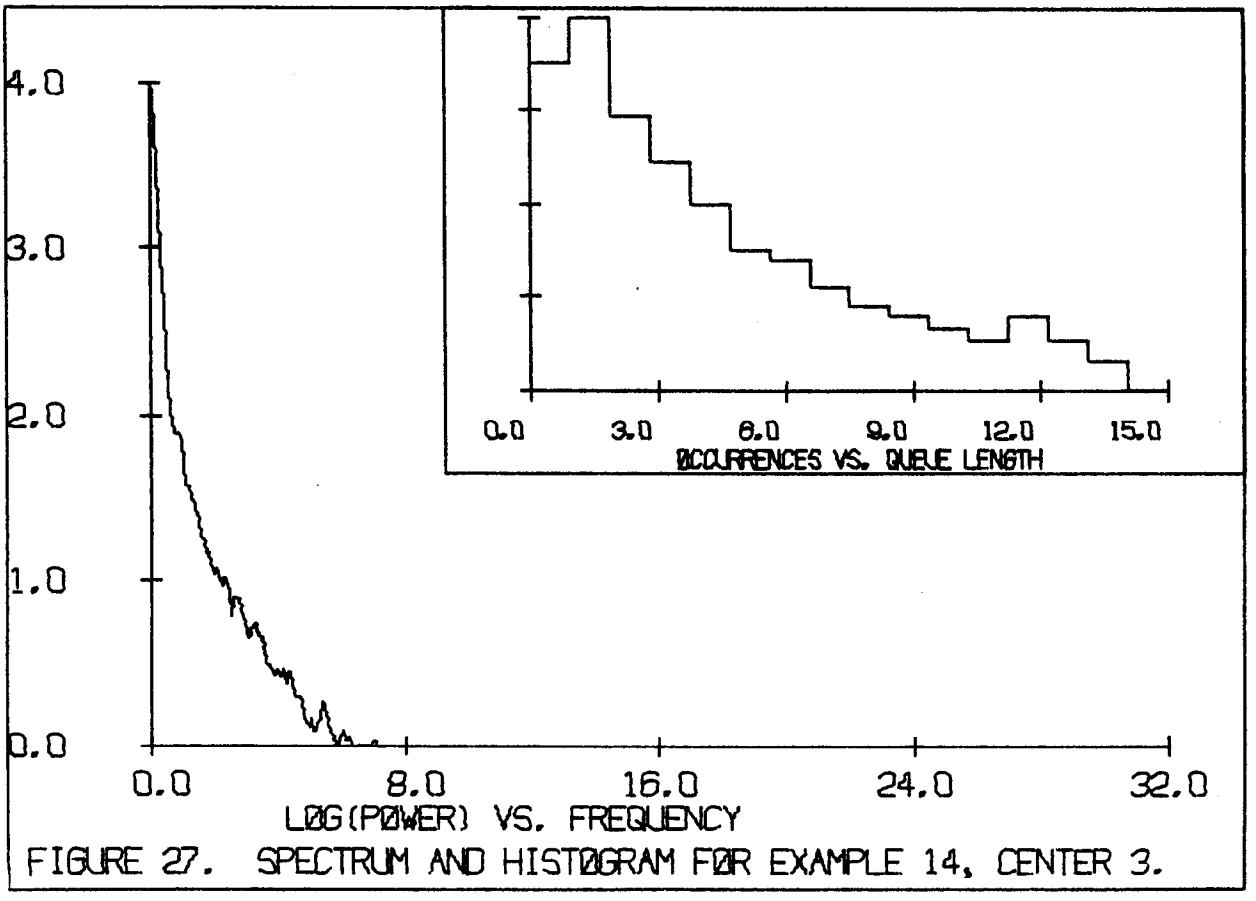


FIGURE 27. SPECTRUM AND HISTØGRAM FØR EXAMPLE 14, CENTER 3.

two may have contributed to the smearing of the right side of the fundamental peak (5.0 cycles) from center three in the spectrum for center one. The primary effect of the hypoexponential distribution, then, has been to broaden and shorten the peaks.

In order to increase the utilization values of example 11, we might take steps to increase the average queue lengths at centers one and three. To this end, in example 14 we change the scheduling methods at centers one and two to LPTF and RR respectively. Figures 25 through 27 show the histograms and power spectra which resulted. From Table 2 we see that the utilization at all three centers has been lowered to 0.70, 0.96, and 0.86 respectively. While the mean queue length did increase at center one, the standard deviation also increased at all centers. The histograms reflect the shift in queue lengths but show little change in the shape of the distribution. The spectra show greater power in the 0-2 cycle frequency range. There are spikes corresponding to the 0.200 constant service time at center three. RR scheduling at center two has removed spikes corresponding to its 0.350 service time.

Since our measures to increase utilization by trying to increase mean queue length have failed, we finally try to reduce queue length. Example 15 is similar to example 11, except SPTF replaces FCFS as the scheduling discipline at center one. Figures 28 through 30 contain the resulting histograms and power spectra. Inspection of Table 2 reveals that the utilization at centers one and three has increased very slightly. The mean queue length has declined at centers one and two while the standard deviation has decreased at center one. The histo-

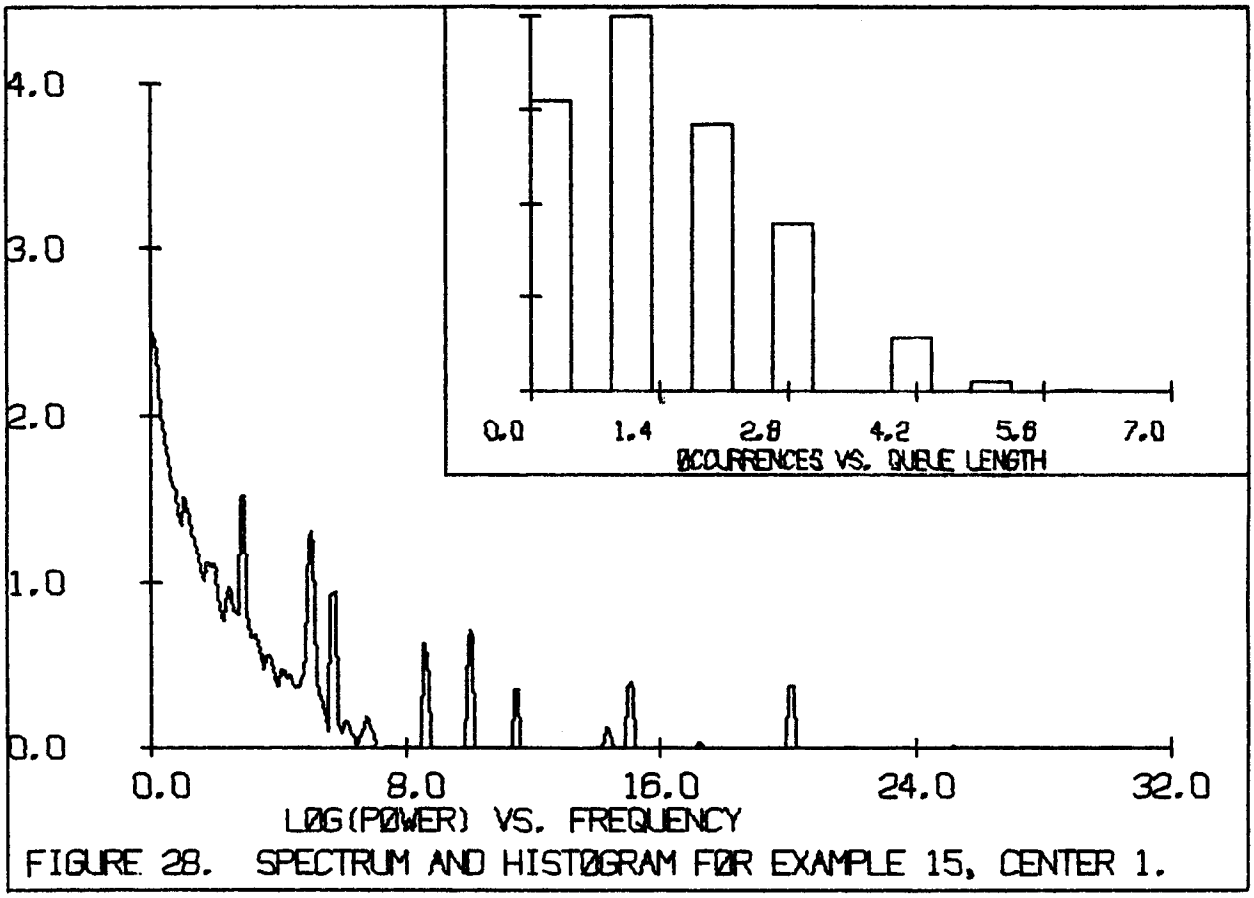


FIGURE 28. SPECTRUM AND HISTØGRAM FØR EXAMPLE 15, CENTER 1.

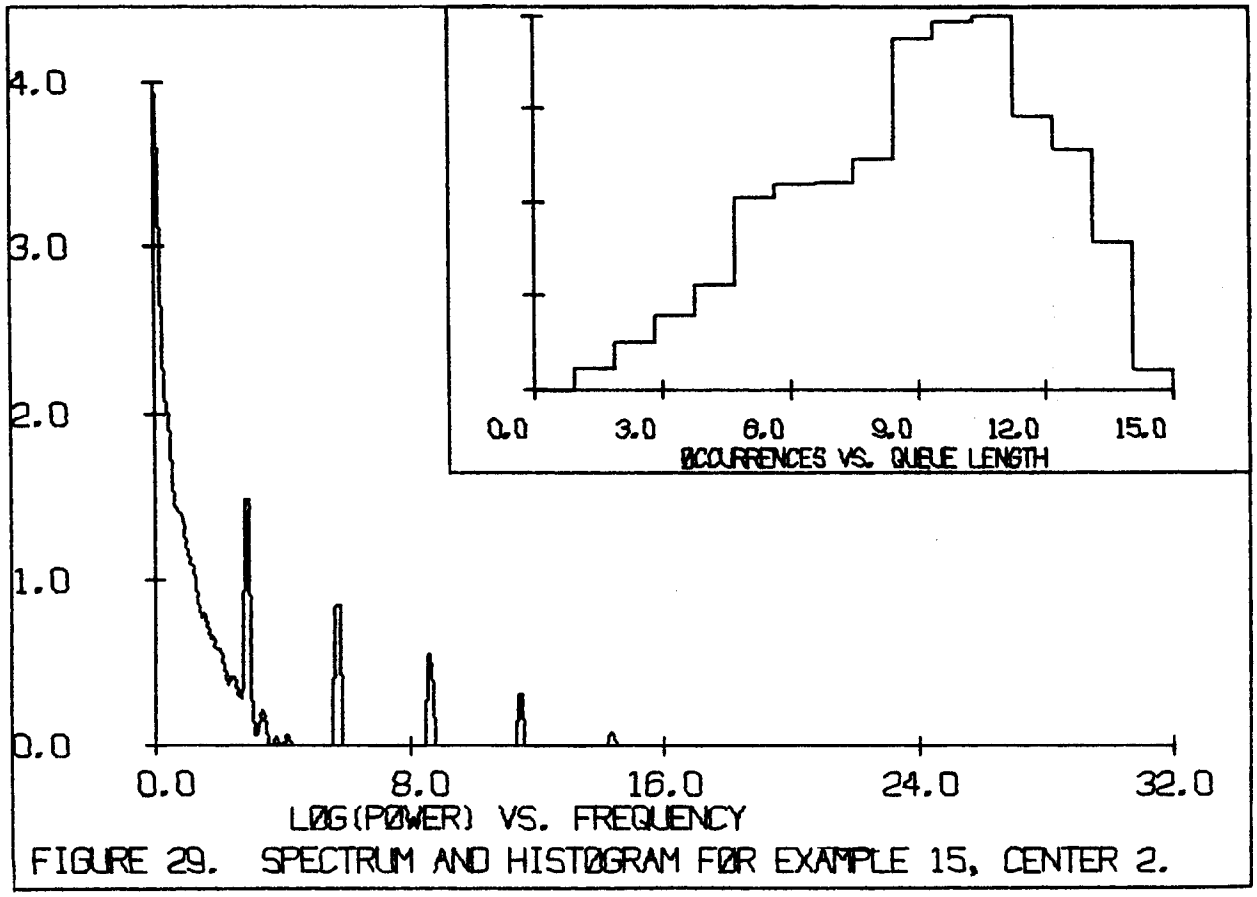


FIGURE 29. SPECTRUM AND HISTØGRAM FØR EXAMPLE 15, CENTER 2.

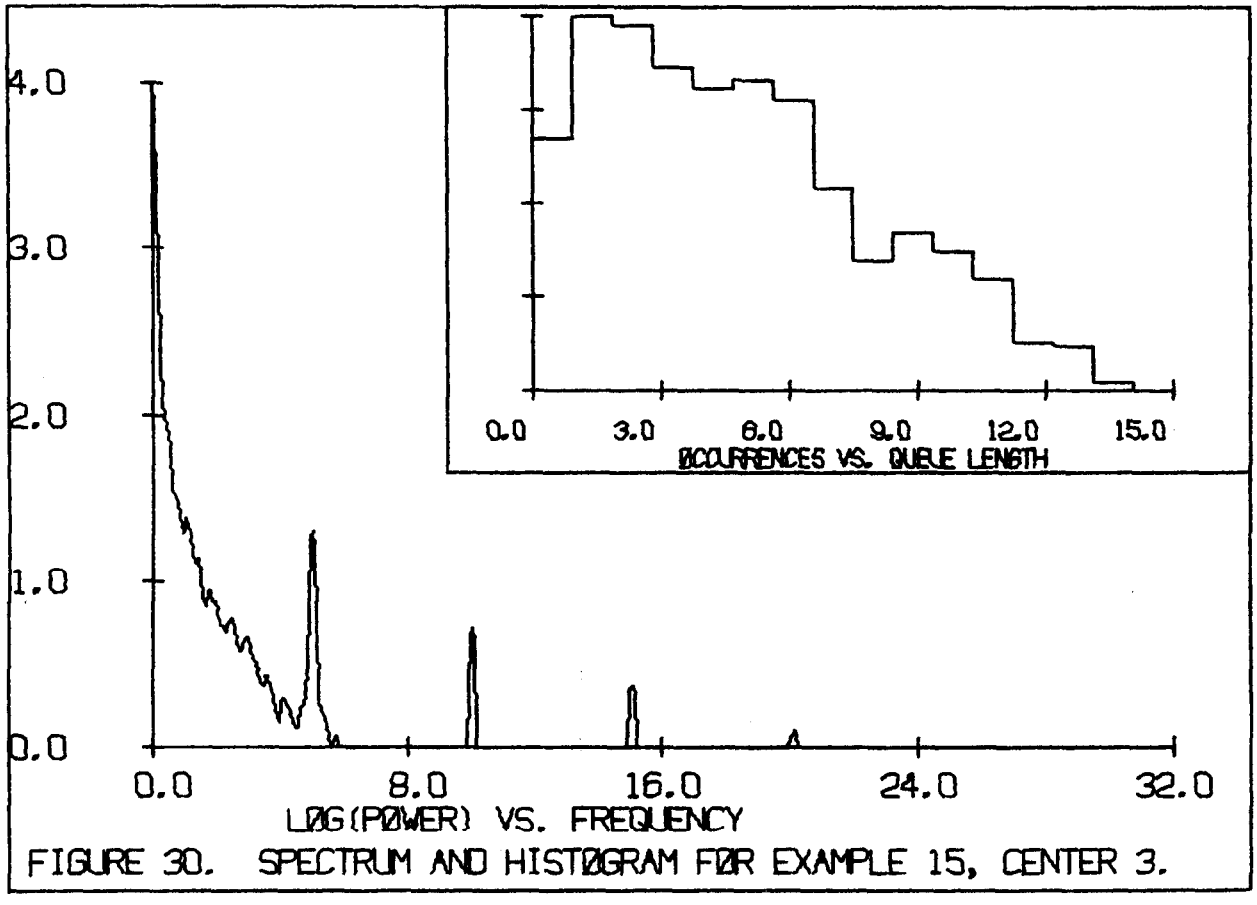


FIGURE 30. SPECTRUM AND HISTØGRAM FØR EXAMPLE 15, CENTER 3.

grams again closely parallel those of example 11. The spectra show the same basic shape. However, the spectrum for center one, in particular exhibits less power in the 0-2 cycle range and more power in the higher frequency spikes.

D. Experimental Results - Structure Three

The experiments having structure three are included principally to illustrate some further capabilities of the computer program. These are three service center open systems with jobs entering and leaving the system at center one. The jobs are in one of two priority classes each having an exponential arrival rate. The mean time between arrivals is 1.0 for priority class one and 2.0 for priority class two. The routing structure is similar to that of structure one except that from center one jobs are routed to center two with probability 0.250, to center three with probability 0.450, and out of the system with probability 0.300. The service time distributions are fixed throughout all cases. All centers have exponential service times with means 0.100, 0.300, and 0.200 respectively. Only the scheduling methods remain to be specified for each example.

In example 16 FCFS scheduling is used at all centers. From Table 2 we conclude that the system is probably underloaded. The utilization factors are 0.55, 0.45, and 0.52 respectively and the mean queue lengths are 1.23, 0.76, and 1.03. The histograms, shown in Figures 31 through 33, all have the same shape, resembling an exponential decay. The spectra, in Figures 31 through 33, all have mostly low frequency power, falling off rapidly over the 0-2 cycle range.

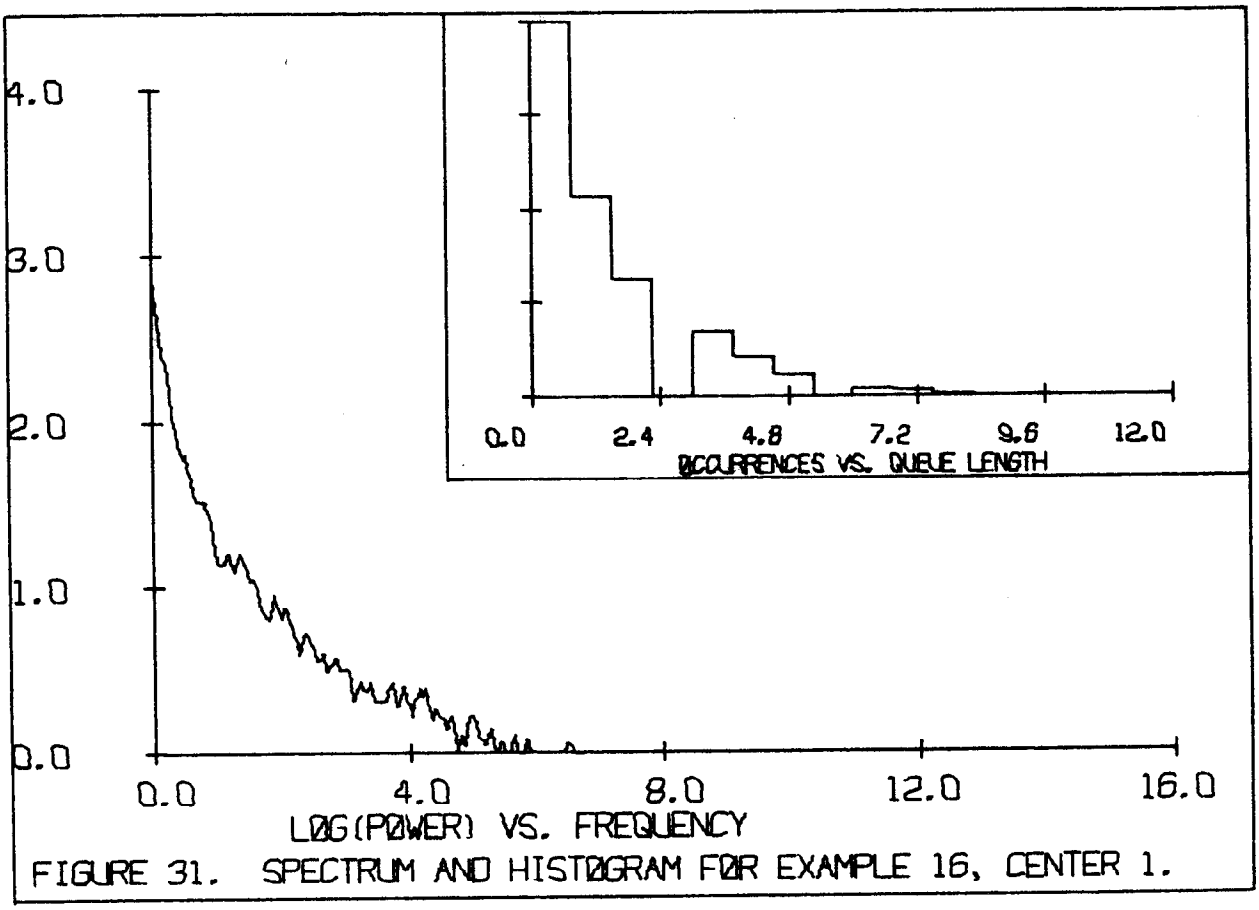


FIGURE 31. SPECTRUM AND HISTØGRAM FØR EXAMPLE 16, CENTER 1.

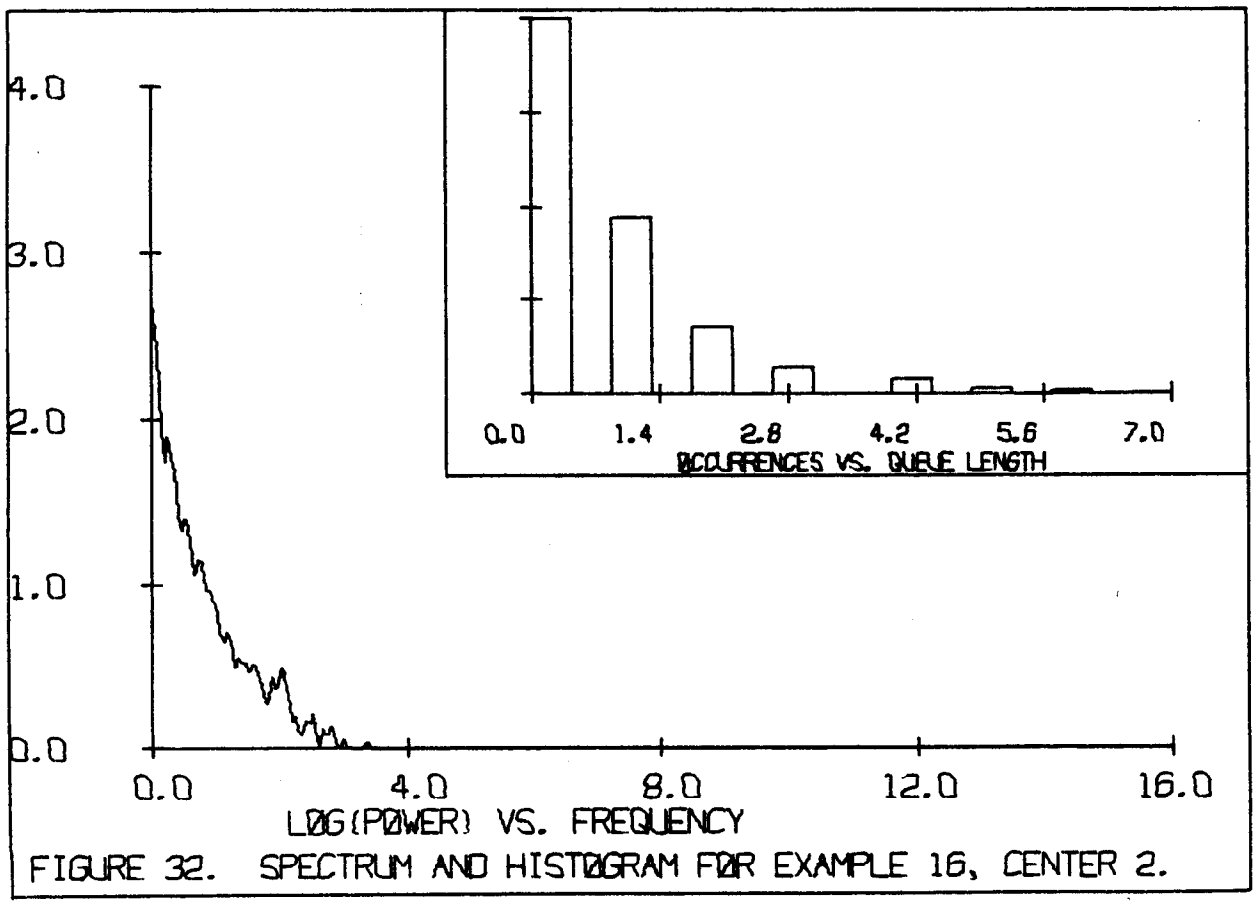


FIGURE 32. SPECTRUM AND HISTØGRAM FØR EXAMPLE 16, CENTER 2.

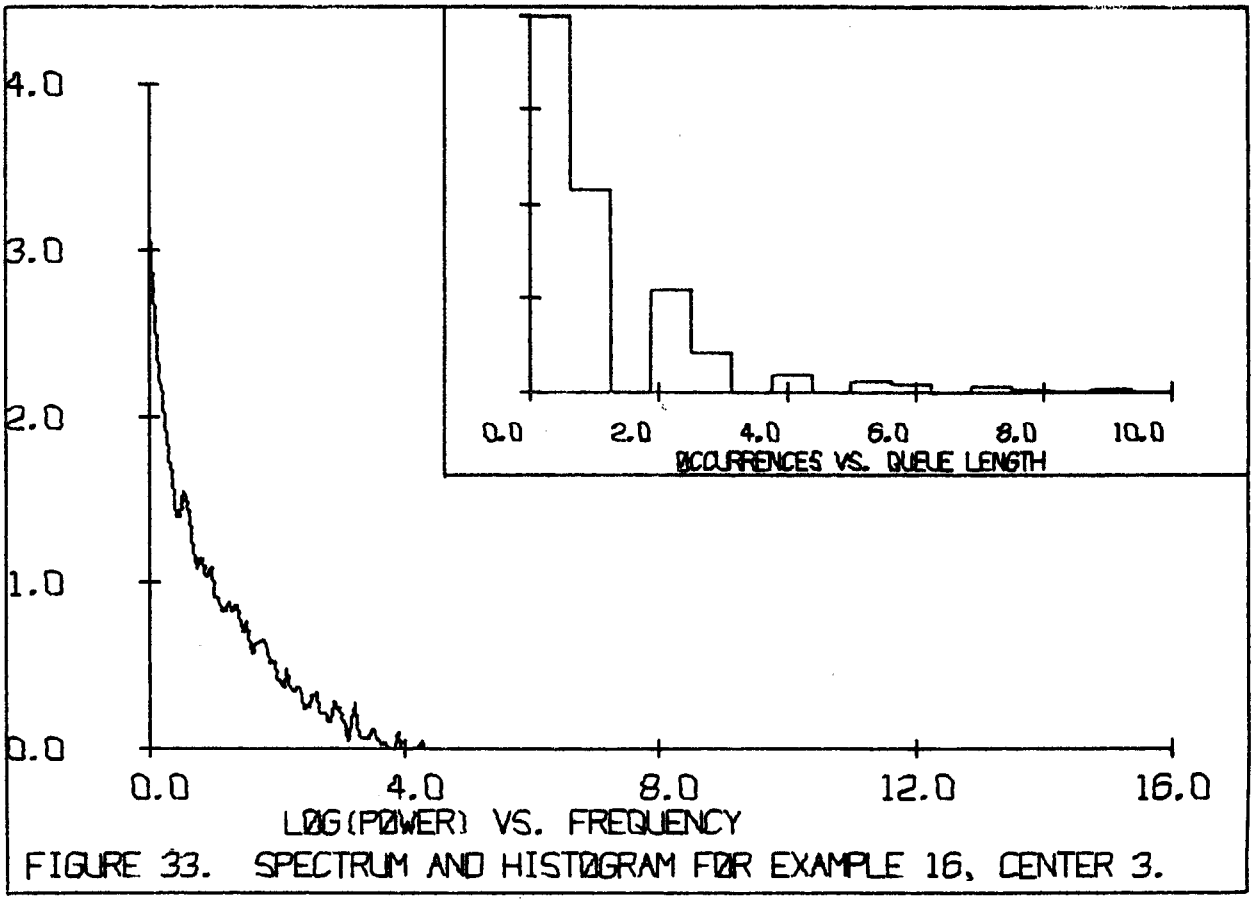


FIGURE 33. SPECTRUM AND HISTØGRAM FØR EXAMPLE 16, CENTER 3.

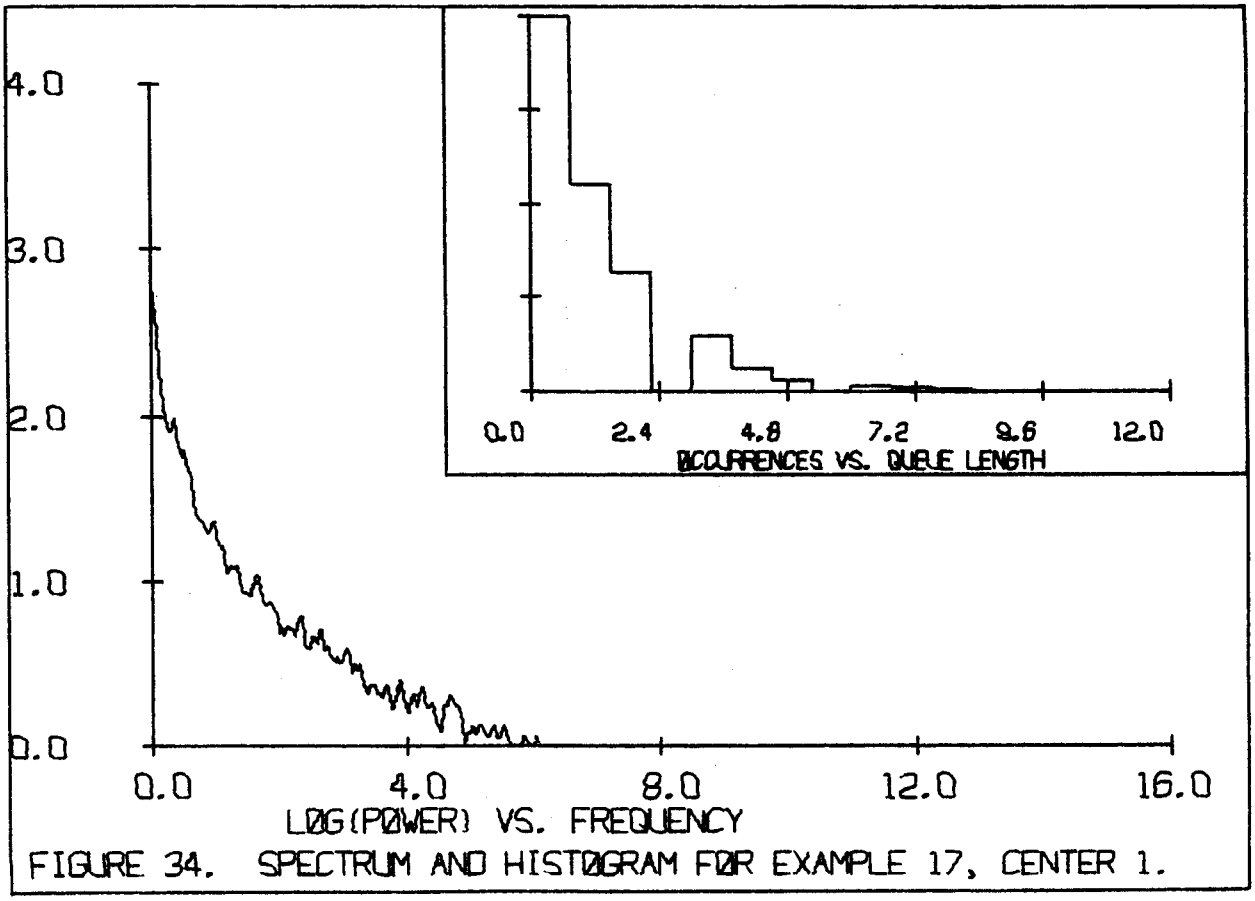


FIGURE 34. SPECTRUM AND HISTØGRAM FØR EXAMPLE 17, CENTER 1.

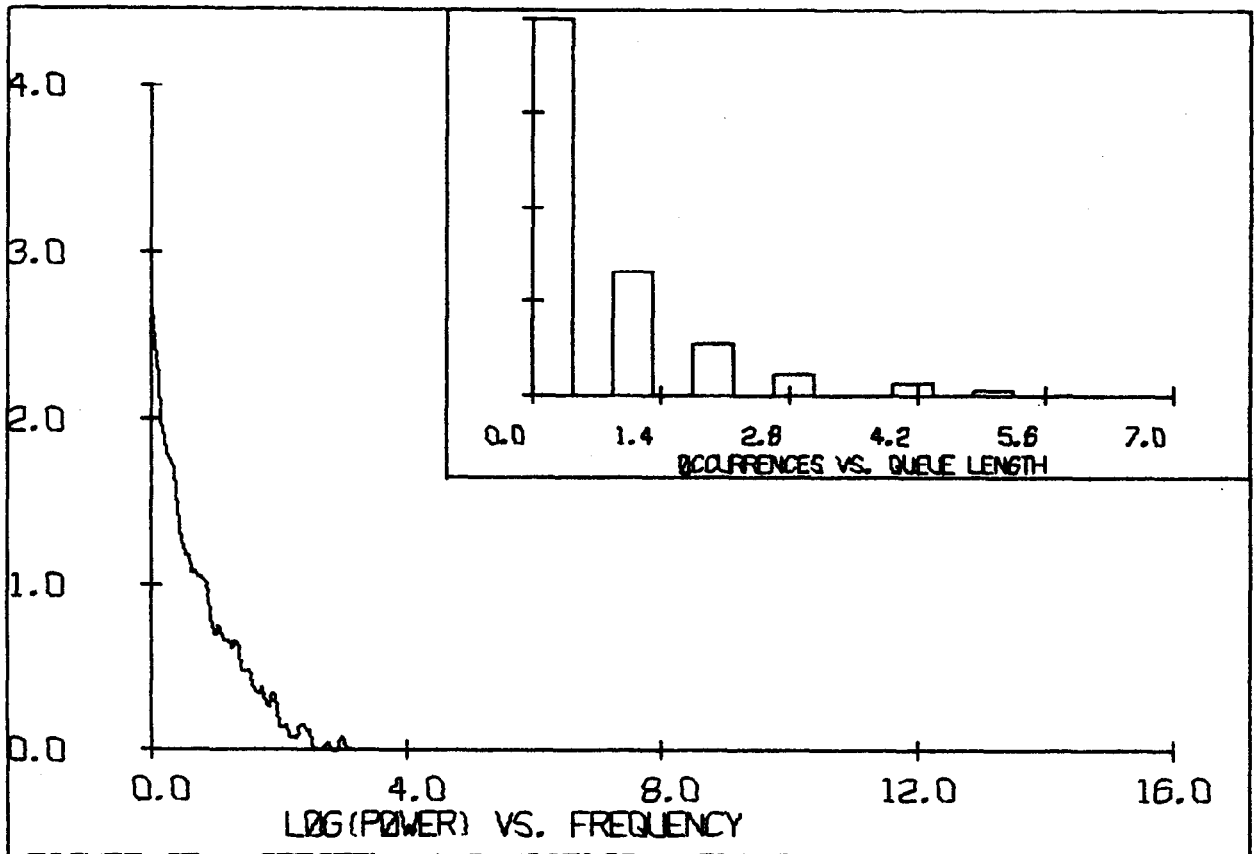


FIGURE 35. SPECTRUM AND HISTØGRAM FØR EXAMPLE 17, CENTER 2.

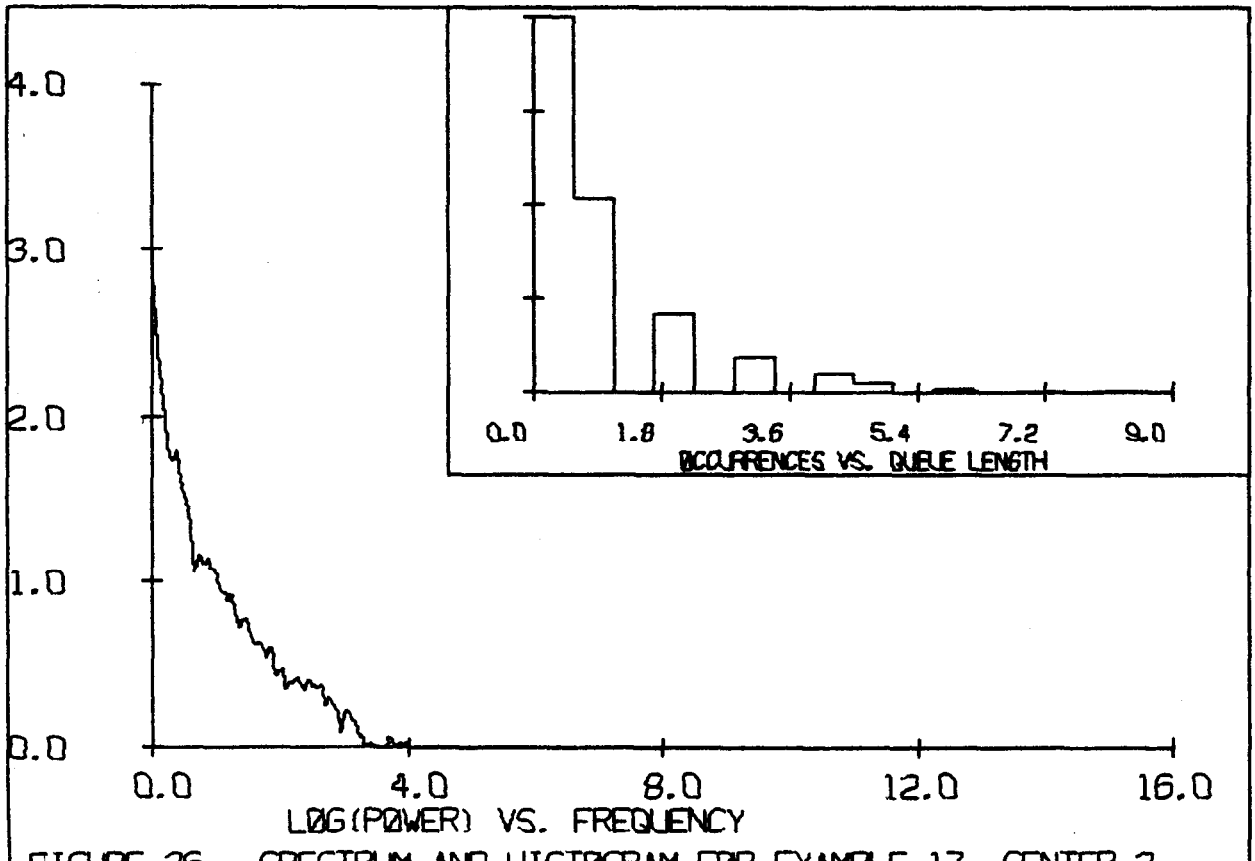


FIGURE 36. SPECTRUM AND HISTØGRAM FØR EXAMPLE 17, CENTER 3.

In example 17, we substitute SPTF for FCFS scheduling at center one. Table 2 shows that the utilization has decreased at all centers, most notably at center two. The mean queue lengths and standard deviations have also decreased. The spectra and histograms, Figures 34 through 36, show virtually no change from those in example 16.

In example 18 we introduce preemptive SPTF scheduling at center one. Jobs in priority class two preempt those of class one. Short jobs preempt longer ones. Table 2 shows a further decrease in the utilization and mean queue length at center one and, to a lesser extent, at centers two and three. The histograms, shown in Figures 37 through 39, are essentially unchanged. The spectra, Figures 37 through 39, are similar except that the spectrum for center one has somewhat less power in the low frequencies.

In example 19 we keep preemptive SPTF scheduling at center one and use non-preemptive SPTF scheduling at centers two and three. A glance at Table 2 shows further decreases in the utilization, mean queue length, and standard deviation, especially at center one. The histograms, Figures 40 through 42, again show little change. The spectra, shown in Figures 40 through 42, resemble these of example 18 very closely.

It is seen in Table 2 that there is a steady decline in the utilization factor from example 16 through example 19. This is not necessarily a bad situation as it was regarded when considering the closed systems having structures one and two. In the open systems discussed here a fixed number of jobs arrive for processing. The total amount of processing is not necessarily the same in all cases because of the

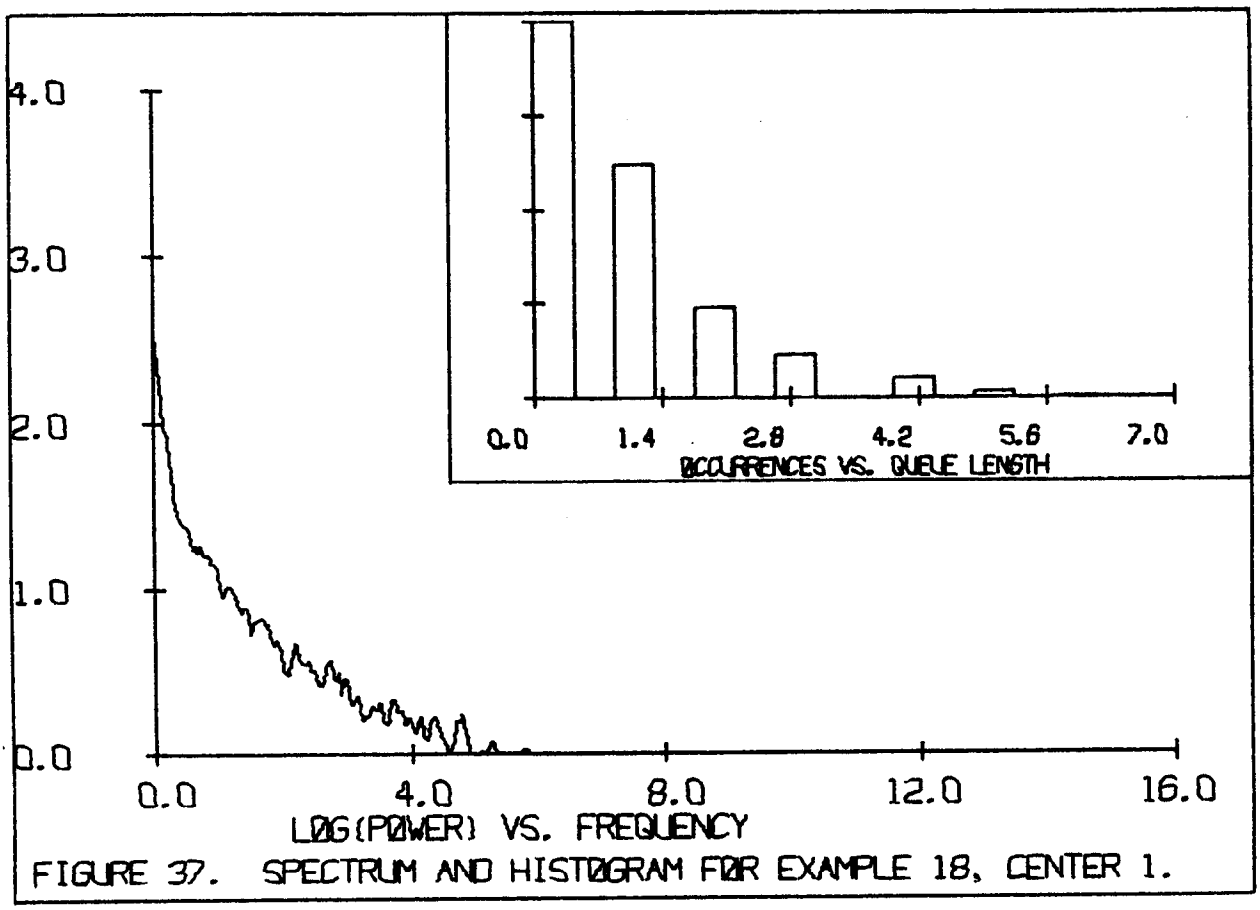


FIGURE 37. SPECTRUM AND HISTØGRAM FØR EXAMPLE 18, CENTER 1.

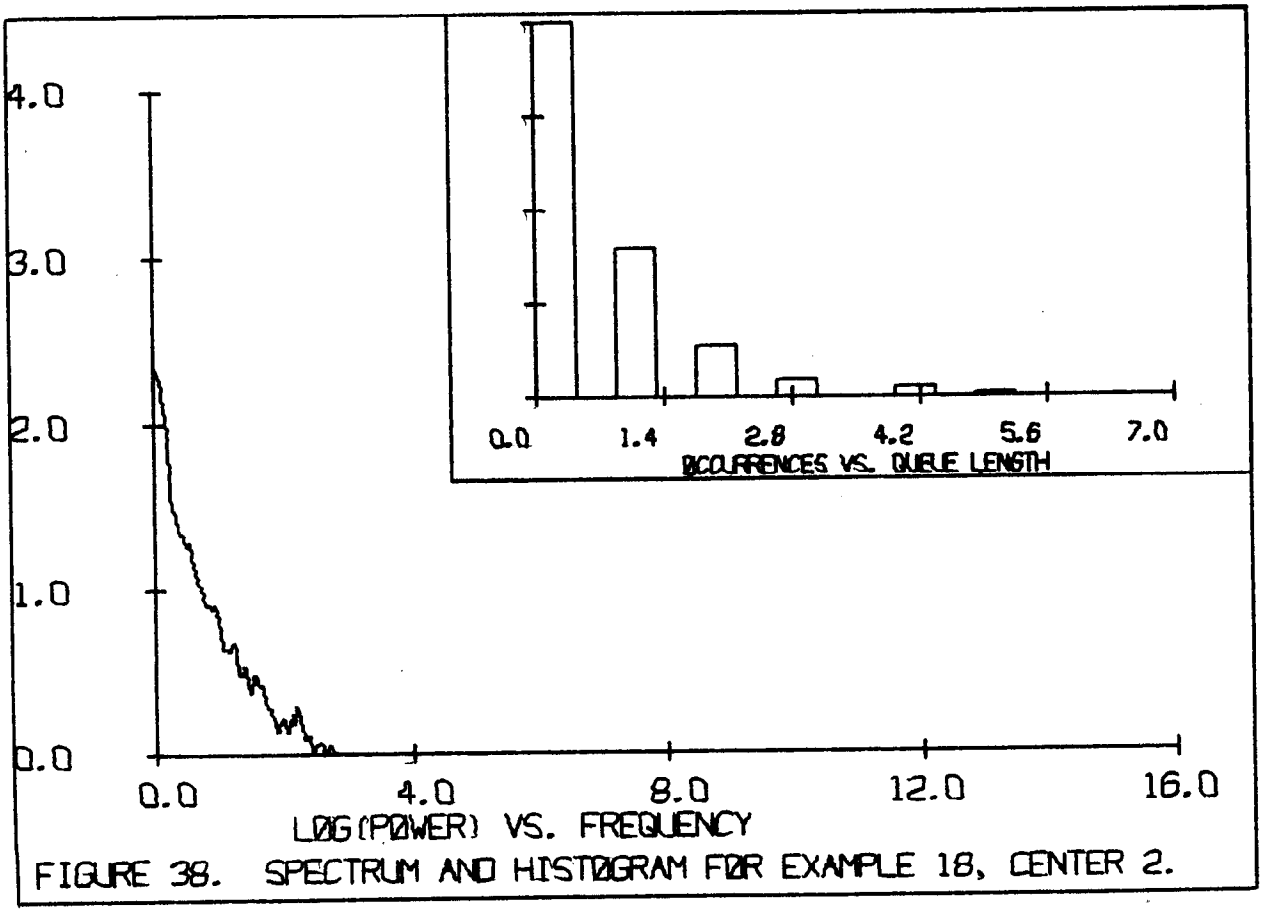


FIGURE 38. SPECTRUM AND HISTØGRAM FØR EXAMPLE 18, CENTER 2.

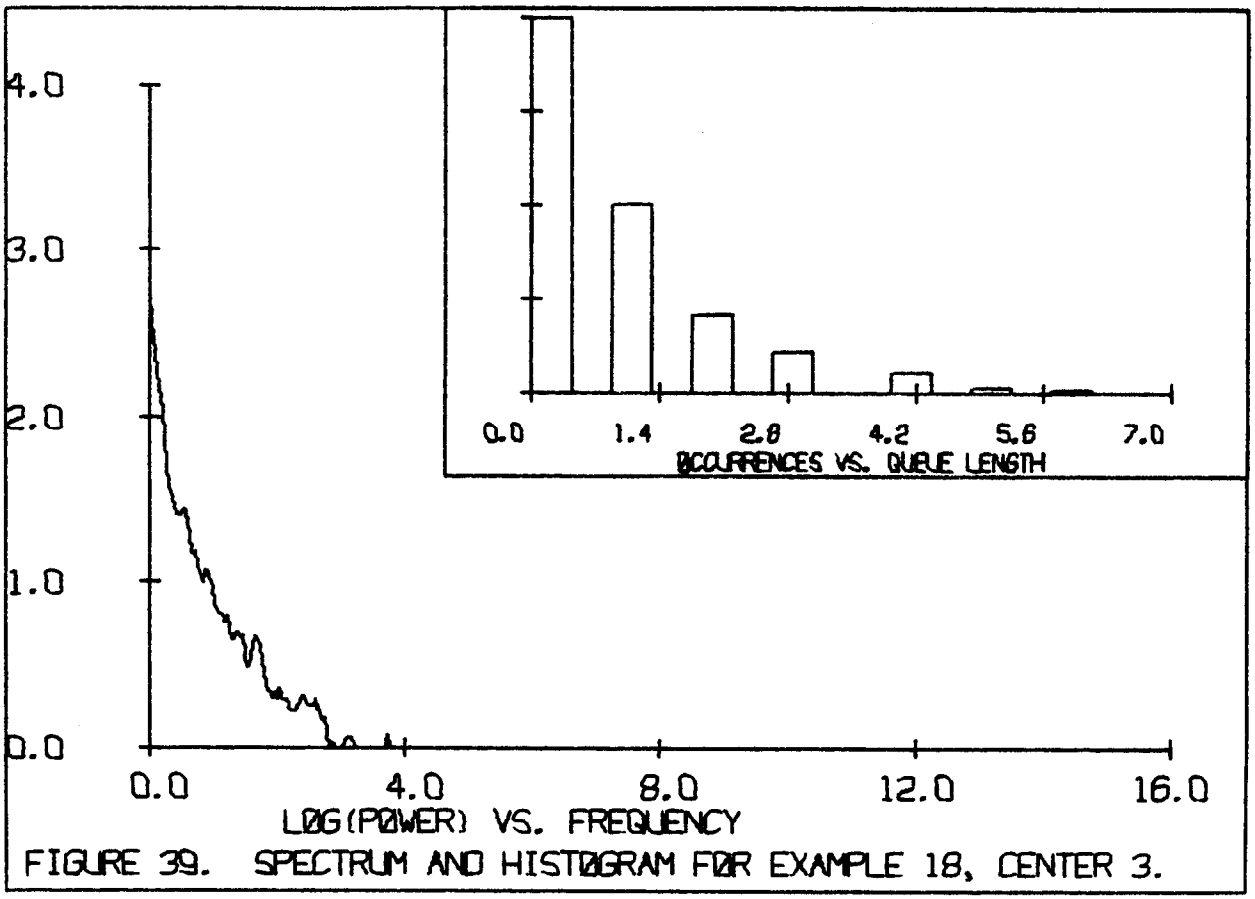


FIGURE 39. SPECTRUM AND HISTØGRAM FØR EXAMPLE 18, CENTER 3.

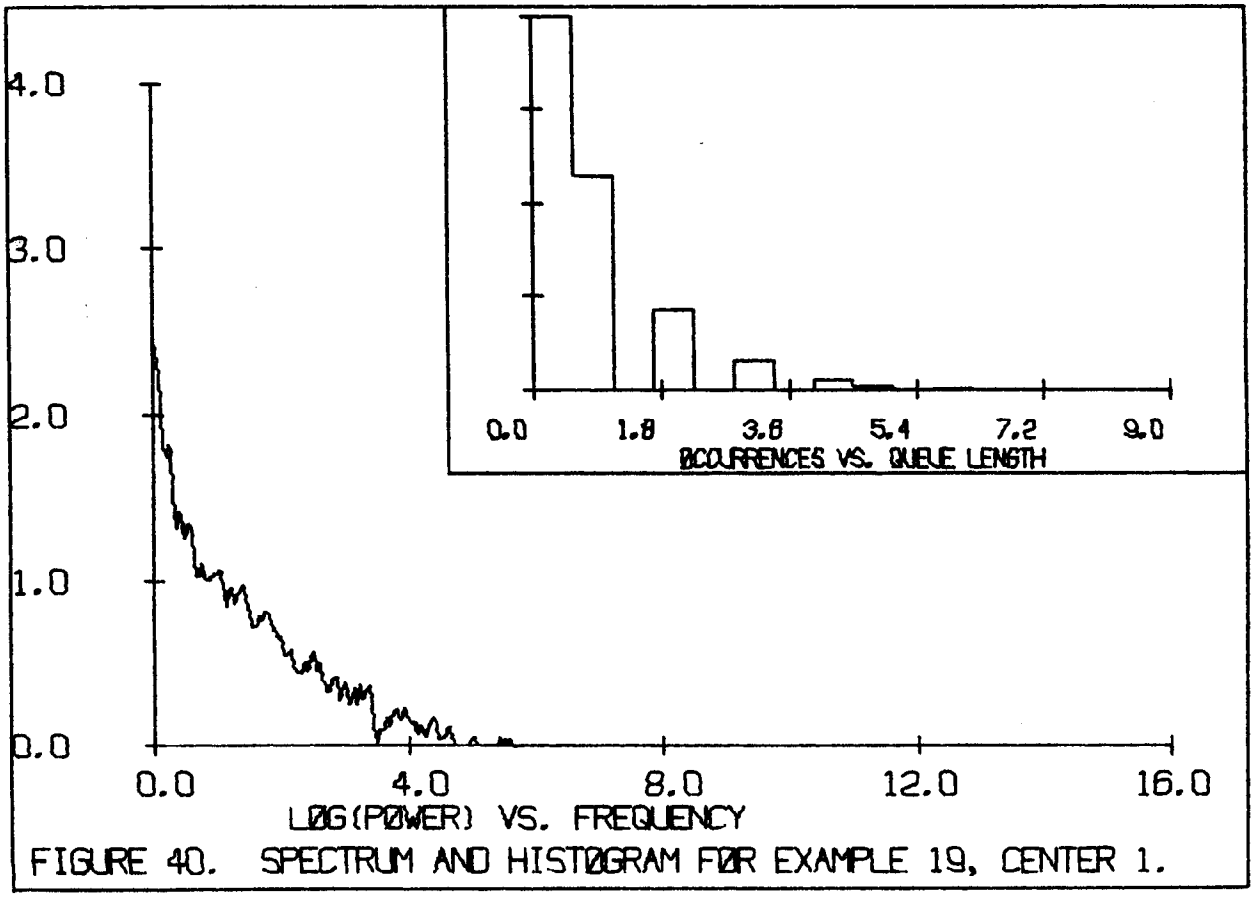


FIGURE 40. SPECTRUM AND HISTØGRAM FØR EXAMPLE 19, CENTER 1.

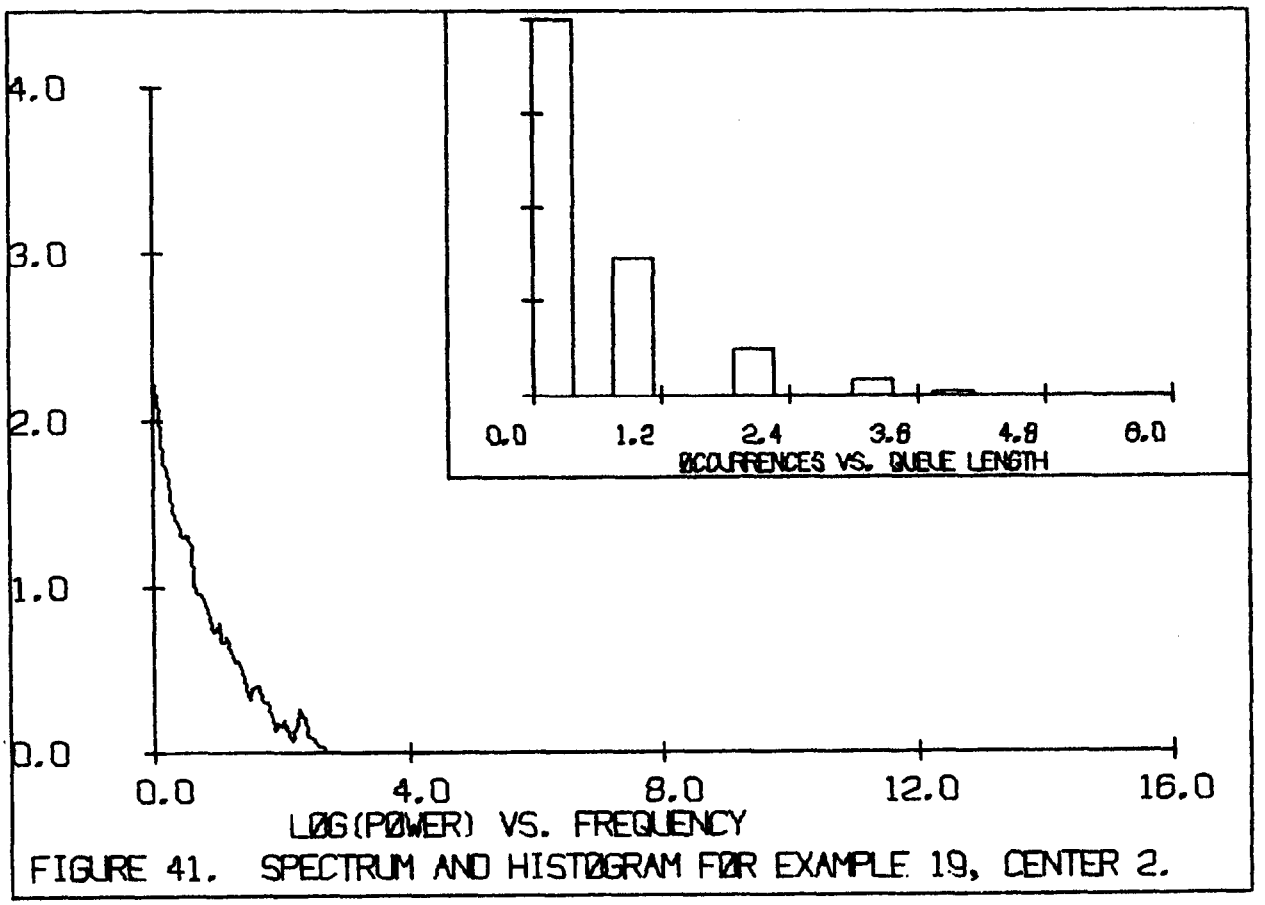


FIGURE 41. SPECTRUM AND HISTØGRAM FØR EXAMPLE 19, CENTER 2.

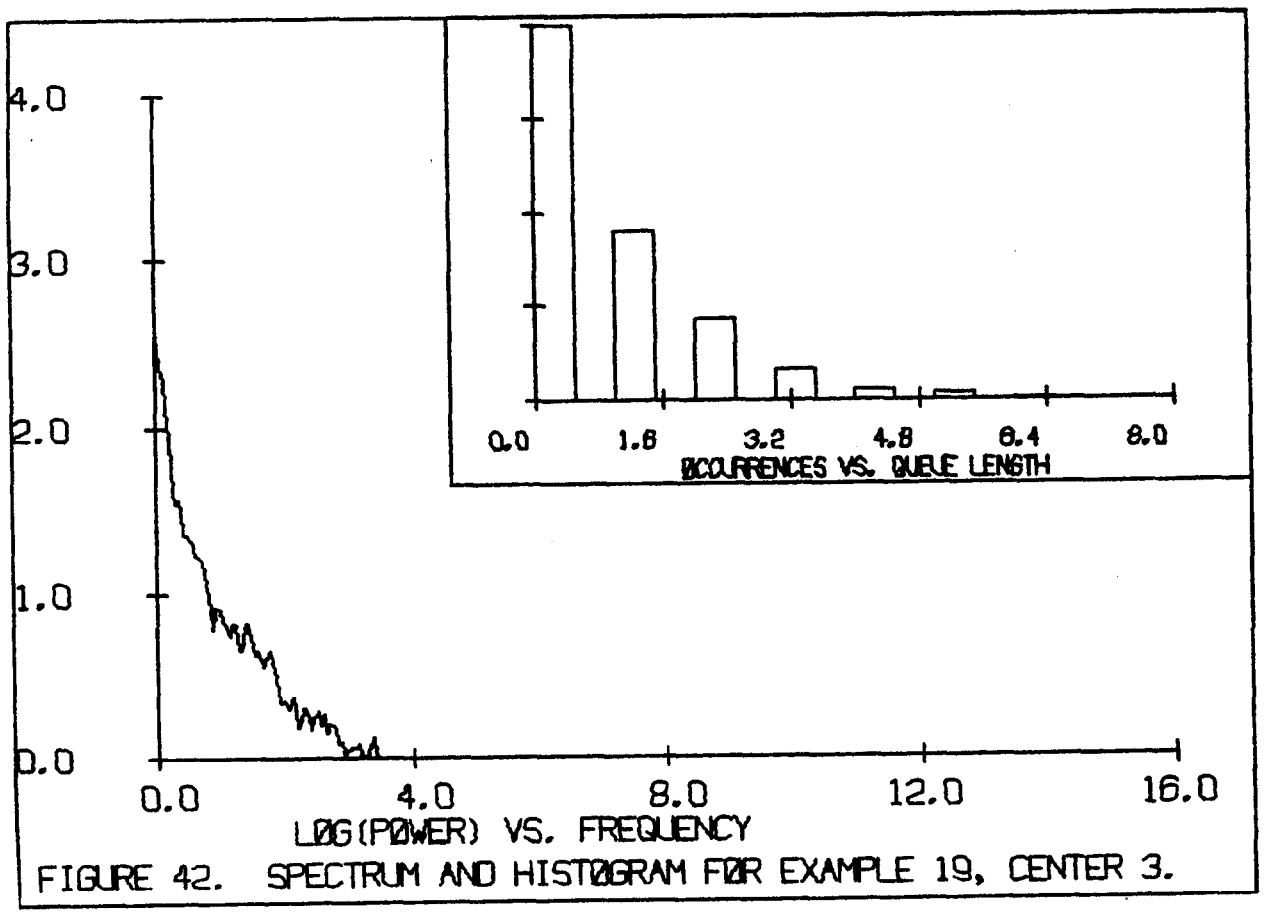


FIGURE 42. SPECTRUM AND HISTØGRAM FØR EXAMPLE 19, CENTER 3.

different methods of sequencing the jobs. As we progress from example 16 through example 19, we discriminate against jobs with long processing times to a greater extent. The shorter jobs are processed more quickly while the longer ones are postponed. However, the decreasing utilizations reflect the ability of the systems to handle a heavier job load. The small and constantly decreasing mean queue lengths and standard deviations support this statement. If these quantities were increasing while the utilization decreased, we would then recognize a problem of congestion as seen in earlier examples.

CHAPTER 6

CONCLUSIONS

There are several general conclusions which can be drawn from this study.

The Monte Carlo simulation technique has proved its worth in modeling computer systems. Building the model was a straightforward procedure as shown in Chapter 4. One can clearly design a model with the precise amount of detail desired. The flexibility of this approach has been demonstrated by the ease of constructing experiments with systems having quite different operating characteristics.

The usefulness of the histogram and power spectrum as tools for the analysis of such models has been established. When used with such time averaged variables as the utilization factor and mean and standard deviation of the queue length, these measurements provide a more thorough description of a system. As an approximation to the distribution function of the queue length random variable, the histogram showed the presence of wide oscillations in the systems of examples 3, ff. The power spectrum displayed the effect of various scheduling disciplines and service time distributions on the contributions of certain frequencies to the total variance.

In addition to demonstrating the usefulness of our method, we are able to state some specific conclusions regarding the behavior of the simple systems presented in Chapter 5. These conclusions are now summarized.

One of the first phenomena to be observed was the extreme low frequency oscillations caused by service distributions with relatively

large variance. This condition, with its associated low utilization and large variance in queue size, is easily recognized when the histogram has a convex shape such as that of example 3. The power spectrum invariably shows a large amount of power in the low frequency range.

The magnitude of the above problem depends on the type of scheduling used within the system. FCFS gave all jobs a "fair" chance at being processed, which resulted in severe congestion. LPTF favored the longer jobs which aggravated the problem. SPTF favored the shorter jobs with the result that the above problem disappeared. The histogram was sharply concave, the variance in queue length was low and the utilization high. A less discriminatory solution was to break the large jobs into smaller pieces for processing, which is the essence of RR scheduling. The favorable results of this action are demonstrated by example 8.

We have seen, in several examples, that a constant or, more generally, a low variance service distribution can give rise to a series of spikes in the power spectrum. Moreover, examples 5 through 8 show that scheduling which favors shorter jobs can shift power from the lower frequencies into these higher frequency spikes.

As pointed out in Chapter 5, the closed systems having structures one or two represent systems in which the workload is maintained at a relatively constant level by some independent process. Hence, an increase in utilization means an increase in throughput or system performance. However, in the open systems of structure three the input stream was part of the model and the total number of jobs entering the system during the simulation was approximately the same for all examples.

But the total amount of processing decreased as we favored the shorter jobs more and hence the utilizations went down. In this case the lower utilizations reflect the ability of the system to handle a higher workload. Hence, for structure three a decrease in utilization can be associated with an increase in performance.

With the above statements in mind we can conclude from Table 2 that an increase in performance among comparable examples is almost invariably associated with a decrease in the variance of the queue length. An inspection of the power spectra will reveal, too, that such performance increases are accompanied by a decrease in the low frequency power of the process.

Another interesting result, from example 9, was that a component connected on the data path between two other components can be viewed as a filter, passing the activity characteristics of one device on to the other within a certain frequency range. The cutoff frequency of this filter seems to depend on the relative magnitudes of the service times in the system.

The virtues of SPTF scheduling were further demonstrated in the open systems of examples 16 through 19. Each case favored shorter jobs more than the previous one and each showed some measure of improvement. Introducing preemptive SPTF brought about the most significant decrease in low frequency power although the change was not particularly great.

Finally, we suggest further areas of investigation based on the results of this study.

This report has dealt only with a small number of computer configurations and performance characteristics. There are certainly many

other interesting examples which could be studied. Such examples may require the model to have different types of service or arrival distributions or scheduling disciplines. The design of the computer program allows such additions to be made without complicating the logic.

Another possible use of the computer program might be the simulation of a particular real computer system. Careful attention would have to be given to choosing the parameters of the model. Since one would be most interested in the quantitative results, one would probably want to implement the computation of confidence limits for the time averaged variables and the spectrum.

SELECTED BIBLIOGRAPHY

Blackman, R.B., and Tukey, J.W. THE MEASUREMENT OF POWER SPECTRA. New York: Dover Publications, Inc., 1958.

Conway, Richard W., Maxwell, William L., and Miller, Louis W. THEORY OF SCHEDULING. Reading: Addison-Wesley Publishing Company, 1967.

Fishman, George S. and Kiviat, Philip J. "Spectral Analysis of Time Series Generated by Simulation Models", RAND Corporation, United States Air Force Project RAND, RM-4393-RM, February 1965.

Jenkins, Gwilym M., and Watts, Donald G. SPECTRAL ANALYSIS AND ITS APPLICATIONS. San Francisco: Holden-Day, Inc., 1968.

Karlin, Samuel. A FIRST COURSE IN STOCHASTIC PROCESSES. New York: Academic Press, Inc., 1969.

Morse, P. QUEUES, INVENTORIES AND MAINTENANCE. New York: John Wiley and Sons, Inc., 1958.

Naylor, Thomas H., Balintfy, Joseph L., Burdick, Donald S., and Chu, Kong. COMPUTER SIMULATION TECHNIQUES. New York: John Wiley and Sons, Inc., 1966.

Parzen, E. STOCHASTIC PROCESSES. San Francisco: Holden Day, Inc., 1962.

Singleton, Richard C. "On computing the Fast Fourier Transform", Comm. ACM 10,10 (October 1967), 647-654.

Webb, Carol. "Practical Use of the Fast Fourier Transform (FFT) Algorithm in Time Series Analysis", Applied Research Laboratories, The University of Texas at Austin, ARL-TR-70-22, June 1970.

```
C THIS PROGRAM ALLOWS ONE TO SIMULATE, IN AN ABSTRACT WAY,
C VARIOUS COMPUTER SYSTEM ORGANIZATIONS.
C
C J.A. PFEFFERLE   STANFORD,LMSC     NOV. 1972
C
C MODEL.
C THE MODEL CONSISTS OF UP TO 'MAXNSC' SERVICE CENTERS, EACH OF
C WHICH MAY HAVE SEVERAL SERVERS.
C JOBS ENTER THE SYSTEM AND ARE ROUTED FROM ONE SERVICE CENTER
C TO ANOTHER, BEING SERVICED BY ONE SERVER AT EACH CENTER.
C THERE CAN BE UP TO 'MAXJOB' JOBS IN THE SYSTEM AT ONCE.
C EACH JOB MAY BE IN ONE OF 'MAXJPC' PRIORITY CLASSES.
C
C CONCEPTUALLY THE SYSTEM CONSISTS OF A SET OF ITEMS (JOBS)
C WHICH ARE MOVED AROUND.
C AN ITEM CONTAINS SUCH QUANTITIES AS JOB NO., JOB CLASS,
C ARRIVAL, SERVICE, AND DEPARTURE TIMES, AND SERVICE CENTER.
C THESE ITEMS SIT ON VARIOUS QUEUES.
C THE SERVICE CENTER QUEUE CONTAINS ALL OF THE JOBS AT THAT
C CENTER. THE SERVICE CENTER ACTIVE QUEUE IS A SUBSET OF IT
C CONTAINING THE ACTIVE JOBS AT THAT CENTER. SIMILARLY FOR THE
C INACTIVE QUEUE, THE EVENT QUEUE IS SIMPLY THE AGGREGATE OF
C ALL THE ACTIVE QUEUES.
C IMPLEMENTATION.
C SERVICE CENTER 1 IS A FAKE CENTER HAVING ONE PSEUDO JOB FOR
C EACH JOB PRIORITY CLASS. THESE PSEUDO JOBS GENERATE THE
C ARRIVALS TO THE SYSTEM.
C SERVICE CENTER 'NSC' IS THE OUTPUT PORT OF THE SYSTEM. JOBS
C ROUTED HERE LEAVE THE SYSTEM. HENCE THE QUEUES FOR THIS CENTER
C ARE EMPTY.
C
C MOST OF THE ITEMS IN THE SYSTEM ARE JOBS OR JOB SLOTS ON THE
C FREE STORAGE LIST. A FEW ITEMS ARE USED AS HEADERS FOR QUEUES
C OR FOR THE PSEUDO JOBS OF CENTER 1.
C
C THE ARRIVAL TIMES, SERVICE TIMES AND ROUTING SEQUENCES ARE ALL
C DETERMINED STOCHASTICALLY.
C
C
C JATIME(J) - ARRIVAL TIME OF JOB J AT CURRENT CENTER
C JSTIME(J) - SERVICE TIME FOR JOB J AT CURRENT CENTER
C JDEP(J) - DEPARTURE TIME FOR JOB J FROM CURRENT CENTER
C JPC(J) - PRIORITY CLASS FOR JOB J
C JRLINK(J) - RIGHT LINK FOR JOB J
C JLLINK(J) - LEFT LINK FOR JOB J
C
C PCMAT(L) - MEAN ARRIVAL TIME FOR JOBS OF CLASS  L
C MJP(L) - MAX. NO. JOBS OF CLASS L ALLOWED INTO SYSTEM
C NJP(L) - NO. JOBS OF CLASS L IN SYSTEM
C
```

```
C SCNS(I) - NUMBER OF SERVERS AT SERVICE CENTER I
C SCNBS(I) - NUMBER OF BUSY SERVERS AT CENTER I
C SCHD(I) - INDEX TO HEAD OF QUEUE FOR SERVICE CENTER I
C SCTL(I) - INDEX TO TAIL OF QUEUE FOR SERVICE CENTER I
C SCHDIA(I) - INDEX TO HEAD OF INACTIVE QUEUE FOR CENTER I
C SCQSIZ(I) - SIZE OF INACTIVE QUEUE AT CENTER I
C SCMAXQ(I) - MAX. SIZE OF QUEUE AT CENTER I
C SCSM(I) - SCHEDULING METHOD AT CENTER I
C           1 = FCFS
C           2 = SPTF
C           3 = RR
C           4 = LPTF
C SCTS(I) - TIME SLICE AT CENTER I (FOR RR)
C SCSD(I) - TYPE OF SERVICE DISTRIBUTION FOR CENTER I
C           1   EXPONENTIAL
C           2   HYPER-EXPONENTIAL
C           3   CONSTANT
C           4   HYPO-EXPONENTIAL
C SCMST(I) - MEAN SERVICE TIME FOR CENTER I
C SCD1(I) - PARAMETER FOR SERVICE DISTRIBUTION FOR CENTER I
C ROUTE(I,K) - CUMULATIVE PROBABILITY DISTRIBUTION OF ROUTING
C              FROM CENTER I TO CENTER K.
C
C MAXJOB - MAXIMUM NUMBER OF JOBS ALLOWED IN SYSTEM
C MAXNSC - MAXIMUM NUMBER OF SERVICE CENTERS ALLOWED
C MAXJPC - MAXIMUM NUMBER OF JOB PRIORITY CLASSES ALLOWED
C MAXITM - MAXIMUM NUMBER OF ITEMS (NODES) ALLOWED
C        = MAXJOB + MAXNSC + MAXJPC
C TIME - CURRENT SYSTEM TIME
C LIMIT - CUT OFF TIME FOR SIMULATION
C FSL - POINTER TO HEAD OF FREE STORAGE LIST OF ITEMS
C ILIST - CONTROLS THE AMOUNT OF LIST OUTPUT.
C     THE LARGER ILIST, THE MORE OUTPUT
C   =10 INPUTS
C   =15 MEAN, VARIANCE STATISTICS
C   = 20 HISTOGRAM
C   = 25 SPECTRAL ANALYSIS PLOT
C   = 30 LIST OF RAW DATA
C   = 30 LIST OF HISTOGRAM AND SPECTRAL ANALYSIS NUMBERS
C   = 35 DIAGNOSTIC PRINTOUT FROM ENTRJB,SLCTJB,RMOVJB
      COMMON /BLK2/ SEED,LOOP,LIMIT
      INTEGER SEED
      REAL LIMIT
      COMMON /BLK3/
     A ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
C ***
      CALL UGOPEN('CAL/10D,DDNAME=PLOTTAPE*',1)
C ***
    5 CONTINUE
```

```
      CALL INPUT
   10 CALL NEXTEV
      CALL SAMPLE
      CALL MOVEJB
      LOOP = LOOP + 1
      IF (TIME .LT. LIMIT) GO TO 10
      CALL OPUT
      CALL STAT1
      GO TO 5
      END
C
C
C****
      SUBROUTINE INPUT
C---------READ AND PRINT MODEL PARAMETERS
      COMMON /BLK1/
     X QSTAT(3,10),BSTAT(3,10),ALLBZY(3,10),SUMBZY(3,10)
      COMMON /BLK2/ SEED,LOOP,LIMIT
      INTEGER SEED
      REAL LIMIT
      COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
      DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
      INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
      REAL JATIME,JSTIME,JDEP
      COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
      DIMENSION
     X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
     X  ,SCSD(10),SCD1(10),SCTS(10)
      INTEGER SCSM,SCSD
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
      COMMON /BLK7/
     X NDATA,T2SAMP,DELSAM,ISTDAT,NSTOR,N22N,NP2,NS,PT
   50 FORMAT(4I10,3F10.2)
   51 FORMAT(40I2)
   52 FORMAT(5I10, 3F10.2)
   53 FORMAT(8F10.2)
   54 FORMAT(20H1SIMULATION PROGRAM     )
   55 FORMAT(25HONO. OF SERVICE CENTERS      ,I3 )
   56 FORMAT(20HOSERVERS PER CENTER    )
   57 FORMAT(1X, 13I10)
```

```
   58 FORMAT(20H0SCHEDULING METHOD    )
   59 FORMAT(32H0SERVICE DISTRIBUTION PARAMETERS    )
   60 FORMAT(1X, 13F10.3)
   61 FORMAT(25H0ROUTING PROBABILITIES         )
   62 FORMAT(25H0NO. OF PRIORITY CLASSES    ,I3)
   63 FORMAT(20H0MEAN ARRIVAL TIMES   )
   64 FORMAT( 5H0SEED  , I12)
   65 FORMAT(20H0MAX. QUEUE SIZE        )
   66 FORMAT(15H0NO. SAMPLES       ,I10)
   68 FORMAT(8I10)
   67 FORMAT(20H0SAMPLE INCREMENT     ,F10.4)
   69 FORMAT(20H0MAX. JOBS IN SYSTEM  )
   70 FORMAT(24H0SMOOTHING PARAMETERS     )
   71 FORMAT(1X,I10,F10.3)
   72 FORMAT(16H0START/STOP TIME   )
C--------INITIALIZE THE SYSTEM
      MAXJOB = 500
      MAXDAT = 34000
      MAXNSC = 10
      MAXJPC = 5
      MAXITM = MAXJOB + MAXNSC + MAXJPC
      XLARGE = 1.0E15
      TIME = 0.0
      LOOP = 0
      MAXBLK = 5
      ILIST = 25
C--------READ DATA BLOCK NO.
  100 READ(5,51) I
C ***
      IF (I .EQ. 0) CALL UGCLOS(1)
C ***
      IF (I .EQ. 0) STOP
      IF (I .LT. 0 .OR. I .GT. MAXBLK)  CALL ERROR(5)
      GO TO (120,140,150,160,170),I
C---------BLOCK 1
  120 READ(5,52) NSC,NJPC,SEED,NP2,NS,PT,LIMIT,T2SAMP
      IF (T2SAMP .LT. 0.0) T2SAMP = 10.0
      N22N = 2**NP2
      DELSAM = (LIMIT - T2SAMP) / N22N
      IF (NSC .LE. 2 .OR. NSC .GT. MAXNSC) CALL ERROR(6)
      IF (NJPC .LE. 0 .OR. NJPC .GT. MAXJPC) CALL ERROR(6)
      IF (SEED .LE. 0) SEED = 17
      GO TO 100
C--------BLOCK 2
  140 READ(5,53) (PCMAT(J),J=1,NJPC)
      READ(5,68) (MJP(J),J=1,NJPC)
      GO TO 100
C--------BLOCK 3
  150 CONTINUE
      DO 155 J=1,NSC
```

```
      READ(5,50) SCNS(J),SCSM(J),SCMAXQ(J),SCSD(J),SCMST(J),
     X    SCD1(J),SCTS(J)
      IF (SCNS(J) .LE. 0) SCNS(J) =1
      IF (SCMAXQ(J).LE. 0) SCMAXQ(J) = 1000
  155 CONTINUE
      GO TO 100
C--------BLOCK 4
  160 DO 165 J=1,NSC
      READ(5,53) (ROUTE(J,K),K=1,NSC)
  165 CONTINUE
      GO TO 100
C--------BLOCK 5
  170 WRITE(6,54)
      IF (ILIST .LT. 10) GO TO 173
      WRITE(6,64) SEED
      WRITE (6,66) N22N
      WRITE (6,67) DELSAM
      WRITE(6,55)NSC
      WRITE(6,56)
      WRITE(6,57) (SCNS(J),J=1,NSC )
      WRITE(6,58)
      WRITE(6,57) (SCSM(J),J=1,NSC)
      WRITE (6,60) (SCTS(J),J=1,NSC)
      WRITE(6,65)
      WRITE(6,57) (SCMAXQ(J),J=1,NSC)
      WRITE(6,59)
      WRITE (6,57) (SCSD(J),J=1,NSC)
      WRITE(6,60) (SCMST(J),J=1,NSC)
      WRITE (6,60) (SCD1(J),J=1,NSC)
      WRITE(6,61)
      DO 175 J=1,NSC
      WRITE(6,60) (ROUTE(J,K),K=1,NSC)
  175 CONTINUE
      WRITE(6,62) NJPC
      WRITE(6,63)
      WRITE(6,60) (PCMAT(J),J=1,NJPC)
      WRITE (6,69)
      WRITE (6,57) (MJP(J),J=1,NJPC)
      WRITE(6,70)
      WRITE (6,71) NS,PT
      WRITE(6,72)
      WRITE(6,53) T2SAMP,LIMIT
  173 CONTINUE
C--------INITIALIZE STATISTICS
      DO 30 I=1,3
      DO 20 J=1,NSC
      SUMBZY(I,J) =0.0
      ALLBZY(I,J) =0.0
      QSTAT(I,J) =0.0
      BSTAT(I,J) = 0.0
```

```
   20 CONTINUE
   30 CONTINUE
      CALL CSINIT(NSC-2,SCMAXQ(2),NWST)
      NDATA = 0
C*****THIS STATEMENT MUST BE CHANGE IF COMMON BLOCK 4 IS
      ISTDAT = MAX0(6*MAXITM+2, N22N+3)
      I = (MAXDAT -ISTDAT+ 1) / NWST
      IF (I .LT. N22N) CALL ERROR(2)
      NSTOR = NWST * N22N
C--------INITIALIZE FOR ARRIVALS
      DO 180 I=1,NJPC
      NJP(I) = 0
  180 CONTINUE
C--------INITIALIZE QUEUES FOR REGULAR SERVICE CENTERS
      DO 125 I=2,NSC
      SCHD(I) = I
      SCTL(I) = I
      SCHDIA(I) = I
      SCQSIZ(I) = 0
      SCNBS(I) = 0
  125 CONTINUE
C--------INITIALIZE SERVICE CENTER 1
      DO 130 I=1,NJPC
      JRLINK(NSC+I) = NSC + I + 1
      JLLINK(NSC+I) = NSC + I - 1
      JPC(NSC+I) = I
      JDEP(NSC+I) = 0.0
  130 CONTINUE
      JLLINK(NSC+1) = 1
      JRLINK(NSC+NJPC) = 1
      SCHD(1) = NSC + 1
      SCTL(1) = NSC + NJPC
      SCQSIZ(1) = 0
      SCHDIA(1) = 1
       SCNBS(1) = 0
C--------SET UP FSL
      FSL = NSC + NJPC + 1
      DO 135 I=FSL,MAXITM
      JRLINK(I) = I+1
      JLLINK(I) = I-1
  135 CONTINUE
      JLLINK(FSL) = MAXITM
      JRLINK(MAXITM) = FSL
      RETURN
      END
C
C****
      SUBROUTINE NEXTEV
C--------FIND THE NEXT ITEM IN THE EVENT QUEUE
      COMMON /BLK3/
```

```
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
       INTEGER OLDCTR
       DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
       DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
       INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
       REAL JATIME,JSTIME,JDEP
       COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
       DIMENSION IDATA(34000),DATA(34000)
       EQUIVALENCE (IDATA,DATA,JLLINK)
       EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
       JOBNO = 0
       TIME = XLARGE
C--------SEARCH EACH ACTIVE QUEUE
       DO 250 I=1,NSC
       J = I
  200  J = JRLINK(J)
C--------CHECK FOR END OF THIS ACTIVE QUEUE
       IF (J .EQ. SCHDIA(I)) GO TO 250
       IF (JDEP(J) .GE. TIME) GO TO 200
       JOBNO = J
       OLDCTR = I
       TIME = JDEP(J)
       GO TO 200
  250  CONTINUE
       IF (JOBNO .EQ. 0) CALL ERROR(1)
C--------UPDATE STATISTICS FOR OLD CENTER
       CALL STAT(OLDCTR)
       RETURN
       END
C
C****
       SUBROUTINE STAT(ICTR)
C--------UPDATE STATISTICS FOR CENTER 'ICTR',
       COMMON /BLK1/
     X QSTAT(3,10),BSTAT(3,10),ALLBZY(3,10),SUMBZY(3,10)
       COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
       INTEGER OLDCTR
       DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
       DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
       INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
       REAL JATIME,JSTIME,JDEP
       COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
```

```
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
      CALL UPDATE(QSTAT(1,ICTR),SCQSIZ(ICTR)+SCNBS(ICTR),TIME)
      CALL UPDATE(BSTAT(1,ICTR),SCNBS(ICTR),TIME)
      CALL UPDATE( SUMBZY(1,ICTR),MINO(SCNBS(ICTR),1),TIME)
      I = 0
      IF (SCNBS(ICTR) .EQ. SCNS(ICTR)) I = 1
      CALL UPDATE (ALLBZY(1,ICTR),I,TIME)
      RETURN
      END
C
C****
      SUBROUTINE UPDATE(STAT,VAL,TIME)
C--------UPDATE TIME AVERAGED STATISTIC
      INTEGER VAL
      DIMENSION STAT(3)
      DELTA = TIME - STAT(3)
      STAT(1) = STAT(1) + VAL * DELTA
      STAT(2) = STAT(2) + VAL**2 * DELTA
      STAT(3) = TIME
      RETURN
      END
C
C****
      SUBROUTINE MOVEJB
C--------JOB MOVER
      COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
      DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
      INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
      REAL JATIME,JSTIME,JDEP
      COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
      DIMENSION
     X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
     X  ,SCSD(10),SCD1(10),SCTS(10)
      INTEGER SCSM,SCSD
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
      IF (OLDCTR .GT. 1) GO TO 510
C--------GENERATE NEW ARRIVAL
```

```
C--------SET UP FOR NEXT ARRIVAL, GET FREE NODE, KLUDGE COUNT
       JCL = JPC(JOBNO)
       NJP(JCL) = MIN0(NJP(JCL)+1, MJP(JCL))
       JDEP(JOBNO) = TIME + ARRIV(JCL)
       JSTIME(JOBNO) = 0.0
       JOBNO = JRLINK(FSL)
       IF (JOBNO .EQ. FSL) CALL ERROR(4)
       JPC(JOBNO) = JCL
       JSTIME(JOBNO) = 0.0
       SCNBS(1) = SCNBS(1) + 2
  510 CONTINUE
C--------FIND NEW CENTER AND UPDATE ITS STATISTICS
       NEWCTR = OLDCTR
       IF (JSTIME(JOBNO) .LT. 1.0E-6) NEWCTR = NEXTC(OLDCTR)
       IF (NEWCTR .EQ. NSC) GO TO 550
       CALL STAT(NEWCTR)
C--------ENTER JOB AT NEW CENTER, SCHEDULE OLD AND NEW CENTERS
       CALL ENTRJB
       IF (OLDCTR .NE. 1) CALL SLCTJB(OLDCTR)
       CALL SLCTJB(NEWCTR)
       RETURN
  550 CONTINUE
C--------REMOVE JOB FROM SYSTEM, SCHEDULE OLD CENTER
       CALL RMOVJB
       CALL SLCTJB(OLDCTR)
       RETURN
       END
C
C****
       SUBROUTINE ENTRJB
C--------ENTER JOB AT NEW SERVICE CENTER
       COMMON /BLK3/
      X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
       INTEGER OLDCTR
       DIMENSION JLLINK(515),JRLINK(515),JPC(515),
      X   JATIME(515),JSTIME(515),JDEP(515)
       DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
      X   SCMAXQ(10),SCHDIA(10)
       INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
       REAL JATIME,JSTIME,JDEP
       COMMON /BLK4/
      X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
       DIMENSION IDATA(34000),DATA(34000)
       EQUIVALENCE (IDATA,DATA,JLLINK)
       EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
      X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
       DIMENSION
      X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
      X  ,SCSD(10),SCD1(10),SCTS(10)
       INTEGER SCSM,SCSD
```

```
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
C--------ENTER ARRIVAL TIME AND SERVICE TIME IF JOB IS
C--------FINISHED AT OLDCTR.
      IF (JSTIME(JOBNO) .GE. 1.0E-6) GO TO 560
      JATIME(JOBNO) = TIME
      JSTIME(JOBNO) = SERVIS(NEWCTR)
  560 CONTINUE
C--------REMOVE JOB FROM OLD CENTER QUEUE
      J = JRLINK(JOBNO)
      K = JLLINK(JOBNO)
      JRLINK(K) = J
      JLLINK(J) = K
      SCNBS(OLDCTR) = SCNBS(OLDCTR) - 1
C--------PUT JOB INTO INACTIVE QUEUE AT NEW CENTER
      J = SCTL(NEWCTR)
      K = JRLINK(J)
      JLLINK(JOBNO) = J
      JRLINK(JOBNO) = K
      JRLINK(J) = JOBNO
      JLLINK(K) = JOBNO
      IF (SCHDIA(NEWCTR) .EQ. NEWCTR) SCHDIA(NEWCTR) = JOBNO
C--------BUMP QUEUE SIZE
      SCQSIZ(NEWCTR) = SCQSIZ(NEWCTR) + 1
      IF (ILIST .GE. 40) WRITE(6,575) TIME,JOBNO,JPC(JOBNO),
     X   JATIME(JOBNO),JSTIME(JOBNO),OLDCTR,SCNBS(OLDCTR),
     X   SCQSIZ(OLDCTR),NEWCTR,SCNBS(NEWCTR),SCQSIZ(NEWCTR)
  575 FORMAT(6H ENTER,F12.3,2I10,2F12.3,6I10)
C--------PREEMPT IF NECESSARY
      IF (SCSM(NEWCTR) .LT.0) CALL PREMPT
      RETURN
      END
C
C****
      SUBROUTINE PREMPT
C--------LET NEW JOB PREEMPT OLD ONE AT NEWCTR, IF ALL
C--------SERVERS ARE BUSY.
      COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
      DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
      INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
      REAL JATIME,JSTIME,JDEP
      COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
```

```
     X    (JDEP,SCQSIZ),(JSTIME,SCNBS)
      DIMENSION
     X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
     X  ,SCSD(10),SCD1(10),SCTS(10)
      INTEGER SCSM,SCSD
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
      INTEGER PRI
      IF (SCNBS(NEWCTR).LT.SCNS(NEWCTR)) RETURN
      K = IABS(SCSM(NEWCTR))
      IF (K.EQ.1 .AND. NJPC.EQ.1) RETURN
      IF (K.EQ.3 .AND. NJPC.EQ.1) RETURN
C--------SEARCH ACTIVE QUEUE FOR LOWEST PRIORITY JOB
      PRI = 1000
      VALUE = 0.0
      IF (K.EQ.4) VALUE = XLARGE
      J = SCHD(NEWCTR)
      JEND = SCHDIA(NEWCTR)
  800 CONTINUE
      IF (J .EQ. JEND) GO TO 850
C--------USE SCHEDULING METHOD
      GO TO (810,820,810,830),K
C--------FCFS,RR
  810 IF (PRI .LT. JPC(J)) GO TO 840
      JOBNO = J
      PRI = JPC(J)
      GO TO 840
C--------SPTF
  820 IF (PRI.LT.JPC(J) .OR. (PRI.EQ.JPC(J) .AND. JDEP(J)-TIME
     X .LT.VALUE))GO TO 840
      JOBNO = J
      VALUE = JDEP(J)-TIME
      PRI = JPC(J)
      GO TO 840
C--------LPTF
  830 IF (PRI.LT.JPC(J) .OR. (PRI.EQ.JPC(J) .AND. JDEP(J)-TIME
     X    .GT.VALUE))      GO TO 840
      JOBNO = J
      VALUE = JDEP(J) - TIME
      PRI = JPC(J)
C--------ADVANCE POINTER
  840 J = JRLINK(J)
      GO TO 800
C--------SEE IF THIS JOB SHOULD BE PREEMPTED
  850 J = SCTL(NEWCTR)
      GO TO (860,870,860,875),K
C--------FCFS,RR
  860 IF (PRI .GE. JPC(J)) RETURN
      GO TO 880
C--------SPTF
  870 IF (PRI .GT. JPC(J) .OR.(PRI .EQ. JPC(J) .AND. VALUE .LE.
```

```
     X    JSTIME(J))) RETURN
      GO TO 880
C--------LPTF
  875 IF (PRI.GT.JPC(J) .OR. (PRI.EQ.JPC(J) .AND. VALUE.GE.
     X           JSTIME(J))) RETURN
C--------PREEMPT, SWAP JOBS
C--------TAKE J OUT OF INACTIVE QUEUE
  880 L = JLLINK(J)
      M = JRLINK(J)
      JLLINK(M) = L
      JRLINK(L) = M
      IF (SCHDIA(NEWCTR) .EQ. J) SCHDIA(NEWCTR) = M
C--------PUT J INTO ACTIVE QUEUE
      L = JRLINK(JOBNO)
      JRLINK(J) = L
      JLLINK(J) = JOBNO
      JRLINK(JOBNO) = J
      JLLINK(L) = J
C--------REMOVE JOBNO FROM ACTIVE QUEUE
      L = JLLINK(JOBNO)
      M = JRLINK(JOBNO)
      JLLINK(M) = L
      JRLINK(L) = M
C--------PUT JOBNO INTO INACTIVE QUEUE
      N = SCHDIA(NEWCTR)
      L = JLLINK(N)
      JRLINK(JOBNO) = N
      JLLINK(JOBNO) = L
      JRLINK(L) = JOBNO
      JLLINK(N) = JOBNO
      SCHDIA(NEWCTR) = JOBNO
      JSTIME(JOBNO) = JDEP(JOBNO)-TIME+JSTIME(JOBNO)
C--------ASSIGN PROPER DEPARTURE TIME AND REMAINING SERVICE TIME
      GO TO (883,883,887,883),K
C--------FCFS,SPTF,LPTF
  883 JDEP(J) = TIME+JSTIME(J)
      JSTIME(J) = 0.0
      GO TO 889
C--------RR
  887 SERV = AMIN1(JSTIME(J),SCTS(NEWCTR))
      JDEP(J) = TIME+SERV
      JSTIME(J) = JSTIME(J)-SERV
  889 CONTINUE
      IF (ILIST .GE. 40) WRITE(6,890) TIME,JOBNO,JPC(JOBNO),
     X   JATIME(JOBNO),JSTIME(JOBNO),JDEP(JOBNO),NEWCTR,
     X   SCNBS(NEWCTR),SCQSIZ(NEWCTR)
  890 FORMAT(5H PRMT,F12.3,2I10,3F12.3,3I10)
      RETURN
      END
C
```

```
C****
      SUBROUTINE SLCTJB(CTR)
C--------SERVICE NEXT JOB ACCORDING TO PRIORITY AND SCHEDULING
C--------METHOD AT THIS SERVICE CENTER.
      INTEGER CTR
      COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
      DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
      INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
      REAL JATIME,JSTIME,JDEP
      COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
      DIMENSION
     X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
     X  ,SCSD(10),SCD1(10),SCTS(10)
      INTEGER SCSM,SCSD
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
      IF (SCNBS(CTR) .GE. SCNS(CTR)) RETURN
      IF (SCQSIZ(CTR) .LE. 0) RETURN
      KK = IABS(SCSM(CTR))
      PRI = -1
      VALUE = XLARGE
      IF (KK.EQ.4) VALUE = -1.0
C--------SEARCH INACTIVE QUEUE FOR NEXT JOB
      J = SCHDIA(CTR)
      IF (J .EQ. CTR) RETURN
  600 CONTINUE
      IF (J .EQ. CTR) GO TO 650
C--------USE SCHEDULING METHOD
      GO TO (610,620,610,630),KK
C--------FCFS,RR
  610 IF (PRI.GT.JPC(J) .OR. (PRI.EQ.JPC(J) .AND. JATIME(J)
     X             .GT.VALUE)) GO TO 640
      JOBNO = J
      VALUE = JATIME(J)
      PRI = JPC(J)
      IF (NJPC .EQ. 1 .AND. SCSM(CTR) .GT. 0) GO TO 650
      GO TO 640
C--------SPTF
  620 IF (PRI.GT.JPC(J) .OR. (PRI.EQ.JPC(J) .AND. JSTIME(J)
     X             .GT.VALUE)) GO TO 640
      JOBNO = J
```

```
      VALUE = JSTIME(J)
      PRI = JPC(J)
      GO TO 640
C---------LPTF
  630 IF (PRI.GT.JPC(J) .OR. (PRI.EQ.JPC(J) .AND. JSTIME(J)
     X              .LT.VALUE)) GO TO 640
      JOBNO = J
      VALUE = JSTIME(J)
      PRI = JPC(J)
C--------ADVANCE POINTER
  640 J = JRLINK(J)
      GO TO 600
C--------HAVE FOUND JOB TO BE PROCESSED. REMOVE FROM INACTIVE
C--------QUEUE.
  650 SCQSIZ(CTR) = SCQSIZ(CTR) - 1
      J = JLLINK(JOBNO)
      K = JRLINK(JOBNO)
      JRLINK(J) = K
      JLLINK(K) = J
      IF (JOBNO .EQ. SCHDIA(CTR)) SCHDIA(CTR) = K
C---------ASSIGN PROPER DEPARTURE TIME AND REMAINING SERVICE TIME
      GO TO (660,660,670,660),KK
C--------FCFS,SPTF,LPTF
  660 JDEP(JOBNO) = TIME + JSTIME(JOBNO)
      JSTIME(JOBNO) = 0.0
      GO TO 675
C---------RR
  670 SERV = AMIN1(JSTIME(JOBNO),SCTS(CTR))
      JDEP(JOBNO) = TIME + SERV
      JSTIME(JOBNO) = JSTIME(JOBNO) - SERV
  675 CONTINUE
C--------PUT JOB ON ACTIVE QUEUE
      SCNBS(CTR) = SCNBS(CTR) +1
      J = SCHDIA(CTR)
      K = JLLINK(J)
      JRLINK(JOBNO) = J
      JLLINK(JOBNO) = K
      JRLINK(K) = JOBNO
      JLLINK(J) = JOBNO
      IF (ILIST .GE. 40) WRITE (6,680) TIME,JOBNO,JPC(JOBNO),
     X   JATIME(JOBNO),JSTIME(JOBNO),JDEP(JOBNO),CTR,SCNBS(CTR),
     X   SCQSIZ(CTR)
  680 FORMAT(5H SLCT,F12.3,2I10,3F12.3,3I10)
      RETURN
      END
C
C****
      SUBROUTINE RMOVJB
C--------REMOVE JOB FROM SYSTEM
      COMMON /BLK3/
```

```
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
      DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
      INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
      REAL JATIME,JSTIME,JDEP
      COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
      DIMENSION
     X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
     X  ,SCSD(10),SCD1(10),SCTS(10)
      INTEGER SCSM,SCSD
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
C--------RESTORE NODE TO FSL
      J = JRLINK(JOBNO)
      K = JLLINK(JOBNO)
      JRLINK(K) = J
      JLLINK(J) = K
      J = JRLINK(FSL)
      JRLINK(FSL) = JOBNO
      JLLINK(J) = JOBNO
      JRLINK(JOBNO) = J
      JLLINK(JOBNO) = FSL
      SCNBS(OLDCTR) = SCNBS(OLDCTR) -1
      J = JPC(JOBNO)
      NJP(J) = NJP(J) - 1
      IF (ILIST .GE. 40) WRITE(6,701) TIME,JOBNO,J,NJP(J),
     X   OLDCTR,SCNBS(OLDCTR),SCQSIZ(OLDCTR)
  701 FORMAT(5H RMOV, F12.3,6I10)
      RETURN
      END
C
C****
      FUNCTION NEXTC(ICTR)
C--------FIND NEXT CENTER TO ROUTE JOB TO
      COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION
     X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
     X  ,SCSD(10),SCD1(10),SCTS(10)
      INTEGER SCSM,SCSD
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
      CALL RANDOM(X)
```

```
      DO 900   NEXTC=1,NSC
      IF (X.LE. ROUTE(ICTR,NEXTC)) RETURN
  900 CONTINUE
      CALL  ERROR(3)
      RETURN
      END
C
C****
      FUNCTION SERVIS(NEWCTR)
C--------COMPUTE SERVICE TIME AT NEW CENTER
      DIMENSION
     X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
     X  ,SCSD(10),SCD1(10),SCTS(10)
      INTEGER SCSM,SCSD
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
      I = SCSD(NEWCTR)
      GO TO (10,20,30,40),I
C--------EXPONENTIAL
   10 CALL RANDOM(X)
      SERVIS = -SCMST(NEWCTR)*ALOG(X)
      RETURN
C--------HYPER-EXPONENTIAL
   20 CALL RANDOM(X)
      PAR = 0.5/SCD1(NEWCTR)
      IF ( X .GT. SCD1(NEWCTR)) PAR = 0.5/(1.0-SCD1(NEWCTR))
      CALL RANDOM(X)
      SERVIS = -PAR * SCMST(NEWCTR) * ALOG(X)
      RETURN
C--------CONSTANT
   30 SERVIS = SCMST(NEWCTR)
      RETURN
C--------HYPO-EXPONENTIAL
   40 SERVIS = 0.0
      K = IFIX(SCD1(NEWCTR)+0.001)
      DO 45 J=1,K
      CALL RANDOM(X)
   45 SERVIS = SERVIS - ALOG(X)
      SERVIS = SERVIS * SCMST(NEWCTR) / SCD1(NEWCTR)
      RETURN
      END
C
C****
      SUBROUTINE RANDOM(X)
C--------RETURN RANDOM NUMBER FROM UNIF. DISTRIBUTION ON (0,1)
      COMMON /BLK2/ SEED,LOOP,LIMIT
      INTEGER SEED,DUMMY(2)
      REAL*8 ZOT,ONE
      EQUIVALENCE (ZOT,DUMMY(1))
      DATA ZOT/Z4600000000000000/,ONE/1.0/
      REAL LIMIT
```

```
      SEED = SEED*314159269 + 453806245
      DUMMY(2) = SEED
      X = ZOT * ONE
      RETURN
      END
C
C****
      FUNCTION ARRIV(CLS)
C--------COMPUTE TIME TO NEXT ARRIVAL FOR THIS CLASS
      INTEGER CLS
      DIMENSION
     X SCSM(10),SCMST(10),ROUTE(10,10),PCMAT(5),MJP(5),NJP(5)
     X  ,SCSD(10),SCD1(10),SCTS(10)
      INTEGER SCSM,SCSD
      COMMON /BLK5/ SCSM,SCTS,SCSD,SCD1,SCMST,ROUTE,PCMAT,MJP,NJP
      ARRIV = 1.0E15
      IF (NJP(CLS) .GE. MJP(CLS)) RETURN
      CALL RANDOM(X)
      ARRIV = -PCMAT(CLS) * ALOG(X)
      RETURN
      END
C
C******************
      SUBROUTINE SAMPLE
C--------SAMPLE QUEUE LENGTHS IF TIME TO
      COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
      DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
      INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
      REAL JATIME,JSTIME,JDEP
      COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
      COMMON /BLK7/
     X NDATA,T2SAMP,DELSAM,ISTDAT,NSTOR,N22N,NP2,NS,PT
      DIMENSION ITEMP(10)
  700 CONTINUE
      IF (NDATA .GE. N22N) RETURN
      IF (TIME .LT. T2SAMP) RETURN
      NDATA = NDATA + 1
      LIM = NSC - 1
      DO 720 I=2,LIM
      ITEMP(I-1) = SCQSIZ(I) + SCNBS(I)
```

```
      IF (ITEMP(I-1) .GE. SCMAXQ(I)) ITEMP(I-1) = SCMAXQ(I) -1
  720 CONTINUE
      CALL CSSTOR(ITEMP,NDATA,SCMAXQ(2),LIM,IDATA(ISTDAT),NSTOR)
      T2SAMP = T2SAMP + DELSAM
      GO TO 700
      END
C
C**************
C         COMPACT STORAGE OF MULTIPLE INTEGER ARRAYS.
C         THESE ROUTINES LET ONE COMPACTLY STORE THE N-TH
C              INSTANCES OF A SET OF INTEGER VARIABLES OR
C              RETRIEVE THE N INSTANCES OF A PARTICULAR VARIABLE
C         NWSTOR - NO. OF WORDS (IN A GROUP) TO STORE 1 OCCURRENCE
C              OF EACH VARIABLE
C         NVAR - NO. OF VARIABLES TO BE SAMPLED
C         IVARMX(I) - MAXIMUM VALUE OF I-TH VARIABLE
C         IWVAR(I) - WORD WITHIN GROUP CONTAINING VARIABLE I
C         IDR(I) - DIVISOR USED TO STRIP OFF STUFF TO RIGHT OF
C              VALUE I
C         IWSTOR(J) - NO. OF VARIABLES STORED IN WORD I OF A GROUP
C         ISTOR - ARRAY WHERE DATA IS PACKED
C         IVAL - I/O ARRAY
C
C
C
C****
      SUBROUTINE CSINIT(NVAR,IVARMX,NWST)
C--------CSINIT - COMPACT STORAGE INITIALIZER
      DIMENSION IVARMX(NVAR)
      DIMENSION IWSTOR(3),IWVAR(10),IDR(10)
      COMMON /BLK6/ NWSTOR,IWSTOR,IWVAR,IDR
      BIG = FLOAT(2**31 - 1)
      IVARS = 1
      MAXNWS = 3
      IADD = 0
      DO 1090 NWSTOR=1,MAXNWS
      IMAX = 1
      DO 1050 IV=IVARS,NVAR
      IVAR = IV
      IF (FLOAT(IMAX) * FLOAT(IVARMX(IVAR)) .GE. BIG) GO TO 1060
      IWVAR(IVAR) = NWSTOR
      IMAX = IMAX * IVARMX(IVAR)
      IF (IVAR .EQ. NVAR) IADD = 1
 1050 CONTINUE
 1060 IWSTOR(NWSTOR) = IVAR - IVARS + IADD
      IMAX = 1
      KLIM = IVAR -1 + IADD
      DO 1070 K=IVARS,KLIM
      J = KLIM + IVARS - K
      IDR(J) = IMAX
```

```
      IMAX = IMAX * IVARMX(J)
 1070 CONTINUE
      IVARS = IVAR
      IF (IADD .GT. 0) GO TO 1100
 1090 CONTINUE
      IF (IVAR .NE. NVAR) STOP
 1100 NWST = NWSTOR
      WRITE (6,1110) NWSTOR,(IWSTOR(I),I=1,NWSTOR),
     X   (IWVAR(I),I=1,NVAR),(IDR(I),I=1,NVAR)
 1110 FORMAT(24H0COMPACT STORAGE PARS     ,/,(12I10))
      RETURN
      END
C
C****
      SUBROUTINE CSSTOR(IVAL,N,IVARMX,NVAR,ISTOR,NSTOR)
C--------CSSTOR - STORE VALUES
      DIMENSION IVAL(NVAR),IVARMX(NVAR),ISTOR(NSTOR)
      DIMENSION IWSTOR(3),IWVAR(10),IDR(10)
      COMMON /BLK6/ NWSTOR,IWSTOR,IWVAR,IDR
      IVAR = 0
      DO 1190 I=1,NWSTOR
      IWRD = (N-1)*NWSTOR+I
      ISTOR(IWRD) = 0
      LIM = IWSTOR(I)
      DO 1150 J=1,LIM
      IVAR =IVAR +1
      ISTOR(IWRD) = ISTOR(IWRD) * IVARMX(IVAR) + IVAL(IVAR)
 1150 CONTINUE
 1190 CONTINUE
      RETURN
      END
C
C****
      SUBROUTINE CSRTRV(I,IVAL,N,IVARMX,NVAR,ISTOR,NSTOR)
C--------CSRTRV - RETRIEVE VALUES
      DIMENSION IVAL(N),IVARMX(NVAR),ISTOR(NSTOR)
      DIMENSION IWSTOR(3),IWVAR(10),IDR(10)
      COMMON /BLK6/ NWSTOR,IWSTOR,IWVAR,IDR
      DO 1290 J=1,N
      IWRD = (J-1)*NWSTOR +IWVAR(I)
      IWRD = ISTOR(IWRD)/IDR(I)
      IVAL(J) = IWRD - IWRD/IVARMX(I) * IVARMX(I)
 1290 CONTINUE
      RETURN
      END
C
C*****
      SUBROUTINE FFTRTV(I,IVAL,N,IVARMX,NVAR,ISTOR,NSTOR)
C--------FFTRTV - SPECIAL ENTRY TO RETRIEVE VALUES FOR
C--------DOING A DOUBLE LENGTH REAL FFT.
```

```
      DIMENSION IVAL(N),IVARMX(NVAR),ISTOR(NSTOR)
      DIMENSION IWSTOR(3),IWVAR(10),IDR(10)
      COMMON /BLK6/ NWSTOR,IWSTOR,IWVAR,IDR
      DO 1390 J=1,N
      IWRD = (J-1)*NWSTOR+IWVAR(I)
      IWRD = ISTOR(IWRD)/IDR(I)
      IWRD = IWRD - IWRD/IVARMX(I)*IVARMX(I)
      IF (J/2*2 .EQ. J) GO TO 1350
      K = J/2+1
      GO TO 1390
 1350 K = N/2+J/2
 1390 IVAL(K) = IWRD
      RETURN
      END
C
C****
      SUBROUTINE OPUT
C--------UPDATE AND PRINT FINAL RESULTS
      COMMON /BLK1/
     X QSTAT(3,10),BSTAT(3,10),ALLBZY(3,10),SUMBZY(3,10)
      COMMON /BLK2/ SEED,LOOP,LIMIT
      INTEGER SEED
      REAL LIMIT
      COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
      DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
      INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
      REAL JATIME,JSTIME,JDEP
      COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
 7000 FORMAT(1H1 , 20HSTATISTICAL SUMMARY      )
 7001 FORMAT(' CENTER NUMBER        MEAN           ',
     X   'STD DEV          COEF VAR')
 7002 FORMAT( // 24X, 12H QUEUE SIZE )
 7003 FORMAT( // 24X, 20H NO. OF BUSY SERVERS  )
 7004 FORMAT( // 24X, 20H SOME SERVERS BUSY    )
 7005 FORMAT( // 24X, 20H ALL SERVERS BUSY     )
 7006 FORMAT(6H LOOP=  ,I10)
 7007 FORMAT(24H FINAL NO. BUSY SERV.      ,10I10)
 7008 FORMAT(18H FINAL QUEUE SIZES     ,6X,10I10)
C
      IF (ILIST .LT. 15) RETURN
```

```
      NS = NSC -1
      DO 7100 I=2,NS
      CALL STAT(I)
 7100 CONTINUE
C
      WRITE(6,7000)
      WRITE(6,7006) LOOP
      WRITE (6,7007) (SCNBS(I),I=1,NSC)
      WRITE (6,7008) (SCQSIZ(I),I=1,NSC)
      WRITE(6,7002)
      WRITE (6,7001)
      DO 7200 I=2,NS
 7200 CALL PRNTL(QSTAT(1,I),TIME,I)
      WRITE(6,7003)
      WRITE (6,7001)
      DO 7300 I=2,NS
 7300 CALL PRNTL(BSTAT(1,I),TIME,I)
      WRITE(6,7004)
      WRITE (6,7001)
      DO 7400 I=2,NS
 7400 CALL PRNTL(SUMBZY(1,I),TIME,I)
      WRITE(6,7005)
      WRITE (6,7001)
      DO 7500 I=2,NS
 7500 CALL PRNTL(ALLBZY(1,I),TIME,I)
      RETURN
      END
C
C****
      SUBROUTINE PRNTL(STAT,TIME,N)
C--------PRINT A LINE OF STATS
      DIMENSION STAT(3)
 7900 FORMAT(I15, 3F16.8)
      STAT(1) = STAT(1) / TIME
      STAT(2) = SQRT(STAT(2)/TIME - STAT(1)**2)
      CV = STAT(2) / STAT(1)
      WRITE (6,7900) N,STAT(1),STAT(2),CV
      RETURN
      END
C
C******************
      SUBROUTINE STAT1
C--------TIME-SERIES ANALYSIS AND OUTPUT
      COMMON /BLK1/
     X QSTAT(3,10),BSTAT(3,10),ALLBZY(3,10),SUMBZY(3,10)
      COMMON /BLK3/
     X ILIST,TIME,XLARGE,JOBNO,OLDCTR,NEWCTR,NSC,NJPC
      INTEGER OLDCTR
      DIMENSION JLLINK(515),JRLINK(515),JPC(515),
     X   JATIME(515),JSTIME(515),JDEP(515)
```

```
      DIMENSION SCQSIZ(10),SCNS(10),SCNBS(10),SCHD(10),SCTL(10),
     X   SCMAXQ(10),SCHDIA(10)
      INTEGER SCQSIZ,SCNS,SCNBS,SCHD,SCTL,SCHDIA,SCMAXQ,FSL
      REAL JATIME,JSTIME,JDEP
      COMMON /BLK4/
     X SCNS,SCMAXQ,JLLINK,JRLINK,JPC,JATIME,JSTIME,JDEP,FSL
      DIMENSION IDATA(34000),DATA(34000)
      EQUIVALENCE (IDATA,DATA,JLLINK)
      EQUIVALENCE (JRLINK,SCHD),(JLLINK,SCTL),(JPC,SCHDIA),
     X   (JDEP,SCQSIZ),(JSTIME,SCNBS)
      COMMON /BLK7/
     X NDATA,T2SAMP,DELSAM,ISTDAT,NSTOR,N22N,NP2,NS,PT
      DIMENSION FREQ(100)
      EQUIVALENCE(FREQ,QSTAT)
      DIMENSION LINE(100),HTIT(3),PTIT(4)
C ***
      INTEGER TEXT(20), BBITS(1)/1/,IA(5)/5*0/
      REAL XA(2)/2*0./,TITLE2(20),TITLE1(20)
      DATA NBBITS/1/,TITLE2/'LOG(POWER) VS. FREQUENCY'/
      DATA TITLE1/'OCCURRENCES VS. QUEUE LENGTH'/
C ***
      DATA HTIT(1),HTIT(2),HTIT(3) / 4H HIS ,4HTOGR ,4HAM    /
      DATA PTIT(1),PTIT(2),PTIT(3),PTIT(4)/4H POW,4HER S,4HPECT,
     X  4HRUM /
  810 FORMAT(24H1QUEUE SIZE DATA, CENTER    ,I5,/,(1X,40I3))
  830 FORMAT(1X,5HMIN.= ,F12.3,5X,5HMAX.= ,F12.3,5X,
     X 16HNYQUIST FREQ. =      ,F12.3,5X,10HNO. POINTS   ,I8)
  850 FORMAT(24H1QUEUE SIZE PSD, CENTER        ,I5,/,(1X,26F5.2))
  860 FORMAT(12H DIAGNOSTICS  ,/,(1X,10E13.3))
  880 FORMAT(20A4)
      LIM = NSC - 1
      FNYQ = 0.5/DELSAM
      N2 = NDATA/2
      N21 = N2 +1
      DO 1000 I=2,LIM
      IF (ILIST .LT. 30) GO TO 890
C--------RETRIEVE AND PRINT DATA IN ACTUAL ORDER
      CALL CSRTRV(I-1,IDATA,NDATA,SCMAXQ(2),LIM,IDATA(ISTDAT)
     X   ,NSTOR)
      WRITE(6,810) I,(IDATA(J),J=1,NDATA)
  890 CONTINUE
C--------RETRIEVE DATA FOR FFT
      CALL FFTRTV(I-1,IDATA,NDATA,SCMAXQ(2),LIM,IDATA(ISTDAT)
     X   ,NSTOR)
      IF (ILIST .GE. 45) WRITE(6,810) I,(IDATA(J),J=1,NDATA)
      DO 905 J=1,NDATA
  905 DATA(J) = IDATA(J)
      CALL MNMX(DATA,NDATA,DMIN,DMAX)
      WRITE(6,830) DMIN,DMAX,FNYQ,NDATA
C--------COMPUTE AND LIST HISTOGRAM
```

```
      IF (ILIST .LT. 20) GO TO 910
      CALL GROOP(DATA,NDATA,FREQ,16,DMIN,DMAX)
      CALL MNMX(FREQ,16,YMIN,YMAX)
      CALL PPLOT(FREQ,16,DMIN,DMAX,YMIN,YMAX,LINE,1,HTIT,3)
  910 CONTINUE
C
C--------GET POWER SPECTRUM
C
      IF (ILIST .LT. 25) GO TO 1000
C--------TAKE OUT DC BIAS AND USE COSINE WINDOW
      CALL DCBIAS(DATA,NDATA)
      IF (ILIST .GE. 45) WRITE (6,860) (DATA(J),J=1,NDATA)
      CALL COSWOE(DATA,NDATA,PT)
      IF (ILIST .GE. 45) WRITE (6,860) (DATA(J),J=1,NDATA)
      DO 920 J=1,N2
      K=N2-J+1
  920 DATA(N2+K+1) = DATA(N2+K)
C--------DO REAL FFT, GET MODIFIED PSD, SMOOTH IT
      CALL FFTPSD(DATA,DATA(N2+2),N2,N21,NP2)
      IF (ILIST .GE. 45) WRITE (6,860) (DATA(J),J=1,N21)
      INC = MAX0(N2/400,1)
      NPLT = N21/INC
      IF (NS.GT.0) CALL SMOOTH(DATA,DATA(N21+1),N21,NS)
      IF (ILIST .GE. 45) WRITE (6,860) (DATA(J),J=1,N21)
C--------DISPLAY THE LOG POWER SPECTRUM
      DO 930 J=1,NPLT
      K=INC*(J-1)+1
      DATA(J) = DATA(K)
      IF (DATA(J) .LE. 0.0001) DATA(J) = 0.0001
      DATA(J) = ALOG10(DATA(J))
  930 CONTINUE
      CALL PPLOT(DATA,NPLT,0.0,FNYQ,0.0,4.0,LINE,1,PTIT,4)
C ***
      DO 935 J=1,NPLT
      KK = NPLT+J
      DATA(KK) = (J-1)*FNYQ/FLOAT(NPLT-1)
  935 CONTINUE
      NPB = 2*(NPLT+1)
      CALL UGEINT('CLEAR*',IDATA(NPB),3000)
      CALL UGELIN('STDY*',0.05,0.1,0,IDATA(NPB))
      CALL UGELIN('STDY*',0.05,0.56,1,IDATA(NPB))
      CALL UGELIN('STDY*',0.70,0.56,1,IDATA(NPB))
      CALL UGELIN('STDY*',0.70,0.1,1,IDATA(NPB))
      CALL UGELIN('STDY*',0.05,0.1,1,IDATA(NPB))
      CALL SCALE(DATA(NPLT+1),NPLT, 0.0, FNYQ, 0.125,0.66)
      CALL SCALE(DATA(1),NPLT,0.0,4.0,0.175,0.52)
      READ(5,880) (TEXT(KK),KK=1,20)
      CALL UGETXT('XSPACING=0.01*',0.065,0.115,TEXT(1),60,
     X   IDATA(NPB))
      CALL UGETXT('XSPACING=0.01*',0.2,0.137, TITLE2(1),24,
```

```
     1    IDATA(NPB))
      CALL UGELNS('STDY*',DATA(NPLT+1),DATA(1), NPLT,BBITS(1),
     1    NBBITS,IDATA(NPB))
      CALL UGAXIS('VERT,LDEC=1*','STDY*','XSPACING=0.01*',0.125,
     1    0.175,0.52,0.0,4.0,5,IDATA(NPB))
      CALL UGAXIS('LDEC=1*','STDY*','XSPACING=0.01*',0.175,0.125,
     1    0.66,0.0,FNYQ,5,IDATA(NPB))
      XB = 0.23
      YB = 0.20
C ***
      CALL UGELIN('STDY*',XB+0.05,YB+0.117,0,IDATA(NPB))
      CALL UGELIN('STDY*',XB+0.05,0.56,1,IDATA(NPB))
      CALL UGELIN('STDY*',0.70,0.56,1,IDATA(NPB))
      CALL UGELIN('STDY*',0.70,YB+0.117,1,IDATA(NPB))
      CALL UGELIN('STDY*',XB+0.05,YB+0.117,1,IDATA(NPB))
      CALL UGETXT('XSPACING=0.007*',XB+0.175,YB+0.125,TITLE1(1),
     1    28,IDATA(NPB))
      CALL UGAXIS('VERT,LDEC=1*','STDY*','SMAL*',XB+0.095,
     1    YB+0.160,YB+0.355,YMIN,YMAX,5,IDATA(NPB))
      CALL UGAXIS('LDEC=1*','STDY*','XSPACING=0.007*',YB+0.160,
     1    XB+0.095,XB+0.43,DMIN,DMAX,6,IDATA(NPB))
      DEL = (DMAX-DMIN)/16
      DATA(1) = 0.0
      DO 940 KK=1,16
      DATA(2*KK) = FREQ(KK)
      DATA(2*KK+1) = FREQ(KK)
      KKK = NPLT+2*KK+1
      DATA(KKK-2) = (KK-1)*DEL+ DMIN
      DATA(KKK-1) = (KK-1)*DEL+ DMIN
  940 CONTINUE
      DATA(KKK) = DMAX
      DATA(KKK+1) = DMAX
      DATA(34) = 0.0
      CALL SCALE(DATA,34,YMIN,YMAX,YB+0.160,YB+0.355)
      CALL SCALE(DATA(NPLT+1),34,DMIN,DMAX,XB+0.095,XB+0.43)
      CALL UGELNS('STDY*',DATA(NPLT+1),DATA(1),34,BBITS(1),
     1    NBBITS,IDATA(NPB))
      CALL UGPICT('CLEAR*',0)
      CALL UGEPUT('ON*',0,IDATA(NPB))
C ***
      IF (ILIST .LT. 35) GO TO 990
      WRITE (6,850) I,(DATA(J),J=1,NPLT)
  990 CONTINUE
 1000 CONTINUE
      RETURN
      END
C
C*****
      SUBROUTINE GROOP(DATA,N,FREQ,NB,DMIN,DMAX)
C--------FREQUENCY DISTRIBUTION
```

```
      DIMENSION DATA(N),FREQ(NB)
      DO 10 I=1,NB
   10 FREQ(I) = 0
      DEL = (DMAX - DMIN) /NB
      DO 30 I=1,N
      DO 20 JJ=1,NB
      J=JJ
      IF (DATA(I) .LT. DMIN + J*DEL) GO TO 25
   20 CONTINUE
   25 FREQ(J) = FREQ(J) + 1
   30 CONTINUE
      RETURN
      END
C
C****
      SUBROUTINE PPLOT(ARR,NA,XMIN,XMAX,YMIN,YMAX,LINE,LI,TIT,NT)
C        PRINTER PLOT OF ARR ARRAY, WHICH HAS NA POINTS
C        XMIN,XMAX- MIN.,MAX. IN X DIRECTION (TOP TO BOT OF PAGE)
C        YMIN,YMAX- MIN.,MAX. IN Y DIRECTION (ACROSS PAGE)
C        LINE - SCRATCH ARRAY
C        LI - INDEX EVERY LI-TH POINT IN X-DIRECTION
C        TIT - TITLE WHICH HAS NT WORDS
      DIMENSION ARR(NA),TEMP(5),LINE(100),TIT(NT)
      DATA IBLNK,ICHAR / 4H     ,4HXXXX  /
    1 FORMAT(1H1,30A4)
    2 FORMAT(5X,1HX,5(F14.2,10X),1HY)
    3 FORMAT(1X,F8.2,6X,100A1,F15.3)
C--------SET UP Y AXIS LABELS
      NYP = 100
      DEL =  (YMAX-YMIN)/4
      DO 10 I=1,5
   10 TEMP(I) = YMIN + (I-1)*DEL
      DO 20 I=1,NYP
   20 LINE(I) = IBLNK
      INC = MAX0(LI,1)
      DEL =(XMAX-XMIN)/FLOAT(NA-1)
      LINECT = 100
C--------NOW PRODUCE PLOT
      DO 100 I=1,NA,INC
      X = XMIN + (I-1)*DEL
      J = NYP*(ARR(I)-YMIN)/(YMAX-YMIN) +1
      IF (J .LE. 0) J=1
      IF (J .GT. NYP) J=NYP
      LINE(J) = ICHAR
      LINECT = LINECT + 1
      IF (LINECT .LE. 55) GO TO 90
      WRITE (6,1) (TIT(K),K=1,NT)
      WRITE (6,2) (TEMP(K),K=1,5)
      LINECT = 0
  90  CONTINUE
```

```
      WRITE(6,3) X,(LINE(K),K=1,NYP),ARR(I)
      LINE(J) = IBLNK
  100 CONTINUE
      RETURN
      END
C
C*****
      SUBROUTINE MNMX(ARR,N,XMIN,XMAX)
C--------FIND MIN. AND MAX. OF ARRAY
      DIMENSION ARR(N)
      XMIN = ARR(1)
      XMAX = XMIN
      DO 100 I=1,N
      IF (ARR(I) .LE. XMAX) GO TO 50
      XMAX = ARR(I)
      GO TO 100
   50 IF (ARR(I) .GE. XMIN) GO TO 100
      XMIN = ARR(I)
  100 CONTINUE
      RETURN
      END
C
C****
      SUBROUTINE FFTSO(N,NM,AR,AI,INP,SC)
C--------FAST FOURIER TRANSFORM STARTING IN SERIAL ORDER
      DIMENSION AR(N),AI(N)
      NX = NM -1
      NSPH = N/2
      NSP = N
      RAD = 6.28318530
      NN = N-1
      K = 2
      RS = SIN(RAD/(N+N))
      DO 1 I=1,NX
      K=K+K
      CN = 1.0
      SN = 0.0
      CM = SN
      JJ = 0
      II = 1
      R = -(RS+RS)**2
      CD = -0.5*R
      RS = SIN(RAD/NSP)
      IF (INP .NE. 2) GO TO 2
      SM = -1.0
      SD = -RS
      GO TO 10
    2 SM = 1.0
      SD = RS
   10 NSP = NSPH
```

```
      NSPH = NSP/2
    4 NXO = II + NSP
      RR = AR(II)-AR(NXO)
      AR(II) = AR(II)+AR(NXO)
      RI = AI(II)-AI(NXO)
      AI(II) = AI(II)+AI(NXO)
      AR(NXO) = CN*RR-SN*RI
      AI(NXO) = SN*RR+CN*RI
      II = II+NSPH
      NXO = NXO+NSPH
      RR = AR(II)-AR(NXO)
      AR(II) = AR(II)+AR(NXO)
      RI = AI(II)-AI(NXO)
      AI(II) = AI(II)+AI(NXO)
      AR(NXO) = CM*RR-SM*RI
      AI(NXO) = SM*RR+CM*RI
      II = NXO+NSPH
      IF (II .LT. N) GO TO 4
      II = II-NN
      JJ = JJ+K
      IF (JJ .GE. N) GO TO 1
      CD = R*CN+CD
      CN = CN+CD
      SM = CN
      SD = R*SN+SD
      SN = SN+SD
      CM = SN
      IF (INP .NE. 2) GO TO 7
      SM = -SM
      GO TO 4
    7 CM = -CM
      GO TO 4
    1 CONTINUE
C
      DO 9 I=1,NN,2
      NXO = I+1
      RR = AR(I)-AR(NXO)
      AR(I) = AR(I)+AR(NXO)
      AR(I) = SC*AR(I)
      AR(NXO) = SC*RR
      RI = AI(I)-AI(NXO)
      AI(I) = AI(I)+AI(NXO)
      AI(I) = SC*AI(I)
      AI(NXO) = SC*RI
    9 CONTINUE
      RETURN
      END
C
C*****
      SUBROUTINE COSWOE(A,NPT,PT)
```

```
C--------APPLY COSINE WINDOW TO DATA IN EVEN-ODD ORDER.
C--------NPT IS POWER OF 2 = NO. DATA POINTS
      DIMENSION A(NPT)
  100 FORMAT(1X,22HNUMBER TAPERRED 0.FR =  ,I10)
      N = PT*NPT
      NPT2 = NPT/2
      WRITE (6,100) N
      PI = 3.141592
      DO 1 I=1,N
      J = I-1
      P = 0.5*(1.0 - COS((PI*J)/N))
      K=I/2
      IF (K*2 .NE. I) K = NPT2+K+1
      A(K) = A(K) * P
      NZ = NPT - J
      K = NZ/2
      IF (K*2 .NE. NZ) K=NPT2 + K + 1
    1 A(K) = A(K) * P
      RETURN
      END
C
C****
      SUBROUTINE DOUBLE(NT,N1,A,B,JJ)
C--------USED IN TRANSFORMING ARRAY OF SIZE NT = 2*N
      DIMENSION A(N1),B(N1)
      N=NT/2
      N2 = N/2+1
      PI = 3.14159265
      CC = 1.0
      SS = 0.0
      RS = SIN(PI/NT)
      R = -(RS+RS)**2
      CD = -0.5*R
      SD = SIN(PI/N)
      IF(JJ .EQ. 2) SD = -SD
      A(N1) = A(1)
      B(N1) = B(1)
      DO 1 I=1,N2
      J = N1-I+1
      AA = (A(I)+A(J))/2.0
      BB = (B(I)-B(J))/2.0
      C = (B(I)+B(J))/2.0
      D = (A(I)-A(J))/2.0
      RE = CC*C + SS*D
      RI = SS*C - CC*D
      A(I) = AA+RE
      A(J) = AA-RE
      B(I) = BB+RI
      B(J) = RI-BB
      CD = R*CC+CD
```

```
      CC = CC+CD
      SD = R*SS+SD
      SS = SS+SD
    1 CONTINUE
      RETURN
      END
C
C*****
      SUBROUTINE SMOOTH(A,D,N,NS)
C--------SMOOTH DATA A WITH NS POINT MOVING AVERAGE
      DIMENSION A(N),D(N)
      DO 10 I=1,N
   10 D(I) = A(I)
      L1 = NS/2
      L2 = L1+1
      SUM = 0.0
      DO 20 I=1,L1
   20 SUM = SUM+D(I)
      DO 30 I=1,L2
      J = L1+I
      SUM = SUM+D(J)
   30 A(I) = SUM/J
      L3 = L2+1
      L4 = N-L1
      DO 40 I=L3,L4
      J=I+L1
      K = I-L2
      SUM = SUM+D(J)-D(K)
   40 A(I) = SUM/NS
      L5 = L4+1
      DO 50 I=L5,N
      K = I-L2
      SUM = SUM-D(K)
   50 A(I) = SUM/(N-I+L2)
      RETURN
      END
C
C****
      SUBROUTINE DCBIAS(A,NO)
C--------REMOVE DC BIAS FROM DATA
      DIMENSION A(NO)
      SUM = 0.0
      DO 1 I=1,NO
    1 SUM = SUM + A(I)
      SUM = SUM/NO
      WRITE (6,10) SUM
   10 FORMAT(12H SAMPLE MEAN      ,F12.3)
      DO 2 I=1,NO
    2 A(I) = A(I)-SUM
      RETURN
```

```
      END
C
C****
      SUBROUTINE XCHANG(N,NM,A,B)
C--------UNSCRAMBLE 2 FFT ARRAYS AT ONCE
      DIMENSION A(N),B(N)
      DO 1 I=1,N
      IX = I-1
      CALL REVERS (NM,IX,IXR)
      L = IXR+1
      HOLD = A(I)
      A(I) = B(L)
    1 B(L) = HOLD
      RETURN
      END
C
C****
      SUBROUTINE FFTPSD(A,B,N,N1,NM)
C--------POWER SPECTRUM  OF 2*N DATA POINTS A,B
      DIMENSION A(N1),B(N1)
      N2 = 2*N
      NP2 = NM-1
      CALL FFTSO(N,NP2,A,B,2,1.0/N)
      CALL XCHANG(N,NP2,A,B)
      CALL DOUBLE(N2,N1,B,A,2)
      S = N2
      DO 50 I=1,N1
   50 A(I) = (B(I)**2+A(I)**2)*S
      A(1) = A(1)/2.0
      RETURN
      END
C
C*****
      SUBROUTINE REVERS(NM,IX,IXR)
C--------PERFORM BIT REVERSING OF ARRAY INDEX
      IXR = 0
      J = IX
      DO 10 K=1,NM
      IXR = 2*IXR
      L = J/2
      IF (L*2 .EQ. J) GO TO 5
      IXR = IXR + 1
    5 J = L
   10 CONTINUE
      RETURN
      END
C
C****
      SUBROUTINE ERROR(ITYPE)
C--------ISSUE ERROR MESSAGE
```

```
      GO TO (9010,9020,9030,9040,9050,9060),ITYPE
9010  WRITE(6,9011)
9011  FORMAT(25H EVENT QUEUE MESSED UP.          )
      GO TO 9999
9020  WRITE(6,9021)
 9021 FORMAT(24H CAN'T SAVE 2**N POINTS      )
      GO TO 9999
9030  WRITE(6,9031)
 9031 FORMAT(25H BAD ROUTE DISTRIBUTION            )
      GO TO 9999
9040  WRITE(6,9041)
9041  FORMAT(25H NO MORE FREE STORAGE.           )
      GO TO 9999
9050  WRITE(6,9051)
9051  FORMAT(25H BAD DATA BLOCK NO.              )
      GO TO 9999
9060  WRITE(6,9061)
9061  FORMAT(25H BAD INPUT BLOCK 1               )
 9999 CONTINUE
      CALL OPUT
      STOP
      END
C
C****
      SUBROUTINE SCALE(A,N,X,Y,W,Z)
C---------SCALE AN ARRAY A FROM (X,Y) TO (W,Z)
      DIMENSION A(N)
      F = (Z-W) / (Y-X)
      DO 10 I=1,N
      IF (A(I) .GT. Y) A(I) = Y
      IF (A(I) .LT. X) A(I) = X
      A(I) = W + F*(A(I) - X)
   10 CONTINUE
      WRITE (6,20) (A(J),J=1,N)
   20 FORMAT(1X, 13F10.3)
      RETURN
      END
```