TR 31
May 1974

AN ANALYTIC MODEL OF A LARGE SCALE INTERACTIVE SYSTEM INCLUDING THE EFFECTS OF FINITE MAIN MEMORY

by

ROBERT MCBURNIE BROWN

May 1974 TR 31

This report constituted the author's M. A. Thesis in Computer Science at The University of Texas at Austin.

This research was supported by the National Science Foundation under Grant GJ-1084.

DEPARTMENT OF COMPUTER SCIENCES

THE UNIVERSITY OF TEXAS AT AUSTIN

ACKNOWLEDGEMENTS

The author would like to thank Dr. James C. Browne, whose support and timely suggestions made this research possible. Dr. K. Mani Chandy is thanked for his patient guidance in the development of the model. Acknowledgement is also made to Dr. John H. Howard, who explained the complexities of the UT-2D event trace data. All three of these people contributed much to the quality of this written work. This research was supported in part by National Science Foundation Grant G J 1084, "Design and Analysis of Operating Systems".

R. M. B.

May, 1974

ABSTRACT

An analytic model is proposed for the analysis of the effects of finite main memory on a class of large scale interactive systems. This model uses a four phase method of analysis, including the division of the total system into an in-memory portion and an out-of-memory portion. The concept of superposition of these subsystems based on a given job field length (memory requirement) probability distribution and a given amount of main memory is introduced and discussed. The capability of the model to produce predictive results pertinent to systems analysis and design is demonstrated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SECTION I

INTRODUCTION TO THE RESEARCH

1.1 Introduction

The performance of a multi-access interactive computer system is dependent upon, and sometimes limited by, the capacity of its executable memory. This research analyses empirical data taken from an interactive computer system and constructs from this data a model which explicitly considers the finite size of executable memory, job load characteristics, system overheads, and swap times. The objectives of the modeling are the development of insight into the behavior of complex memory systems and the prediction of system behavior in parametrically defined environments. A non-paged memory system was chosen for this research because of the detailed data for CDC 6000 series equipment that was available from the UT-2D operating system event trace package and because of the relative lack of reported analytic work on non-paged systems. This non-paged memory can, in essence, be regarded as a segmented memory in which each user job is considered to be a segment. The analytic model produces predictions which include the resolution of executable memory into utilized, overhead, and idle fractions in addition to the more standard model predictions of response time, device utilizations, and system throughput. Relations between memory usage parameters and the standard system performance measures are developed.

1.2 Discussion of Previous Work

Research in the area of memory management is directed toward the discovery of efficient techniques for the allocation of the basic memory resources. The introduction of virtual memory systems changed the memory allocation problem into the paging problem. Much work has been reported on allocation techniques for virtual memory systems (9,16). Considerably less work has been reported on the memory management problems for non-paged systems although such systems are in widespread use. Analysis of non-paged systems is reported in (1,3,4,10,13) and in (2) Baskett, Browne, and Lan investigate the interaction of CPU scheduling and memory scheduling.

Advanced techniques for the mathematical evaluation and solution of queueing network models have made it possible to model complex, realistic computer systems analytically (5,6,8,14). The analysis of some classes of queuing networks has been simplified by concepts of local balance (7).

This research employs techniques of queueing network analysis and probability theory in order to investigate the relationships between job load characteristics such as job memory requirements, job I/O requirements, and job CPU requirements and system performance measures such as device utilizations, memory utilization, and response time. This work is applied to an interactive system having a non-paged memory system.

1.3 Description of the System

This research is based on data taken from a CDC 6400 system which has 64K of 60 bit words for main memory, half a million words of ECS (Extended Core Storage) for use as a fast swap area, four CDC 808 disks, and an 8-spindle CDC 841 disk system for auxiliary storage. The 6400 is supported by seven CDC peripheral processors (PP's). Two of these are dedicated to system functions and five are usually available for user job related tasks. Thornton (18) discusses the characteristics of the CDC equipment. The 6400 is part of a dual 6600/6400 system that is governed by a locally written operating system called UT-2D. This dual system is designed for both batch and interactive systems and the interactive system is handled primarily by the 6400. An interactive job is allowed one real-time second in main memory to complete its interaction. If the interaction is not completed in the second, the job is swapped to ECS in order to allow other interactive jobs to be scheduled. The basic strategy for memory scheduling is a one second (real time) round-robin schedule for all interactive jobs which are ready to run. The central processor is scheduled in a 16 millisecond round-robin fashion. A detailed description of the UT-2D system has been given by Howard (11).

1.4 Summary of the Sections

Section II presents a detailed description of the proposed analytic model of the system including the calculations involved.

Section III includes a brief description of the event trace package in the UT-2D operating system, an explanation of the methods used to reduce the magnetic tape traces of system activity, and examples of the reduced statistical data obtained from the event trace tapes.

Section IV presents the parameterization techniques and validation information for the model described in Section II.

Section V discusses experimental work with the model and the analysis of the results.

Section VI contains a summary of the results of the research, conclusions based on these results, and possible extensions of this work.

SECTION II

THE ANALYTIC MODEL

2.1 Introduction

The purpose of this modeling analysis is to achieve three basic goals: (A) To provide an accurate, but relatively simple, model of the CDC 6400 interactive system, including the restrictions of finite main memory; (B) To further simplify the model by maintaining constant values for those system characteristics which show only a slight variation; and (C) To establish and maintain a predictive capability to determine the effects on system performance of changes in either the job load characteristics or the system configuration.

Figure 1 shows a network representation of the overall system to be modeled. The two dotted boxes indicate those portions of the network which require the job to be in either main memory or ECS as indicated in the figure. Jobs in the swap delay queue are physically located in ECS but are logically (to the scheduler) located in main memory.

There are several problems in the analysis of this system by previously developed techniques. No effective

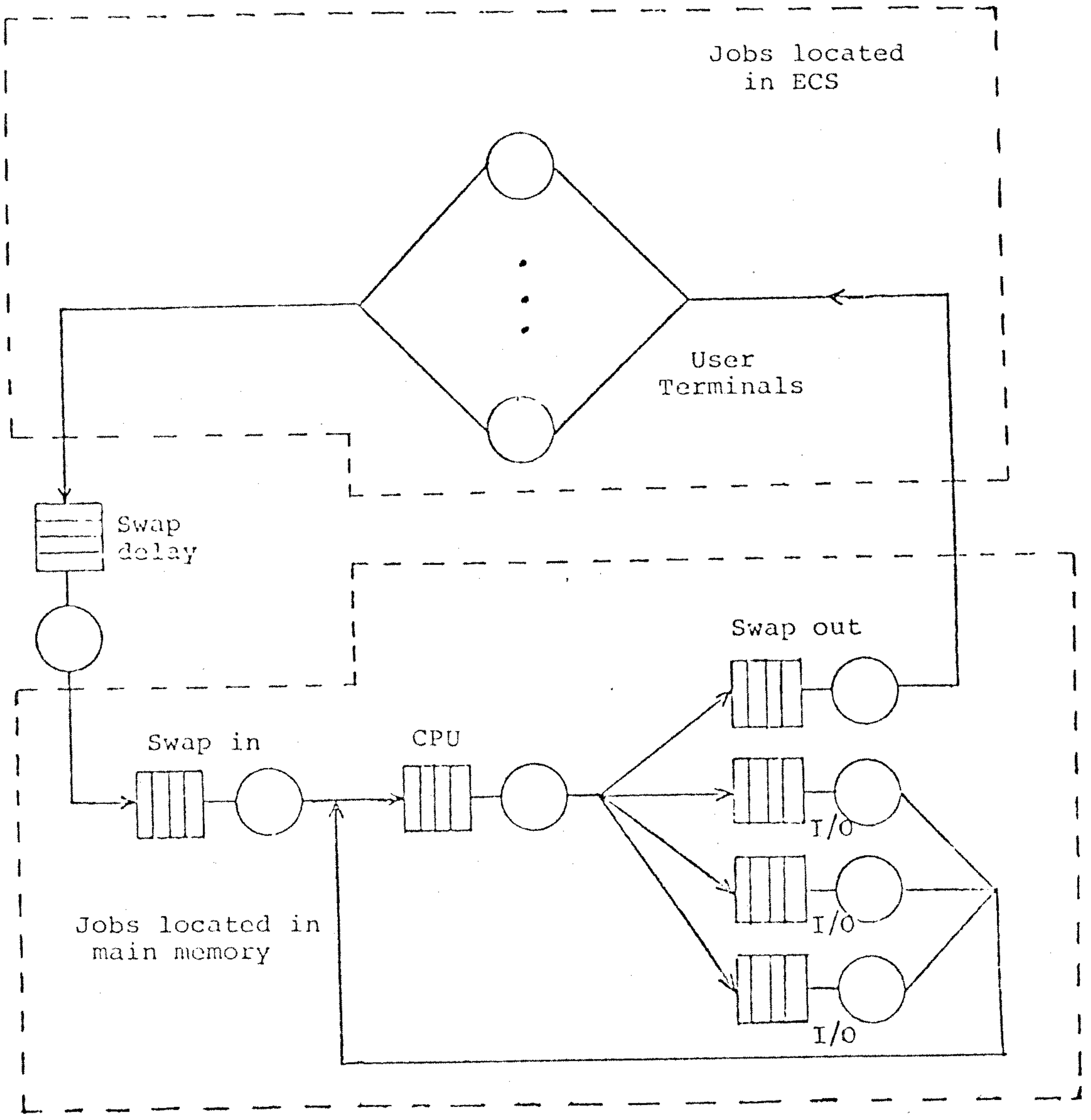


Figure 1. Overall system to be modeled.

techniques are available for: restriction of the number of jobs in the main memory portion of the network in accordance with the limitation of finite main memory and some distribution of user job main memory requirements; or determination of the fractions of main memory that are idle, used for storage of executing user jobs, or used for system overhead purposes such as swapping.

2.2 Structure of the Model

A four phase analysis technique is proposed to overcome the problems involved in the overall system evaluation. The system in Figure 1 is divided into the two systems shown in Figure 2 to use this technique. Phase 1 of the calculations is the analysis of the system shown in Figure 2a. Phase 2 is an intermediate probability analysis which controls the superposition of the two systems based on the given amount of available main memory and the given job field length (memory requirement) distribution. The third phase is the analysis of the system of Figure 2b, using the results of the previous phases to characterize the "Computer system" queue. The final phase is the construction of distributions for device utilizations, response time, throughput, the number of ready jobs in the system, and the three classes of memory utilization. A complete diagram of these phases and their interrelationships is given in Figure 3.

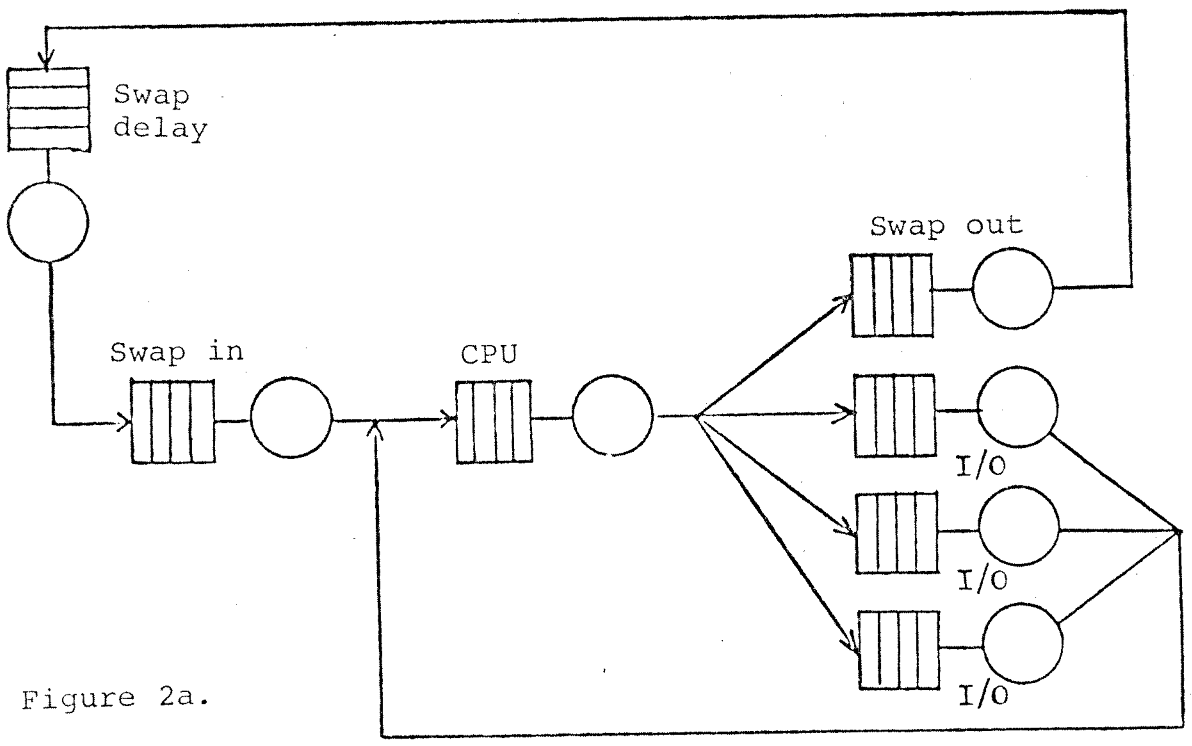


Figure 2a.

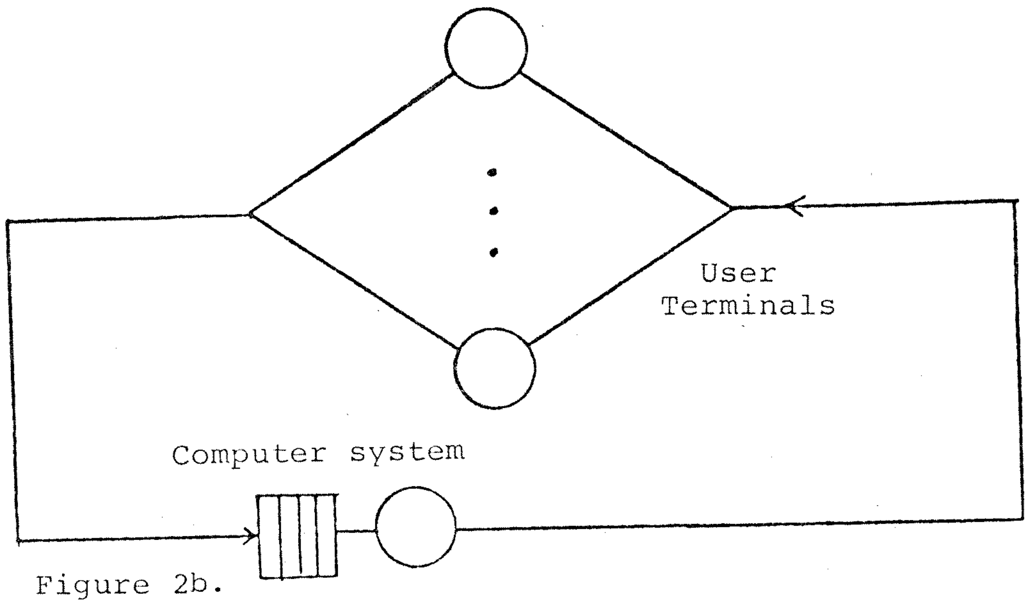


Figure 2b.

Figure 2. Component models of the overall system.

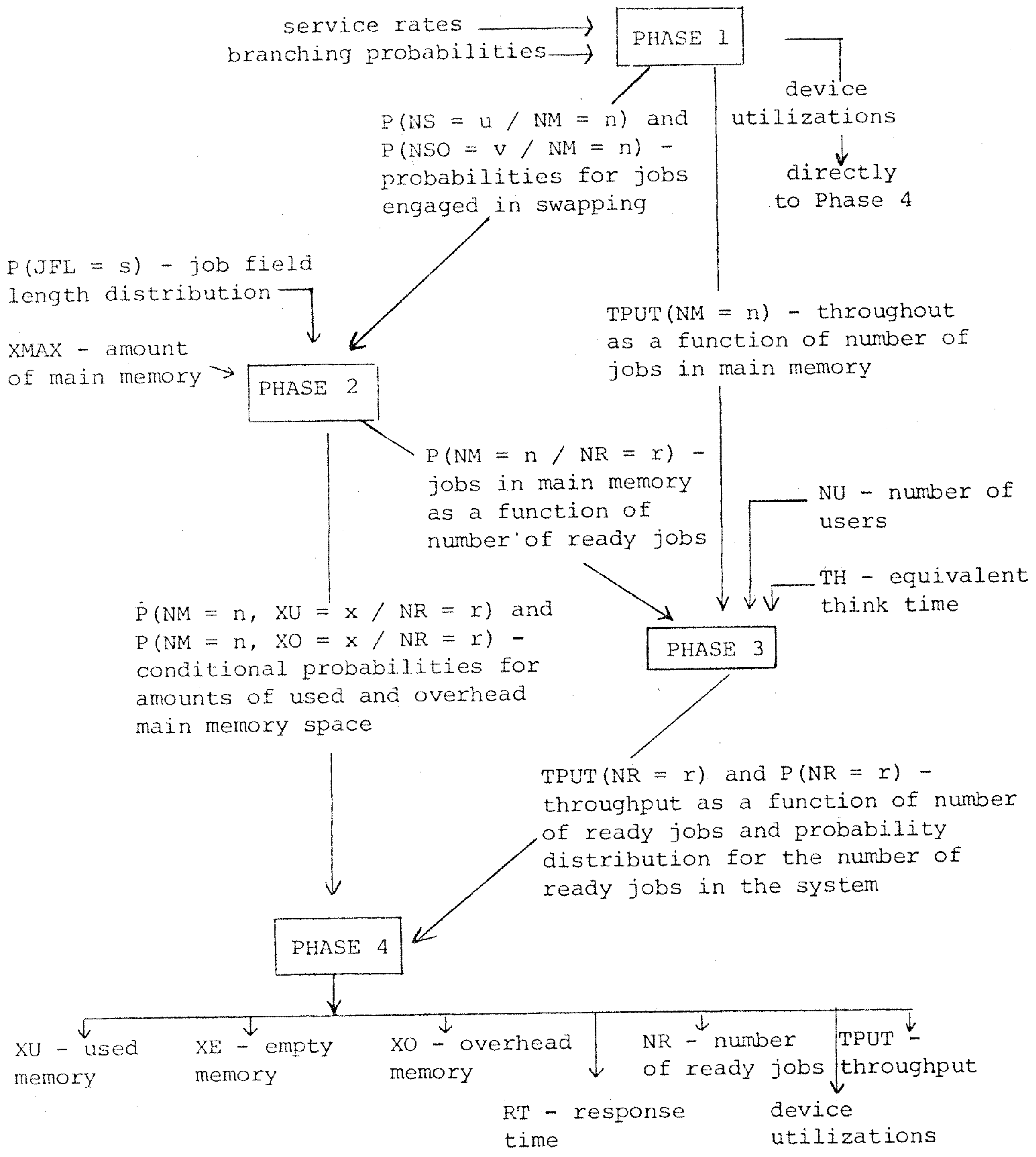

service rates
branching probabilities
PHASE 1
device utilizations directly to Phase 4
P(NS = u / NM = n) and P(NSO = v / NM = n) - probabilities for jobs engaged in swapping
P(JFL = s) - job field length distribution
XMAX - amount of main memory
PHASE 2
TPUT(NM = n) - throughout as a function of number of jobs in main memory
P(NM = n / NR = r) - jobs in main memory as a function of number of ready jobs
NU - number of users
TH - equivalent think time
P(NM = n, XU = x / NR = r) and P(NM = n, XO = x / NR = r) - conditional probabilities for amounts of used and overhead main memory space
PHASE 3
TPUT(NR = r) and P(NR = r) - throughput as a function of number of ready jobs and probability distribution for the number of ready jobs in the system
PHASE 4
XU - used memory
XE - empty memory
XO - overhead memory
RT - response time
NR - number of ready jobs
device utilizations
TPUT - throughput

Figure 3. Diagram of the interrelationships of phases of analysis.

2.2.1 Input Parameters for the Model

The following model parameters are assumed to be dependent on the system characteristics. They are assumed invarient if no changes in the system are considered (i.e. for validation runs), although it is recognized that these values will vary slightly for each individual data set taken from a real system.

(a) the amount of main memory available for user jobs

(b) the mean user think time between interactions

(c) the fraction of CPU active time consumed by system processes

(d) the mean PP service time required for an I/O operation

(e) the mean PP service time required for a swap out

(f) the mean PP service time required for a swap in

For clarification of the figures, the service times for the swap delay and swap in queues (both part of the swap in process) are defined respectively as the time from the scheduler decision to swap in a job to the time that job is allocated its required space in main memory, and the time from this allocation to the completion of the swap in process. A similar division of the swap out process is not necessary because jobs in the swap out queue release their storage at the completion of the swap out process.

The remaining model input parameters are considered dependent on the job load characteristics because they varied significantly in the empirical data sets from the real system.

(g) the number of active on-line users

(h) the mean number of I/O operations per main memory residence

(i) the probability distribution for the field length of a ready to run user job

(j) the mean CPU service time required by a user job in each burst of CPU activity

(k) the mean equivalent user think time

All of the input parameters except the last are observable from analysis of event trace data. The mean equivalent user think time is calculated by multiplying the observed probability that a swap out indicates a completed interaction by the mean user think time. User jobs in the real system may require more than one residence per interaction due to the one second (real time) quantum used for interactive job scheduling. If a job does not complete its interaction before the swap out, then that job is ready to run immediately upon completion of the swap out (zero think time). This adjustment to the think time is therefore necessary in order to compare the model results with data from the real system.

2.2.2 <u>Phase 1 Calculations</u>

These calculations are made using the network shown in Figure 2a. The necessary input parameters are the service rates for the seven queues in the system, the branching probabilities for job flow following CPU service, and the number of jobs in the system. The service times for all of the queues except the CPU are directly obtainable from the event trace data. The mean CPU burst time used in the model (TCPU) is computed from the mean user CPU burst requirement (TCPU*) and the fraction of CPU active time consumed by system processes (FSYS) by using:

$$\text{TCPU} = \text{TCPU*} \,/\, (1 - \text{FSYS}).$$

The branching probability (P(CPU - SO)) for job flow from the CPU queue to the swap out queue is determined from the observed number of I/O operations per residence (NIO) by: $P(\text{CPU} - \text{SO}) = 1 / (\text{NIO} + 1)$. The probabilities for job flow from the CPU to each of three I/O queues is assumed to be equal, that is, one third of the quantity (1 - P(CPU - SO)). The number of jobs in this network is exactly the number of jobs which occupy physically or logically (to the scheduler) portions of main memory. The network should be evaluated for levels of multiprogramming from one to the number of users in order to consider all of the possibilities. In the implementation, however, the model is evaluated for multiprogramming levels from one to ten because the probability of more than ten jobs fitting into the available main memory of this system is negligible. All large interactive systems should have similar upper bounds that are significantly less than the typical number of users. For levels of multiprogramming above ten, the model output is approximated by the model output with ten jobs in the system.

The steady state probabilities for the possible states of this model are calculated using techniques of local balance (7). The CPU is assumed to be processor shared as an approximation to the 16 millisecond, round robin scheduling technique of the real system. Swapping and I/O service is assumed to be on a FCFS basis because the channels in the real system are allocated on FCFS criterion. The service times for the devices in the system are assumed to be independent, exponential random variables. Analysis of this model produces, as functions of the level of multiprogramming, the system throughput (job flow through the swap out queue), the device utilizations, and probability distributions for the number of jobs engaged in swapping. Memory residence time, the total service time that a job requires per residence, is also obtainable from this model.

In the model there are six queues that require allocation of a peripheral processor (swap delay, swap in, swap out, and three I/O queues). In the real system there are usually five pool peripheral processors available for users. It is however, not necessary to include the effects of PP request interference because of the relatively low utilizations of these queues. This approximation is also supported by data taken from the real system which indicates no significant wait times for PP allocation.

2.2.3 Phase 2 Calculations

The calculations for this phase of the analysis use the ordered quadruple (NM, XU, XO, NR) to represent the states of the system where:

NM is the number of user jobs that occupy main memory;

XU is the amount of main memory that is used for executing user jobs;

XO is the amount of main memory that is used for swapping user jobs;

NR is the number of user jobs that are ready to run.

A typical state diagram, including the possible transition paths, is shown in Figure 4. The necessary input parameters are the amount of main memory available for user jobs, the job field length probability distribution for ready jobs, the probability distributions for the number of jobs engaged in swapping as a function of the number of jobs that occupy main memory (from Phase 1), and the system memory scheduling algorithm. This data will be used to construct the conditional probability distributions for the amount of main memory that is idle, used for executing user jobs, and used for swapping user jobs (overhead) as functions of the number of ready jobs in the system. The conditional probability distribution for the number of jobs occupying main memory space will also be constructed as a function of the number of ready jobs.

Let $P(NM = n, XC = x / NR = r)$ be the conditional probability that n jobs are in main memory and x units of main memory are committed to those jobs (memory space is committed to jobs that are either executing or engaged in swapping) given that r jobs are ready to run.

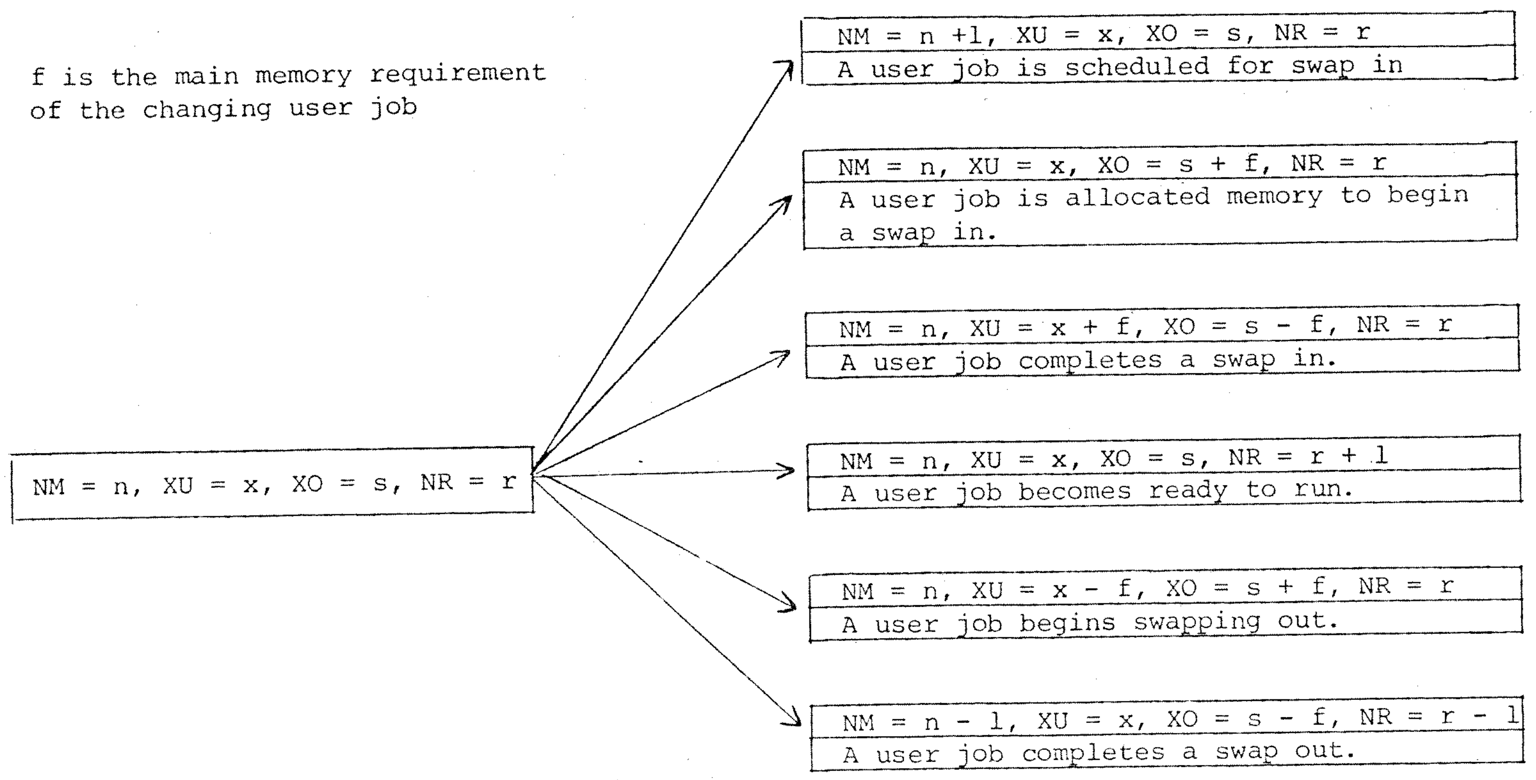

f is the main memory requirement
of the changing user job
NM = n, XU = x, XO = s, NR = r
NM = n +1, XU = x, XO = s, NR = r
A user job is scheduled for swap in
NM = n, XU = x, XO = s + f, NR = r
A user job is allocated memory to begin a swap in.
NM = n, XU = x + f, XO = s - f, NR = r
A user job completes a swap in.
NM = n, XU = x, XO = s, NR = r + 1
A user job becomes ready to run.
NM = n, XU = x - f, XO = s + f, NR = r
A user job begins swapping out.
NM = n - 1, XU = x, XO = s - f, NR = r - 1
A user job completes a swap out.

Figure 4. Typical state for the main memory system.

Let P(JFL = s) be the probability that a ready job has a field length of exactly s units of memory and P(JFL $\geq$ s) be the probability that a ready job has a field length greater than or equal to s units of memory. These two distributions are easily derived from the given job field length probability distribution. The probability distribution for the ready job field lengths was empirically constructed by dividing the number of observed jobs that became ready to run and had a field length of s units of memory by the number of observed jobs that became ready to run. It is assumed that the field length distribution is independent of the state of the memory system because the number of jobs occupying main memory is usually small compared to the number of simultaneous system users.

The current job scheduling algorithm for the CDC 6400 interactive system is a frist fit scheduling discipline with the queue order determined by a FIFO policy. Figure 5 illustrates the concepts of the following calculations.

Clearly, P(NM = 1, XC = x / NR = 1) = P(JFL = x) for x = 1,2,...,XMAX where XMAX is the total main memory available for users. For NR greater than NM, the ready jobs that are not in main memory must be too large to fit into the empty main memory. Thus, P(NM = 1, XC = x / NR = r) = P(JFL = x) x P(JFL $\geq$ XMAX - x + 1)$^{r-1}$ for x = 1,2,..., XMAX and r = 1,2,...,NU where NU is the

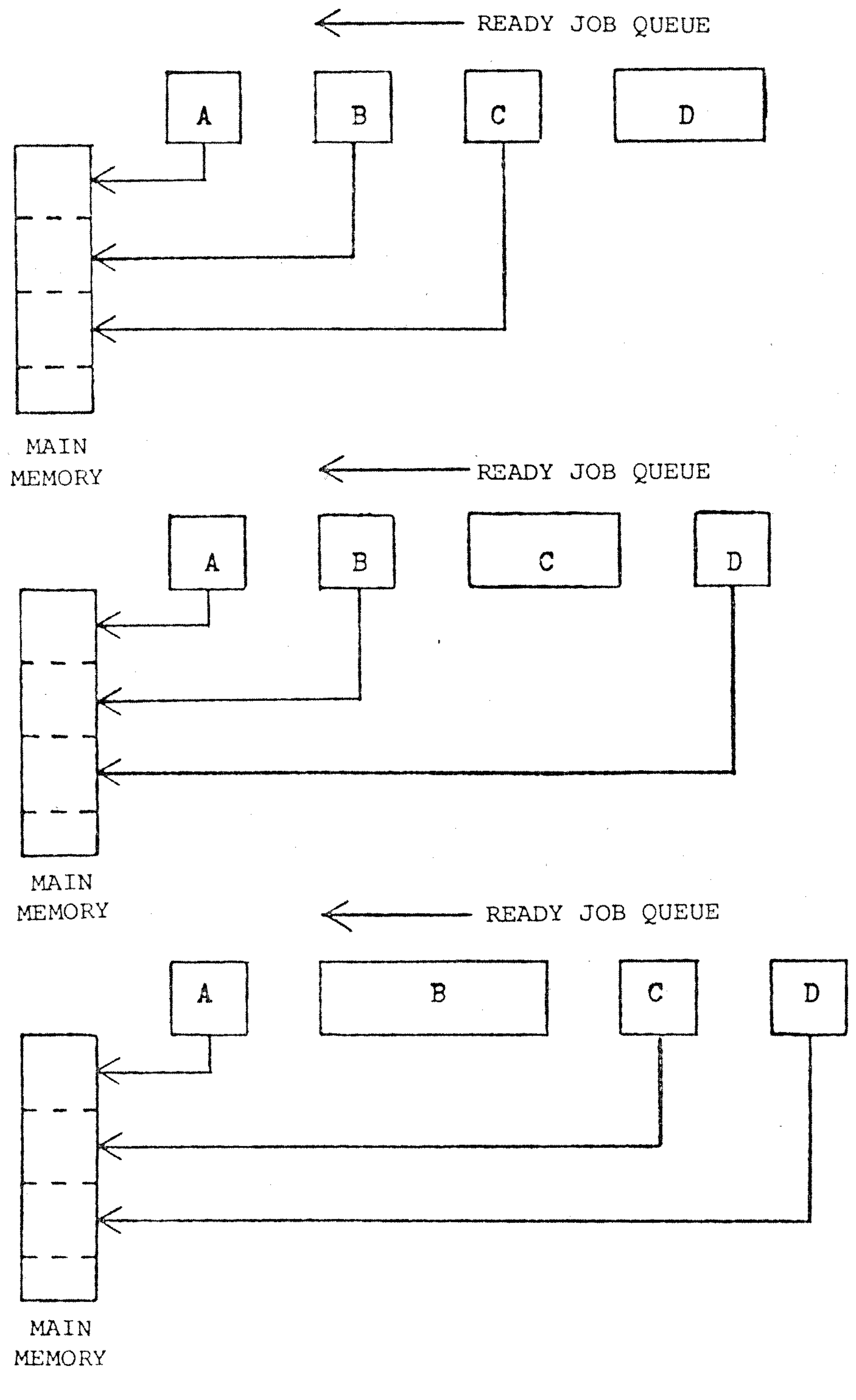


Figure 5. Possible conditions for three jobs in main memory and x units of memory being occupied given that four jobs are ready to run.

number of users. In the model implementation, a practical upper bound (NRMAX) is placed on r. Phase 3 calculations produce an estimate of the lost probabilities resulting from the use of NRMAX instead of NU as an upper bound. NRMAX is always chosen large enough to insure that the total lost probabilities are less than one tenth of one percent.

The remaining probabilities can be calculated recursively by observing that for two jobs in main memory:

$$P(NM = 2,\ XC = x \ /\ NR = 1) = 0 \text{ for all } x.$$

$$P(NM = 2,\ XC = x \ /\ NR = 2) = \sum_{s=1}^{x-1} P(JFL = x - s) \times P(JFL = s)$$

$$= \sum_{s=1}^{x-1} P(NM = 1,\ XC = x - s \ /\ NR = 1) \times P(JFL = s)$$

Similarly,

$$P(NM = 2,\ XC = x \ /\ NR = r)$$

$$= \sum_{s=1}^{x-1} P(NM = 1,\ XC = x - s \ /\ NR = r - 1) \times P(JFL = s) + P(NM = 2,\ XC = x \ /\ NR = r - 1) \times P(JFL \geq XMAX - x + 1)$$

for x = 1,2,..., XMAX and r = 3,4,..., NU.

In general, n jobs occupy x units of main memory given r ready jobs if either n - 1 jobs occupy x - s units given r - 1 ready jobs and the rth job is of size s units or n jobs occupy x units of memory given r - 1 ready jobs and the rth job is too large to fit into the remaining empty main memory. Thus,

$$P(NM = n, XC = x / NR = r)$$
$$= \sum_{s=1}^{x-n+1} P(NM = n-1, XC = x - s / NR = r - 1)$$
$$\times P(JFL = s) + P(NM = n, XC = x / NR = r - 1)$$
$$\times P(JFL \geq XMAX - x + 1)$$

for $n = 2,3,\ldots,NU$, $x = n, n+1, \ldots, XMAX$, and $r = n, n+1, \ldots, NU$ where $P(NM = n, XC = x / NR = r) = 0$ for all r less than n. These probabilities are the conditional probabilities for n jobs occupying x units of main memory given r ready jobs. Part or all of this occupied memory space may contain one or more user jobs that are executing (used memory space). Some of this occupied memory may also be committed (as seen by the scheduler) to jobs in the swap delay queue. This memory is considered to be empty memory space. The probability distributions for the number of jobs engaged in swapping, computed in Phase 1, are needed in order to determine the proper division of this occupied memory space into used, overhead, and empty memory.

Let:

P(NS = u / NM = n) be the conditional probability that u jobs are in the swap delay, swap in, or swap out queues given that n jobs are in main memory. The memory that these u jobs occupy is either overhead memory space or empty memory space.

P(NS0 = v / NM = n) be the conditional probability that v jobs are in the swap in or swap out queues given that n jobs are in main memory. The memory that these v jobs occupy is considered overhead memory space.

P(JS = s / NJ = j) be the conditional probability that a total of s units of main memory are required by the given j user jobs.

P(JS = s / NJ = j, NM = n, XC = x) be the conditional probability that s units of main memory are occupied by the given j jobs in main memory, given that there are a total of n jobs in main memory and that those n jobs occupy x units of main memory.

P(NM = n, XU = x / NR = r) be the conditional probability that n jobs occupy portions of main memory and that x units of main memory are used for executing user jobs given that r jobs are ready.

P(NM = n, X0 = x / NR = r) be the conditional probability that n jobs occupy portions of main memory and that x units of main memory are used for swapping user jobs (overhead) given that r jobs are ready.

The probabilities, P(JS = s / NJ = j), can be easily determined from the given job field length distribution by assuming independence of the job field lengths and computing recursively:

$$P(JS = s / NJ = 1) = P(JFL = s)$$

$$P(JS = s / NJ = j) = \sum_{t=1}^{s-j+1} P(JS = s - t / NJ = j - 1) \times P(JFL = t).$$

These probabilities can then be used to compute:

$$P(JS = s / NM = n, XC = x, NJ = j)$$

$$= \frac{P(JS = s, NM = n, XC = x, NJ = j)}{P(NM = n, XC = x, NJ = j)}$$

$$= \frac{(PJS = s, XC = x / NM = n, NJ = j)}{P(XC = x / NM = n, NJ = j)}$$

$$= \frac{P(JS = s / NJ = j) \times P(JS = x - s / NJ = n - j)}{P(XC = x / NM = n)}$$

Note that P(XC = x / NM = n, NJ = j) = P(XC = x / NM = n) = P(JS = x / NJ = n) since the amount of memory required by the n jobs is independent of the subset of those n jobs that are engaged in swapping. However, in the model implementation, the following approximation was used:

$$P(JS = s \,/\, NM = n,\ XC = x,\ NJ = j)$$

$$= \frac{P(JS = s \,/\, NJ = k)}{\sum_{k=1}^{x-n+j} P(JS = k \,/\, NJ =)} \quad \text{for } j < n$$

$$= 1 \quad \text{for } j = n$$

$$= 0 \quad \text{for } j > n$$

This approximation assumes that the probability distribution for the j jobs engaged in swapping is the probability distribution for the memory requirements of j jobs given that those j jobs must occupy at most x - n + j units of memory.

The probabilities related to the used memory space can now be calculated, noting that $P(NS = u \,/\, NM = n, NR = r) = P(NS = u \,/\, NM = n)$ because the number of jobs engaged in swapping is independent of the number of ready jobs if the number of jobs in main memory is given. For NM = 1 the amount of used space is either equal to the amount of occupied memory or it is equal to zero. Thus,

$$P(NM = 1,\ XU - x \,/\, NR = r) = P(NM = 1,\ XC = x \,/\, NR = r) \times P(NS = 0 \,/\, NM = 1).$$

For NM > 1 a second term must be included which subtracts the memory space committed to swapping jobs from the total committed memory space.

$$\begin{aligned} & P(NM = n,\ XU = x\ /\ NR = r) \\ & \quad = P(NM = n,\ XC = x\ /\ NR = r) \times P(NS = 0\ /\ NM = n) \\ & \qquad + \sum_{s = x + 1}^{XMAX} P(NM = n,\ XC = s\ /\ NR = r) \\ & \qquad \times \sum_{j = 1}^{n} P(NS = j\ /\ NM = n) \\ & \qquad\quad \times P(JS = s - x\ /\ NM = n,\ XC - x,\ NJ = j) \end{aligned}$$

The probabilities related to the overhead memory space are calculated by accumulating the probabilites that were subtracted from the occupied memory in the calculations above. The difference is that $P(NSO = v\ /\ NM = n)$ is used in place of $P(NS = u\ /\ NM = n)$ because the memory space occupied by jobs in the swap delay queue is considered empty memory, not overhead memory. Therefore, the general expression is similar to the second term in the calculation of the used memory probabilities, noting again that $P(NSO = v\ /\ NM = n,\ NR = r) = P(NSO = v\ /\ NM = n)$.

$$\begin{aligned} & P(NM = n,\ XO = x\ /\ NR = r) \\ & \quad = \sum_{s = n}^{XMAX} P(NM = n,\ XC = s\ /\ NR = r) \\ & \qquad \times \sum_{j = 1}^{n} P(NSO = j\ /\ NM = n) \\ & \qquad \times P(JS = s - x\ /\ NM = n,\ XC = s,\ NJ = j) \end{aligned}$$

λ represents 1/TH, the reciprocal of the equivalent user think time

N represents NU, the number of users

TPUT (r) represents TPUT(NR = r), the system throughout given that r jobs are ready

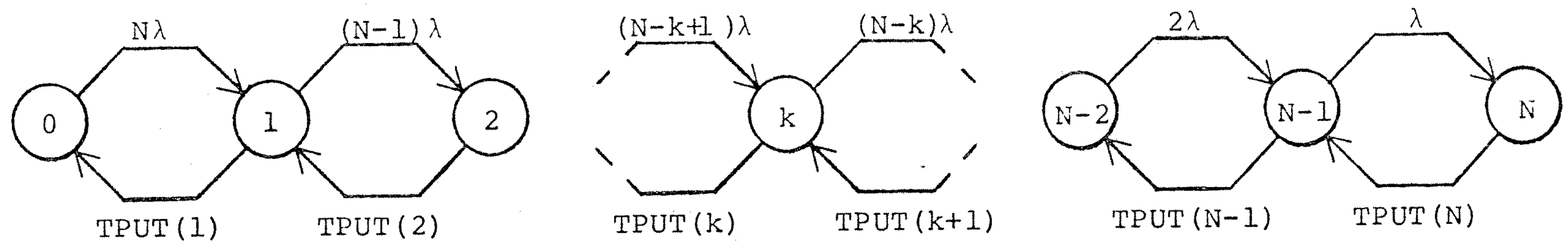


Figure 6. State transition graph for Phase 3 analysis.

The conditional probability distribution for the number of user jobs in main memory (level of multiprogramming) as a function of the number of ready jobs can also be calculated from P(NM = n, XC = x / NR = r). Let P(NM = n / NR = r) be the conditional probability that n jobs occupy portions of main memory given that r jobs are ready. Then

$$P(NM = n / NR = r) = \sum_{x=1}^{XMAX} P(NM = n, XC = x / NR = r)$$

for n = 1,2,...,NU and r = n,n+1,...,NU.

These calculations are complex and necessarily the most time consuming portion of this modeling technqiue. However, the distributions that are obtained are those necessary to control the superposition of the two systems analysed in Phases 1 and 3.

2.2.4 Phase 3 Calculations

A network representation of the system considered in this phase of the analysis is shown in Figure 2b and a state transition diagram for the system is shown in Figure 6. The necessary input parameters are the number of users, the mean equivalent user think time, and the throughput of the system as a function of the number of ready jobs in the system. The last parameter can be determined from the results of the preceding two phases of analysis. Let TPUT(NM = n) be the system throughput as a function of the level of multiprogramming (computed in Phase 1). Let TPUT(NR = r) be the throughput of the system when r jobs are ready. Then $\mathrm{TPUT}(NR = r) = \sum_{n=1}^{r} P(NM = n \,/\, NR = r)$ $\times\ \mathrm{TPUT}(NM = n)$ for $r = 1,2,\ldots,NU$.

The computer system is assumed to be processor shared and the service times are assumed to be independent and their density functions differentiable (14). The states of this system, Figure 6, are identified by the corresponding number of ready to run jobs in the computer system. Let P(NR = r) be the probability that r jobs are ready to run in the computer system, TH be the mean equivalent user think time, and NU be the number of users. The state probabilities can be calculated by observing from the state diagram that:

$$P(NR = r) = P(NR = r - 1) \times (NU - r + 1) \\ /(TH \times \mathrm{TPUT}(NR = r))$$

for r = 1,2,...,NU. This relationship gives NU equations with NU + 1 unknown probabilities. The remaining condition that is necessary for the solution to be unique is that:

$$\sum_{r=0}^{NU} P(NR = r) = 1 \text{ by the definition of the probabilities.}$$

These equations are solved in the model by setting P(NR = 0) to one, computing the P(NR = r) for r = 1,2,..., NU, and then dividing each state probability by the sum of state probabilities in order to satisfy the last condition. These state probabilities are equivalent to the probability distribution for the number of ready jobs in the system. Also, it is possible to estimate the error resulting from the use of the practical upper bound NRMAX in the Phase 2 calculations by computing P(NR > NRMAX).

2.2.5 Phase 4 Calculations

This phase of the analysis consists only of the construction of desired information from the combined results of the preceding phases of analysis. It is clear that many different types of information are available and that the specific information calculated depends on the objectives of the modeling. The following are examples of several of the possible calculations:

(1) Let P(XU) = x / NR = r) be the probability that x units of main memory are used for executing user jobs given that r jobs are ready and P(XU = x) be the probability that x units of main memory are used for executing user jobs. Then, using the results of Phase 2:

$$P(XU = x / NR = r) = \sum_{n=1}^{r} P(NM = n, XU = x / NR = r)$$

for all possible x and r.

$$P(XU = x) = \sum_{r=1}^{NU} P(XU = x / NR = r)$$

$\times$ P(NR = r) for all possible x.

(2) The probability distribution for the memory space used for swapping user jobs (overhead) can be constructed in a manner similar to the calculations for the used memory space distribution. The equations are the same as those in (1) with XO replacing XU.

(3) The probability distribution for the empty memory space can be constructed from (1) and (2). Let P(XO = y / XU = x) be the conditional probability that y units of main memory are used for overhead purposes given that x units of main memory are used for executing user jobs. An approximate calculation is:

$$P(XO = y \,/\, XU = x) = P(XO = y) \,/ \sum_{s=1}^{XMAX - x} P(XO = s)$$

Let P(XE = x) be the probability that x units of main memory are empty. Then, knowing that XU + XO + XE = XMAX,

$$P(XE = x) = \sum_{s=0}^{XMAX - x} P(XU = s)$$

$$x \; P(XO = XMAX - x - s \,/\, XU = s)$$

for all possible values of x.

(4) The response time of the system as a function of the number of ready jobs in the system, RT(NR = r), is calculated by: RT(NR = r) = r / TPUT(NR = r) for all possible values of r. The mean response time is simply

$$\overline{RT} = \sum_{r=1}^{NU} RT(NR = r) \; x \; P(NR = r).$$

The response time given by these equation is the time interval from the user input signal that the job is ready to run to the completion of the swap out of that job. It is, therefore, the response time for a job that requires one residence to complete its interaction.

(5) The mean values and standard deviations of the distributions resulting from all of the phases of analysis are obtained by standard techniques.

2.3 Summary of Model Capabilities

Using a four phase probabilitistic analysis, this model produces probability distributions for used memory space, overhead memory space, empty memory space, the number of ready jobs in the system, the system throughput, the device utilizations, and the response time of the interactive system. Additionally, the model is constructed so that all of the input parameters may be varied within wide ranges although some have been considered constant system characteristics for the simplicity of the model. This model provides a flexible, relatively simple, method of system evaluation for a large interactive system including the restriction of finite main memory.

SECTION III

OBTAINING DATA FROM THE REAL SYSTEM

3.1 Data Generation

The data presented in this section was obtained from the CDC 6400 interactive system described in Section I. This was accomplished through the use of a micro-level event trace package embedded in the UT-2D operating system. Details of the event trace package can be found in (12,17). This probe records on magnetic tape a chronological sequence of requests for the allocation, preemption, and deallocation of system resources by processes. The basic design of the event trace is to record only the lowest level of these requests and the corresponding responses of the operating system.

3.2 Data Reduction

The event trace package operates in a multiprocessing environment and therefore produces an event string resulting from the interleaving of several simultaneous process event sets. In order to obatin detailed information about individual job requirements, it is necessary to sort the events into sets corresponding to their associated jobs or processes. Approximately sixty percent of the recorded events are relevant to this analysis. Some of the retained events are:

(a) Scheduling decisions to swap in or out a job.

(b) Loading of a peripheral processor with a swap program to comply with the scheduled decisions.

(c) A job's request for and the system allocation of memory space, peripheral processors and their functions (including I/O), and CPU service.

(d) Deallocation or preemption of resources by either a job or the system.

(e) System monitor decisions to relocate the position of a job in main memory.

(f) The response from a terminal which places a job on ready to run status.

(g) The response of the system to the user which places the job on a wait for terminal response status.

Tracking these types of events and their times of occurrence made it possible to obtain observed statistical data to serve as both input and validation values for the model.

3.3 Data Analysis

It was possible, after these events were sorted by process and correlated, to construct from them a statistical description of user job characteristics and also statistical data for system performance measures. Table 1 shows the reduced statistical data for a typical event trace tape. This is a summary of the observed distributions represented by their mean values and standard deviations. Figures 7 and 8 show the probability distributions for the amount of empty main memory space, the amount of overhead main memory, and the number of ready jobs in the system. Figures 9, 10, and 11 show the probability distributions for job resource requirements such as main memory, I/o activity, and CPU activity.

TABLE 1

REDUCED FORM OF OBSERVED STATISTICAL DATA

DATA	MEAN	σ
Job field length	8.4K	6.8K
Unused main memory	8.7K	7.8K
Total overhead memory	3.8K	7.6K
Overhead: swaps using ECS	1.7K	4.9K
Overhead: swaps using disk (infrequent)	0.3K	2.4K
Overhead: relocation	1.8K	6.0K
Swap rate	8.7/sec.	1.4/sec.
Queue length for memory	7.2 jobs	3.5 jobs
Scheduler run interval	126 msec.	19 msec.
Job wait for scheduler swap in decision	720 msec.	724 msec.
Job wait for PP during swap in	13 msec.	22 msec.
Job wait for transfer to main memory	80 msec.	127 msec.
Multiprogramming	3.676 jobs	1.292 jobs

Total sample time: 287.412 seconds

Memory available for users: 33K (K = 1024_{10})

Users on line: 52

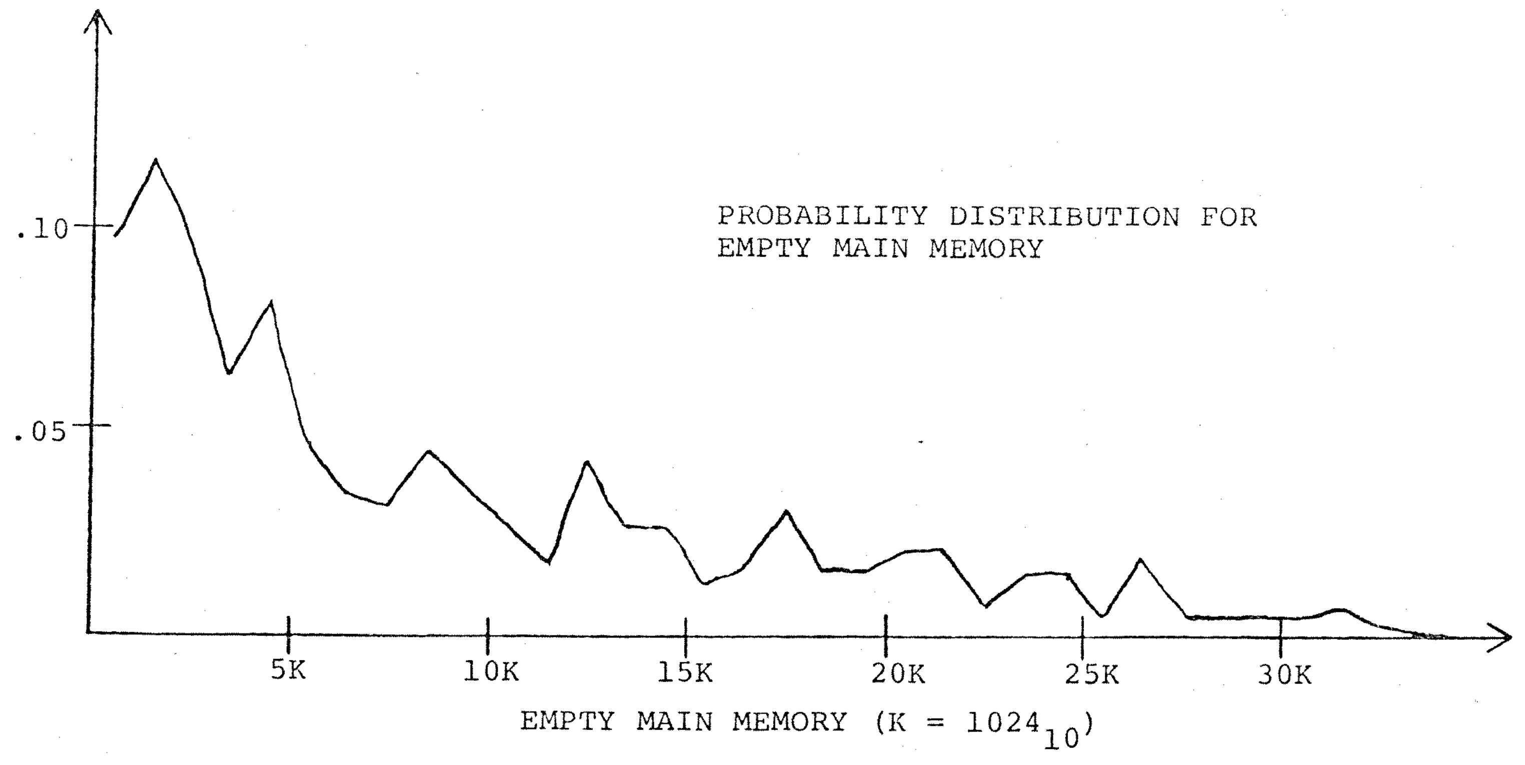


Figure 7. Probability distributions for empty main memory space.

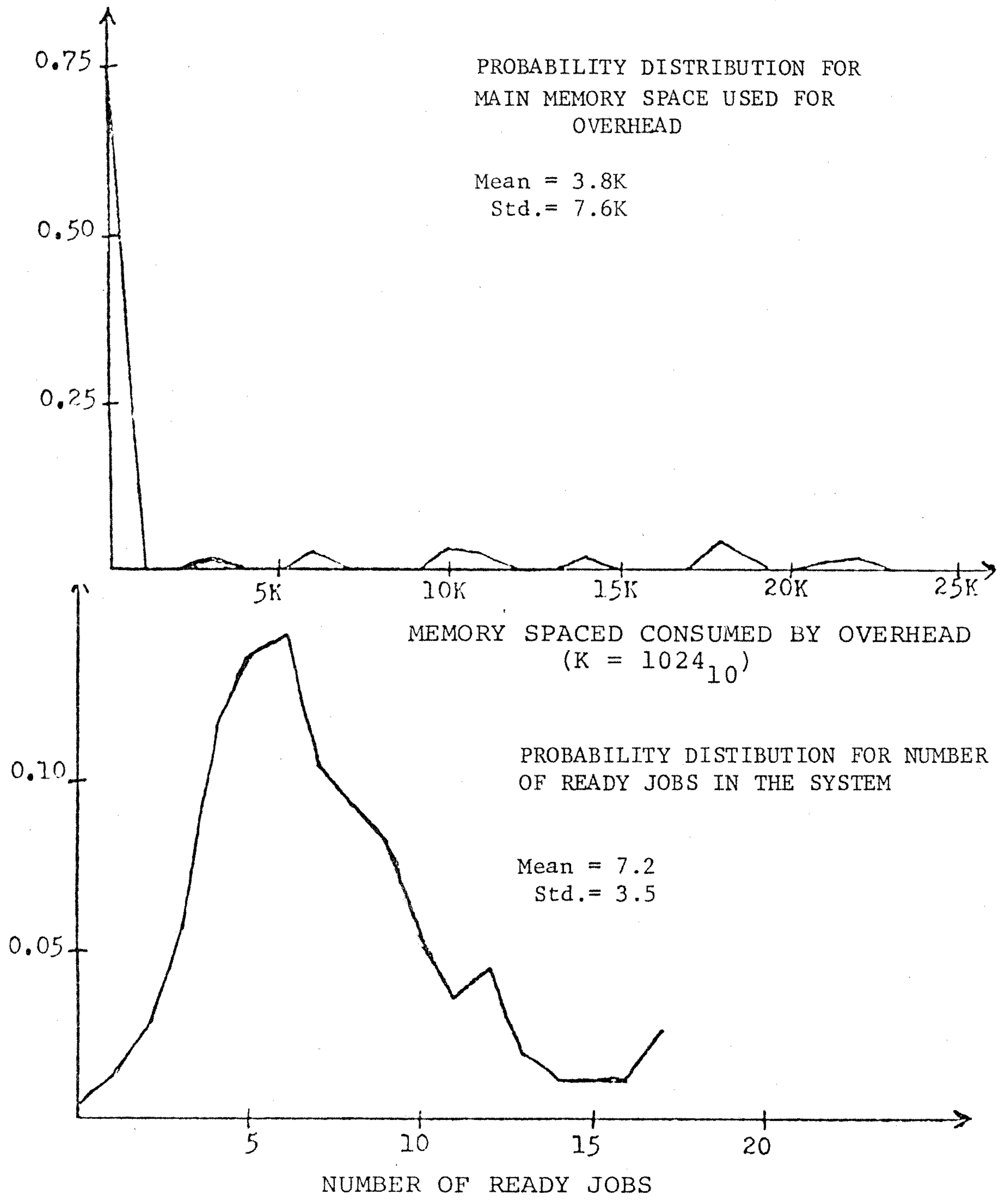


Figure 8. Probability distributions for the amount of overhead memory space and the number of ready jobs in the system.

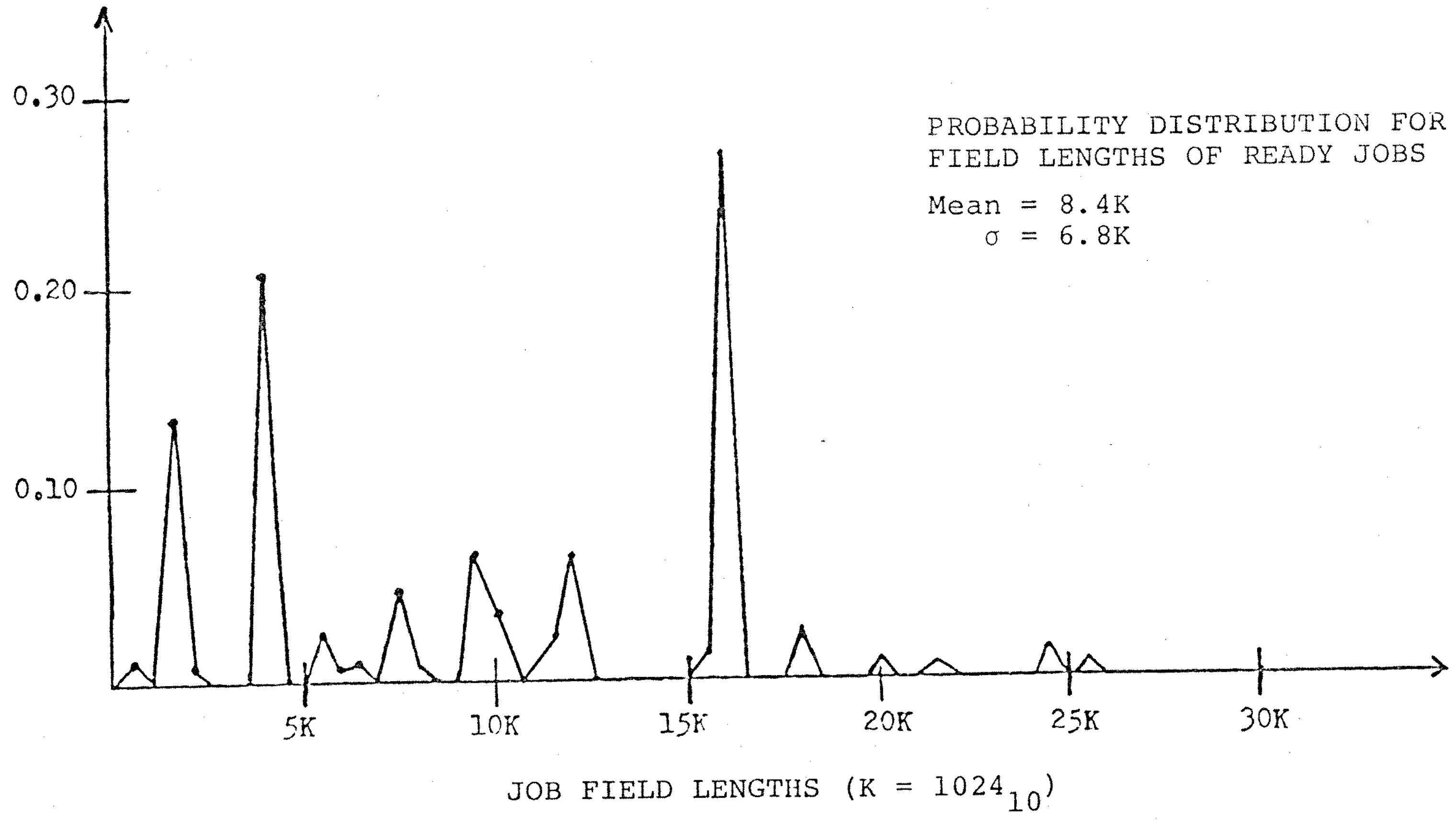
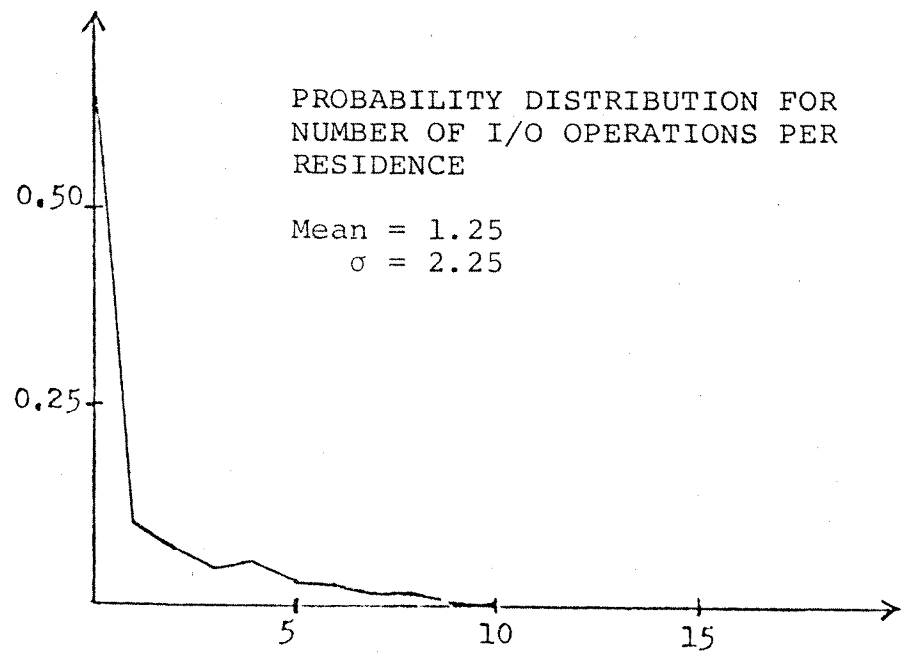


Figure 9. Probability distribution for field lengths of ready jobs.



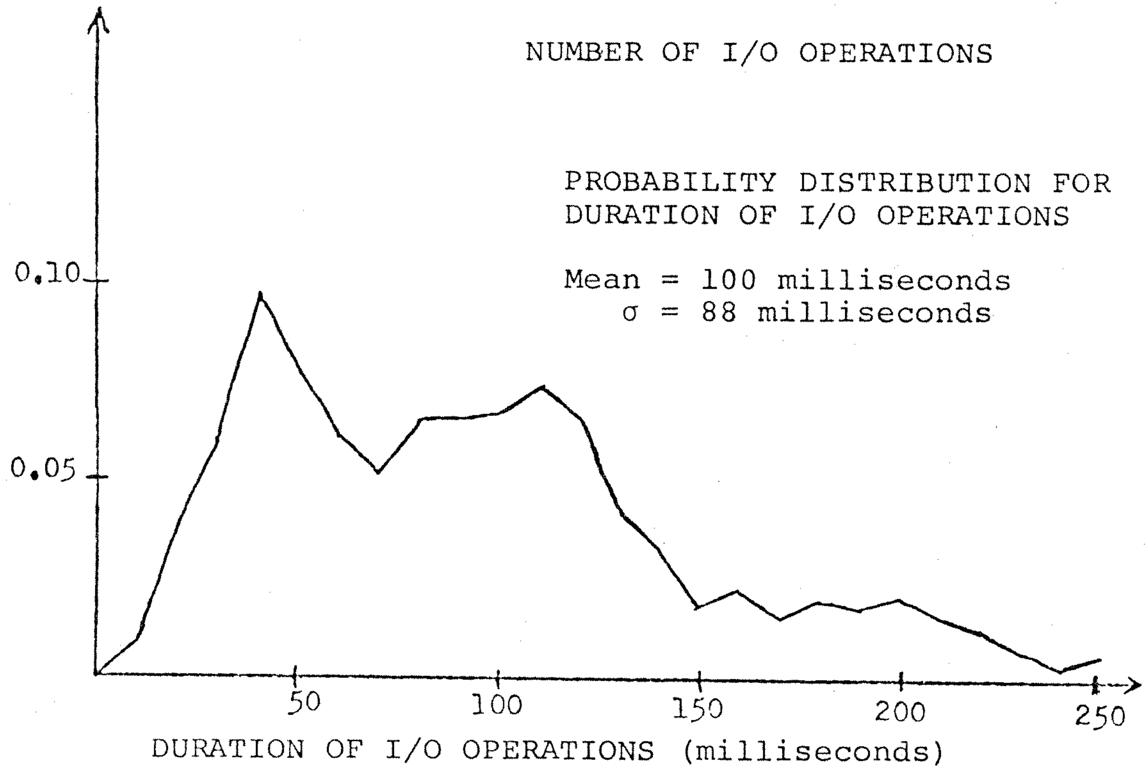
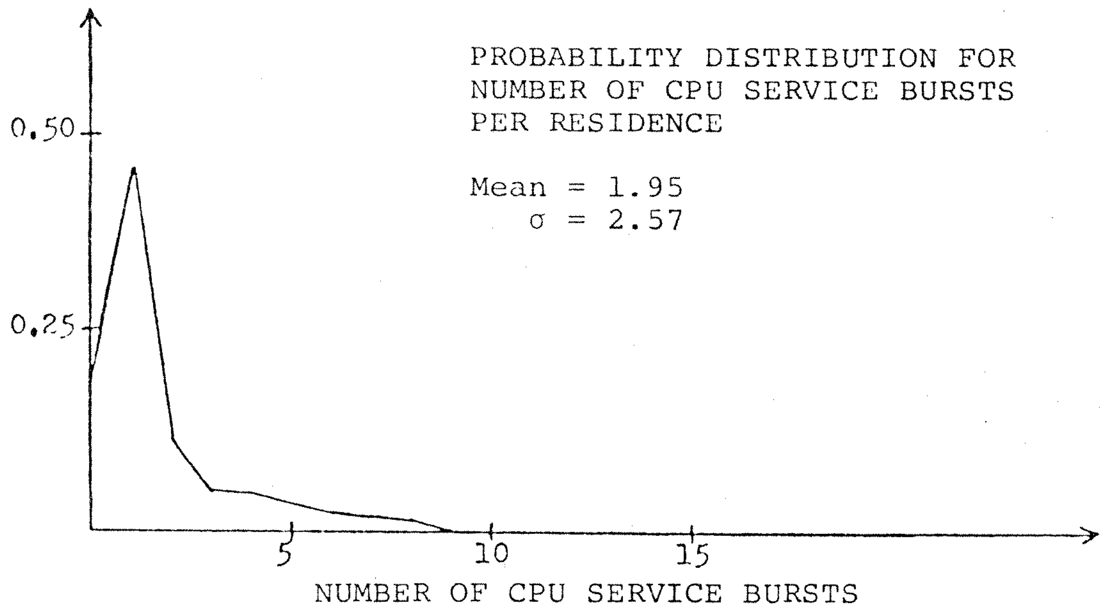


Figure 10. Probability distributions for job I/O characteristics.



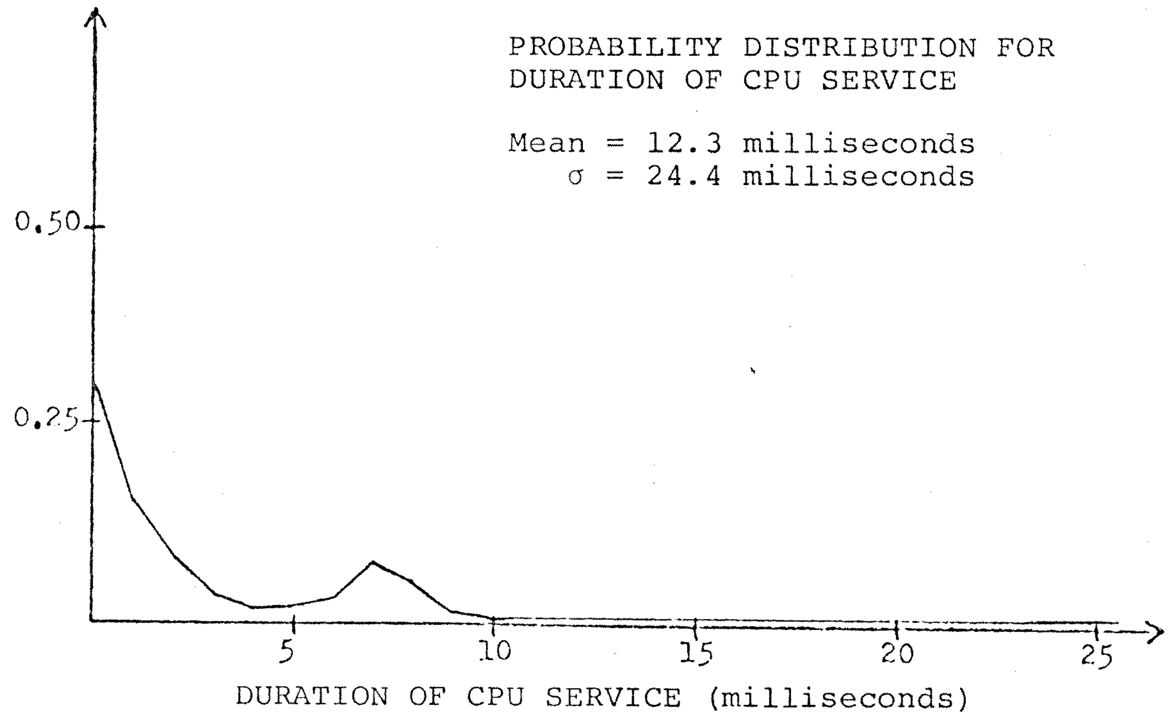


Figure 11. Probability distributions for job CPU requirements.

SECTION IV

MODEL PARAMETERIZATION AND VALIDATION

4.1 Parameterization of the Model

The input parameters used for the validation of the model were obtained from the analysis of three event trace tapes. Those input parameters which were assumed to be constant for the validation were estimated from the first two event trace data sets using no knowledge from the third. The following values were used:

(a) The amount of main memory available for users was estimated to be 66 units. One model unit of memory was defined to be 512 words (1000_8). The observed available memory space for user jobs varied from approximately 64 units to 68 units depending on the dynamic sizes of system buffers.

(b) The mean user think time was estimated to be 18.7 seconds. This estimate was made using the observed values for system throughput and mean number of residences per interaction.

(c) The fraction of CPU active time consumed by the system was assumed to be 0.835. In this system user jobs compete with system jobs and the central system monitor for CPU service. Because of the high priority of the system CUP requirements, user jobs receive only 15 to 18 percent of the total CPU service, typically in one or two millisecond bursts.

(d) The mean time required for an I/O operation was estimated as 100 milliseconds and the distributions for I/O service were assumed to be exponential. The actual observed mean time required for an I/O operation varied from 96 to 119 milliseconds.

(e) The mean time required for a swap out was 30 milliseconds. The observed times required for swaps out varied from 20 to 40 milliseconds.

(f) The mean time required for a swap in was estimated as 80 milliseconds. This time is divided into two parts: 60 milliseconds before the allocation of memory space by the system, and 20 milliseconds from allocation to completion of the swap in. This mean time varied from 79 to 80 milliseconds on the two tapes.

The variable input parameters were calculated as described in Section II. They are shown for each of the three tapes in the top portion of Table 2. The job field length probability distributions for each of the tapes are shown in Figures 12, 13, and 14. Note that, although there are some differences in these distributions, a

TABLE 2

VALIDATION INFORMATION FOR THE MEANS OF THE OBSERVED AND CALCULATED DISTRIBUTIONS

DATA	TAPE 1	MODEL#1	TAPE 2	MODEL#2	TAPE 3	MODEL#3	
Number of users	52	52	30	30	49	49	INPUT
Equivalent think time (seconds)	9.44	9.44	5.71	5.71	9.91	9.91	
CPU activity burst (milliseconds)	12.25	12.25	13.18	13.18	11.22	11.22	
Number of I/O operations per residence	1.25	1.25	1.65	1.65	1.29	1.29	
Response time (seconds)	1.323	1.273	1.241	1.600	0.970	0.740	OUTPUT
Throughput (jobs/second)	4.344	4.812	4.871	4.063	3.281	4.551	
Multiprogramming level	3.676	3.470	3.304	3.988	2.345	3.125	
CPU utilization	.9284	.8020	.9257	.8621	.8330	.7082	
Overhead memory (in model units)	4.148	4.860	3.479	3.726	2.834	3.044	
Empty memory (in model units)	17.336	22.917	19.871	19.989	24.662	33.681	

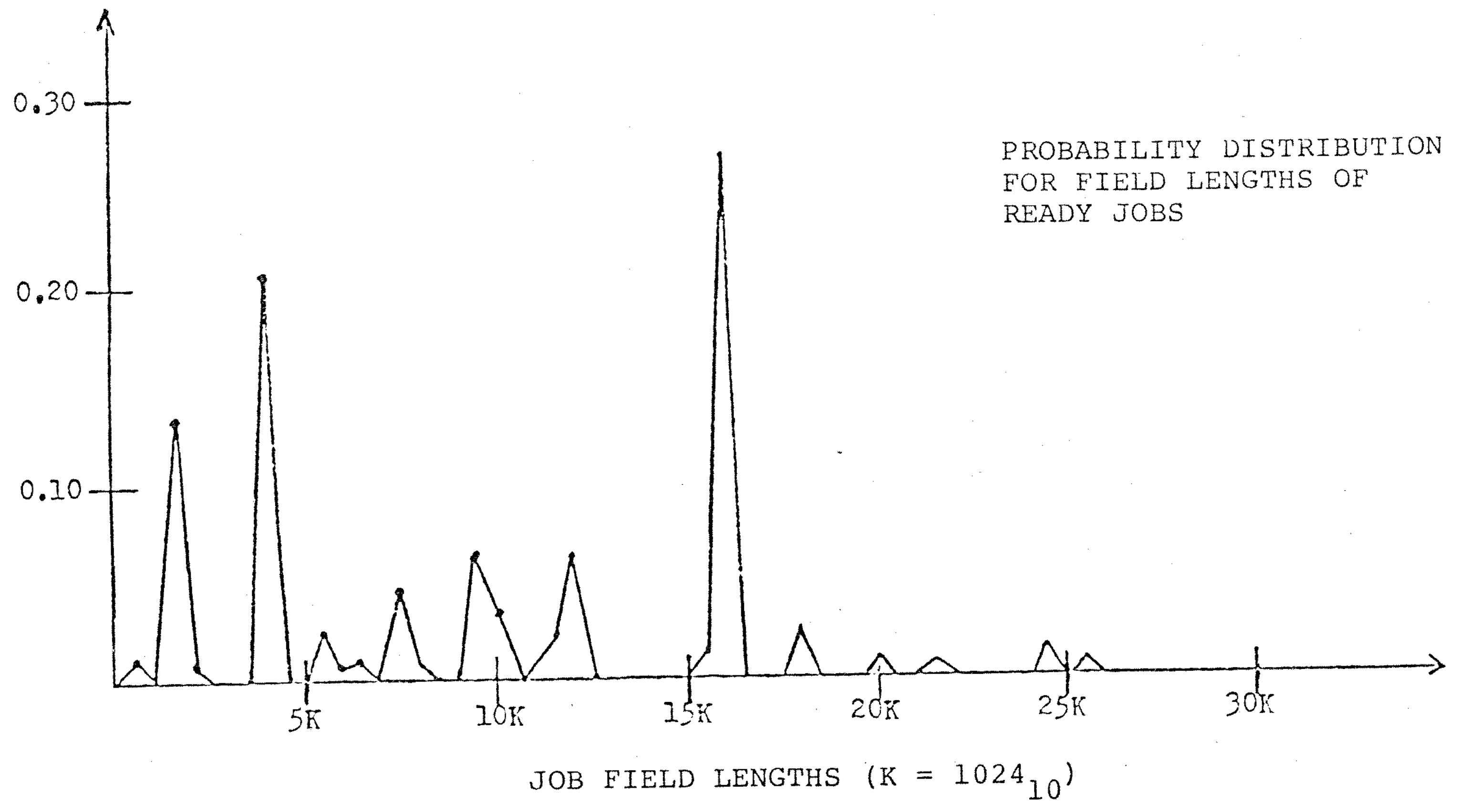


Figure 12. Job field length probability distribution for Tape 1.

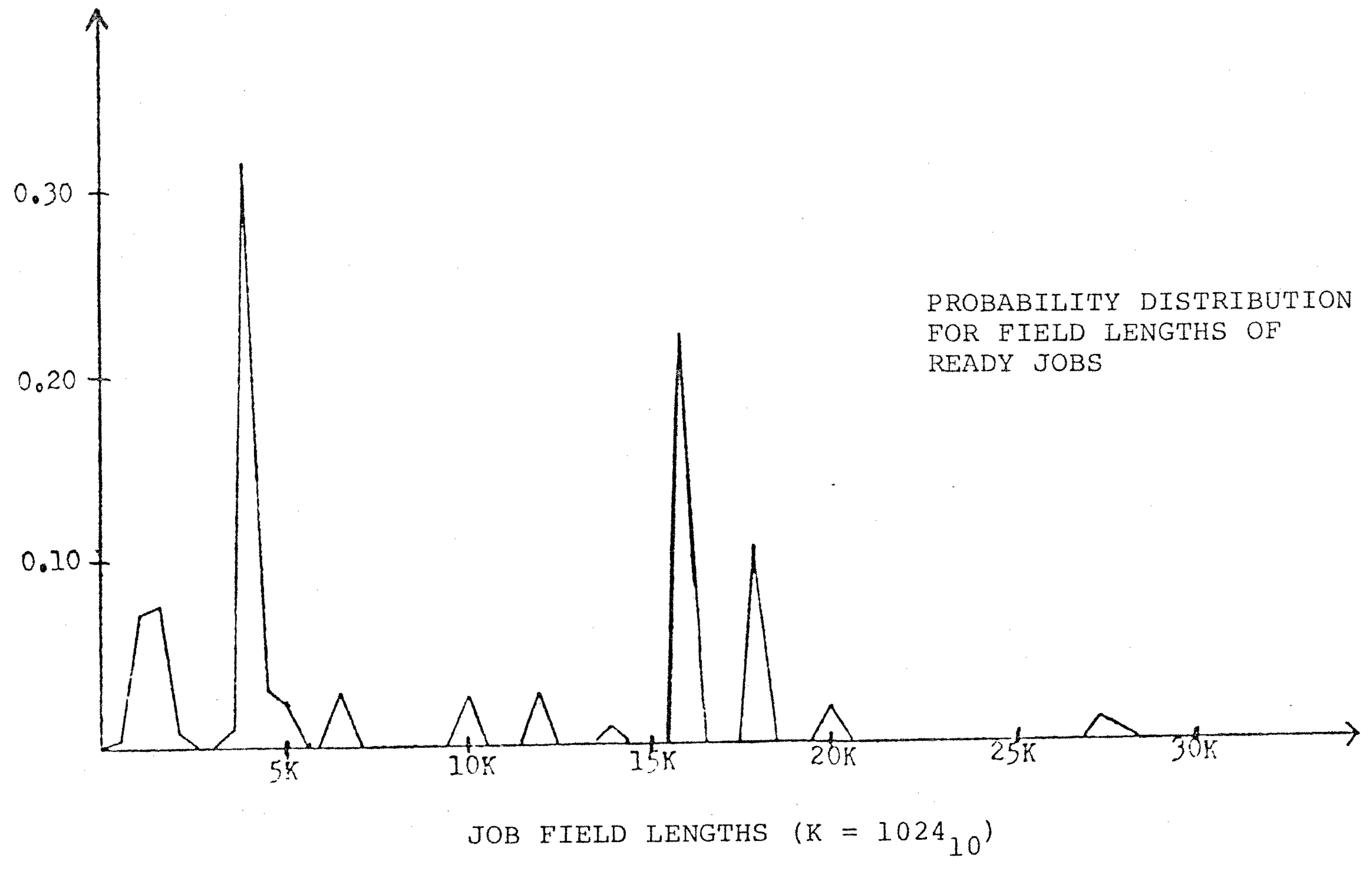


Figure 13. Job field length probability distribution for Tape 2.

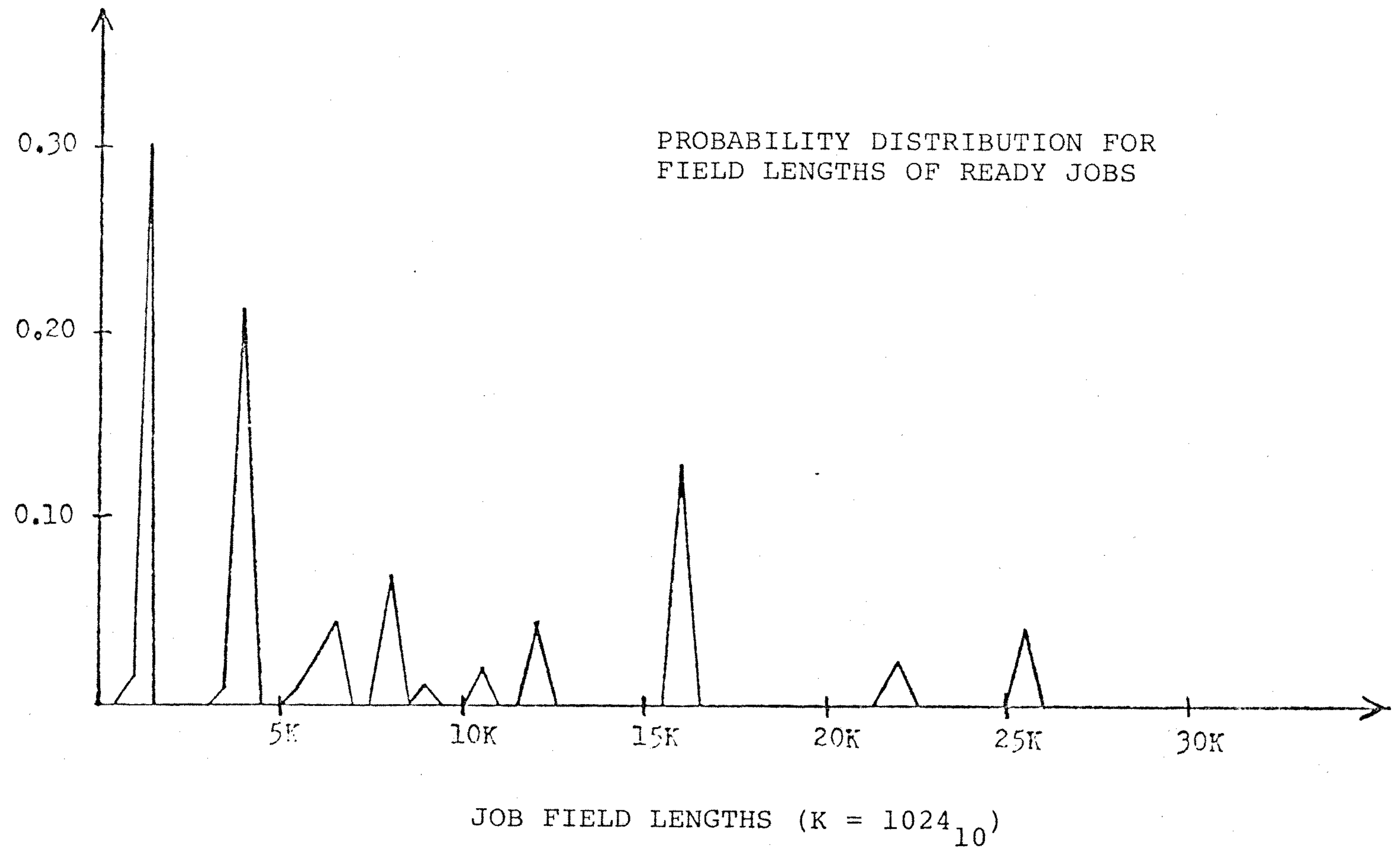


Figure 14. Job field length probability distribution for Tape 3.

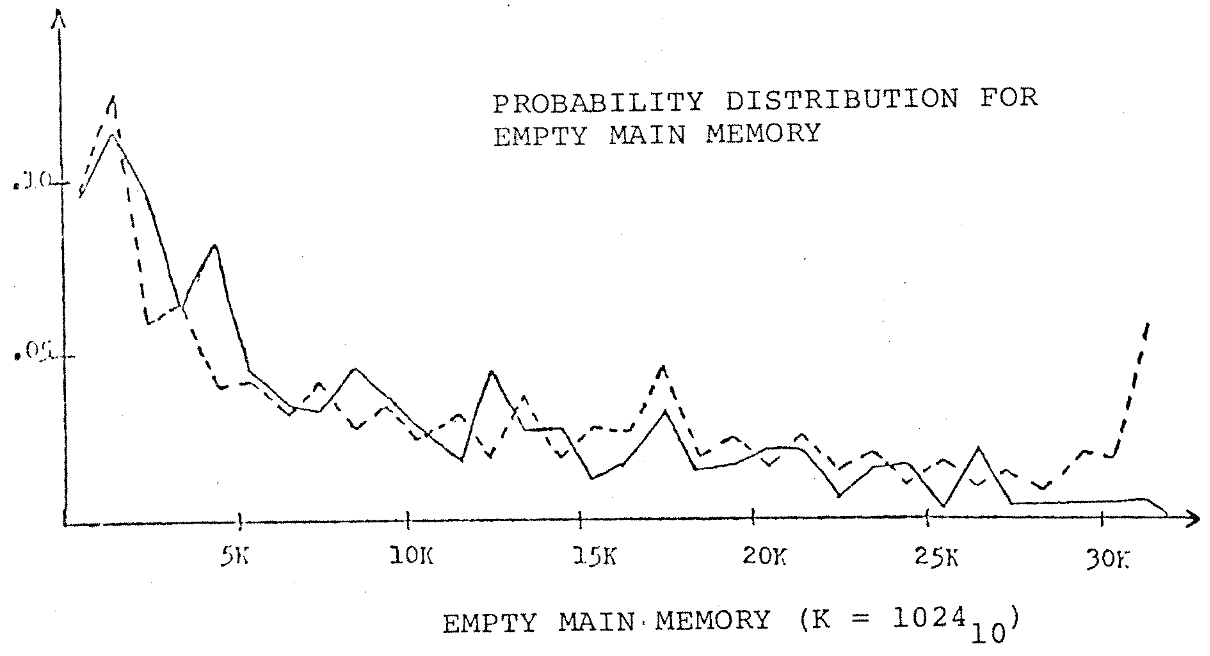
definite common pattern exists for the job field length distribution at this installation.

4.2 Validation of the Model

In order to validate the modeling accuracy, the observed input parameter sets from the three tapes were used for model runs. Validation is determined by the comparison of model output with the corresponding observed data set. This is done for the means of the model output distributions described in Section II, the entire empty memory space distribution, and the entire probability distribution for the number of ready jobs in the system. The comparison of the distribution means for the observed data and the model output for all three tapes is shown in the lower portion of Table 2. The comparison of the complete distributions is shown in Figures 15, 16, and 17.

4.3 Comments on Model Validation

Figures 15, 16, and 17 indicate that this model does provide good approximations for the probability distributions for empty main memory space and the number of ready jobs in the system. Table 2 shows that the mean values of all the output distributions are reasonable approximations to the observed data. Differences in the model output are the result of three general error producing



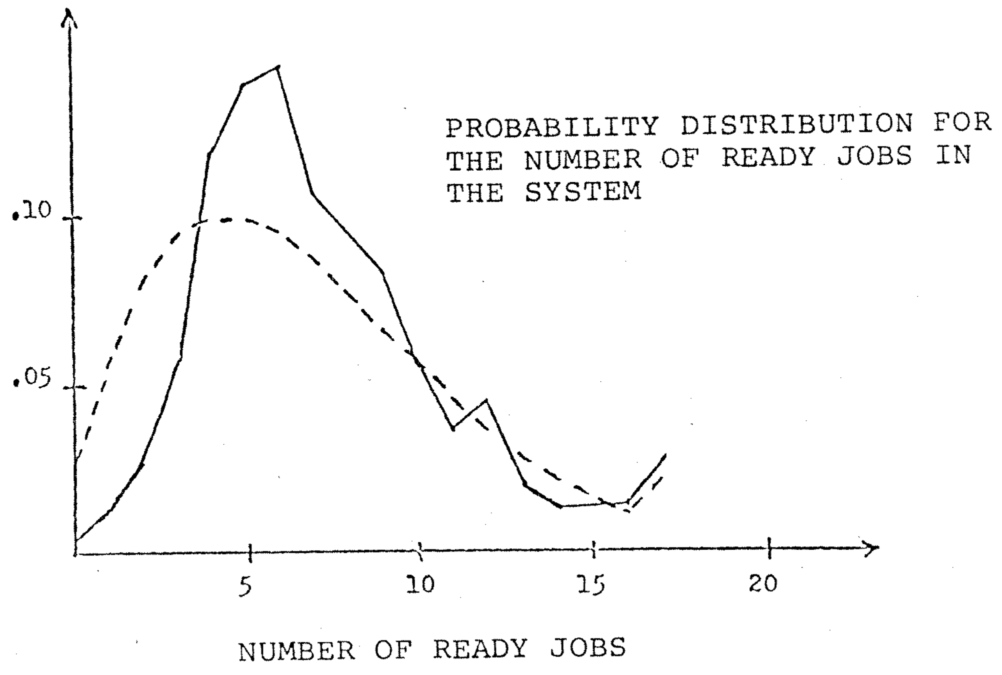
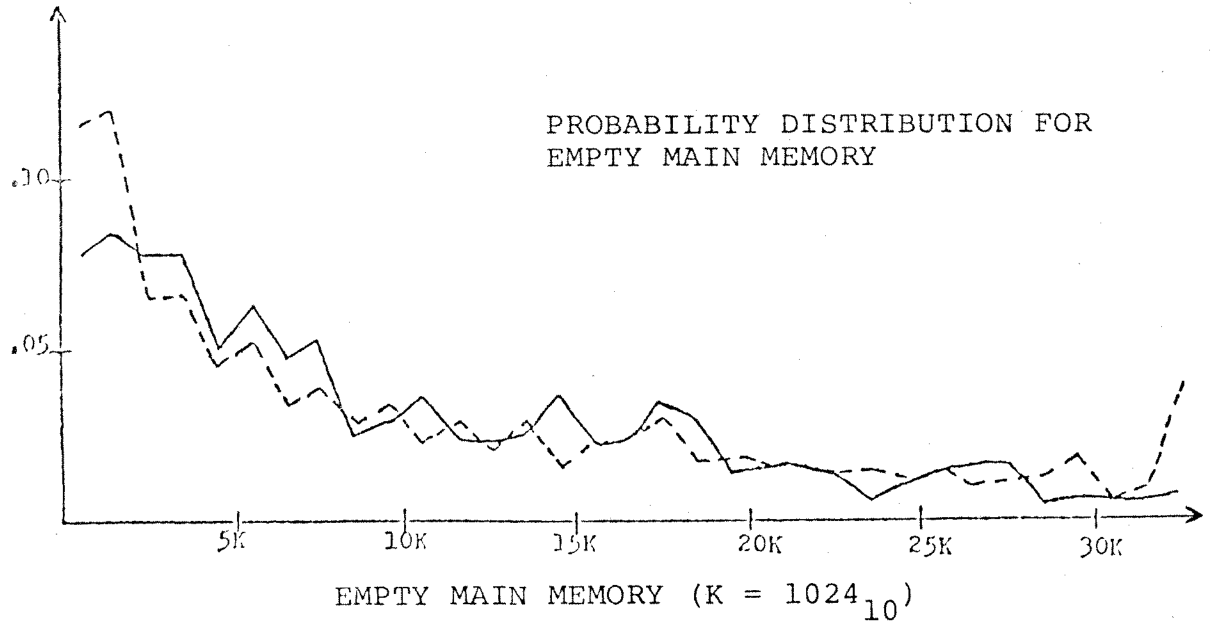


Figure 15. Validation comparisons for Tape 1 (solid) and and Model #1 (dashed).



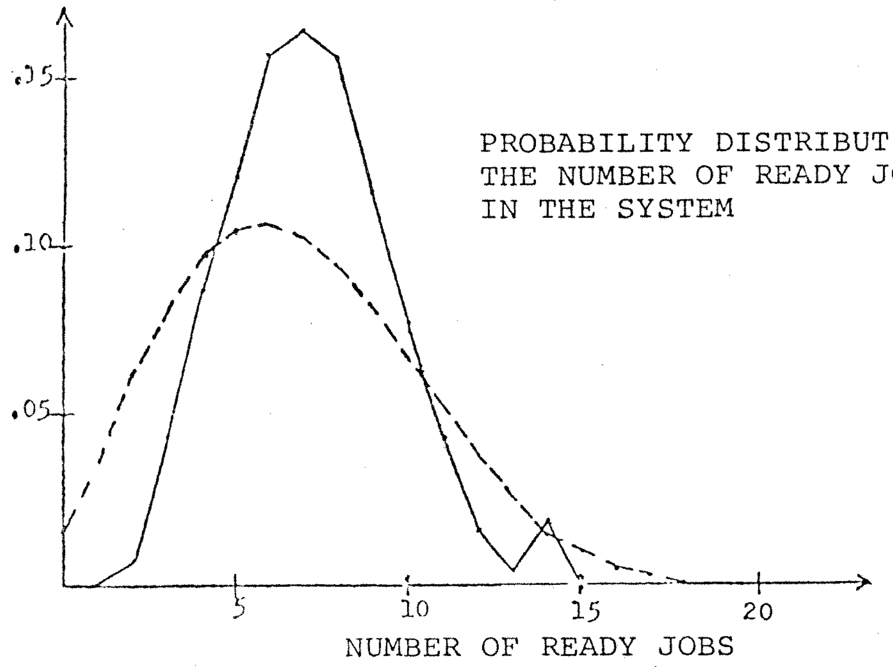
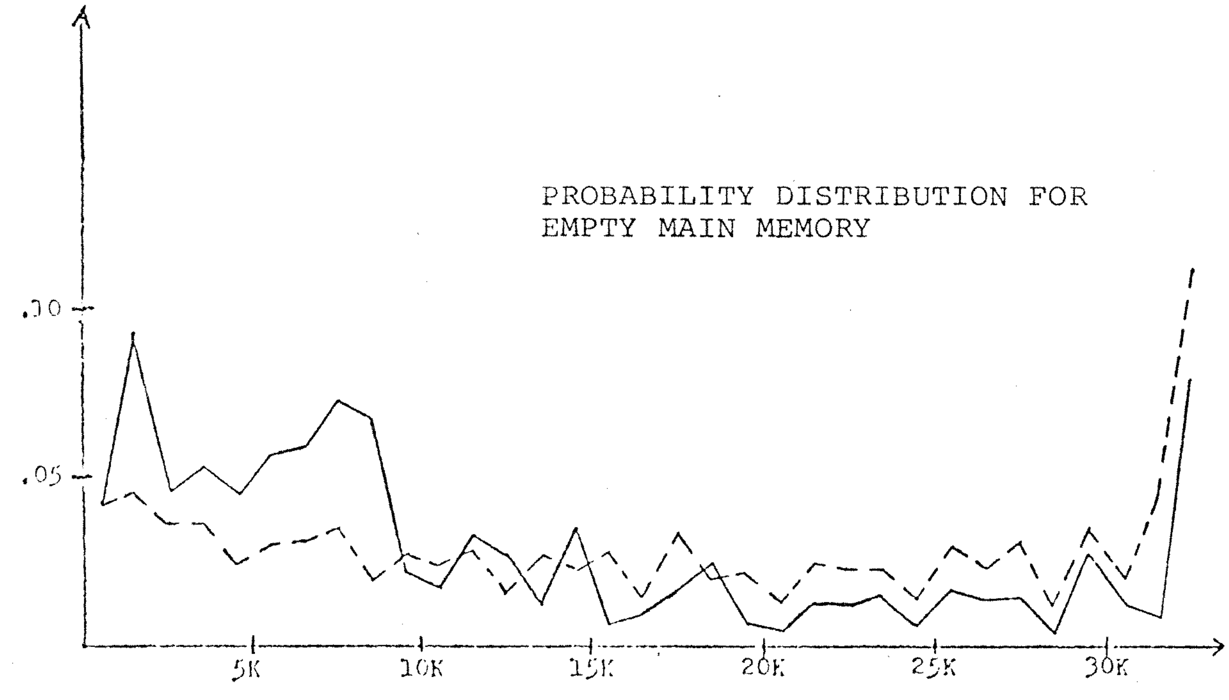
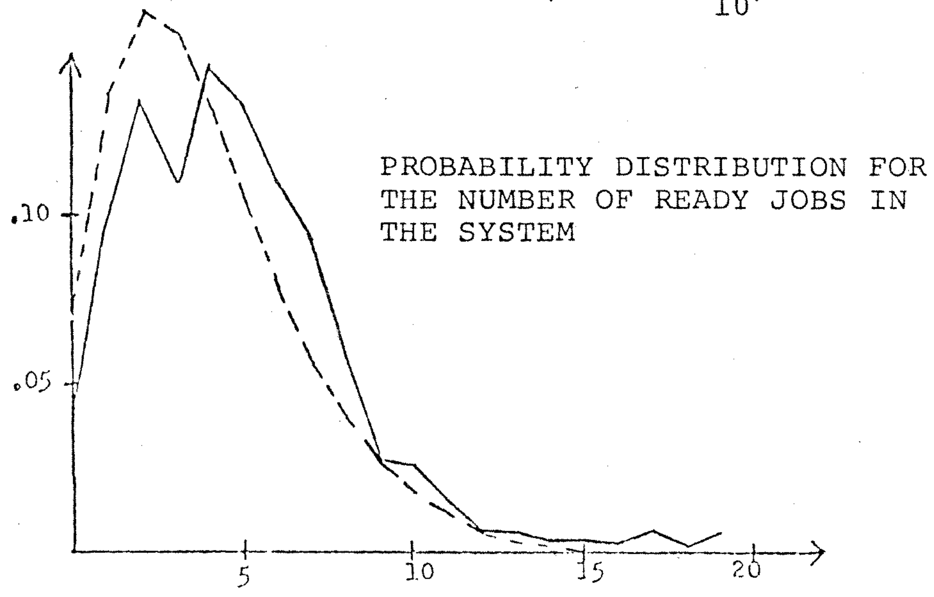


Figure 16. Validation comparisons of Tape 2 (solid) and Model #2 (dashed).



EMPTY MAIN MEMORY (K = 1024_{10})



NUMBER OF READY JOBS

Figure 17. Validation comparisons of Tape 3 (solid) and Model #3 (dashed).

circumstances. First, the assumptions of the model that certain input parameters are constant values leads to application of the model to slightly incorrect input data. Second, the approximations inherent in some of the model calculations produce a degree of error. Third, the data taken from the real system is subject to errors resulting from local fluctuations in the job load or system characteristics. The data for Tape 1, for example, indicates that the fraction of CPU active time used for system processes is 0.852 instead of the assumed value of 0.835. The CPU burst time used in the model was therefore shorter than the actual CPU burst time, resulting in the lower CPU utilization, lower level of multiprogramming, and higher throughput of the model output. For Tape 2 the observed fraction of CPU active time used by the system was 0.819 which again accounts for the model differences. For Tape 3 the average user think time was longer than the assumed 18.7 seconds, the fraction of CPU active time used by the system was 0.805, and the time required for an I/O operation was 119 milliseconds instead of the assumed 100 milliseconds. It is obvious that the assumed constant parameters introduce some error into the model. However, real systems seldom function exactly like any mathematical model and this model does provide a good approximation to a real system. The simplicity and speed of computation gained by the model assumptions make it possible for the model to be used as a tool for systems analysis or design.

SECTION V

EXPERIMENTS WITH THE MODEL

5.1 Design of the Experiments

This model can be used as an effective tool for the analysis of existing systems or for the design of new systems. To fully investigate the potential uses of this model, three basic groups of experiments were conducted. The first group of experiments analyses system performance for changing job load characteristics. The second group determines the effects on system performance of changes in the capabilities of the system. The third group consists of two examples of the model as a tool for system design.

5.2 Experiments With Variations in Job Load Characteristics

All of the experiments in this group used the data taken from Tape 1 as the basis for the model input parameters. The first three experiments attempt to discover the effect of changes in the job field length probability distribution. Experiment 1 used the data from Tape 1 with the job field length-distribution simplified. The new field length distribution (Figure 18) exhibits the same characteristics as the three observed distributions and is easier to analyse. The second experiment was conducted with the new distribution

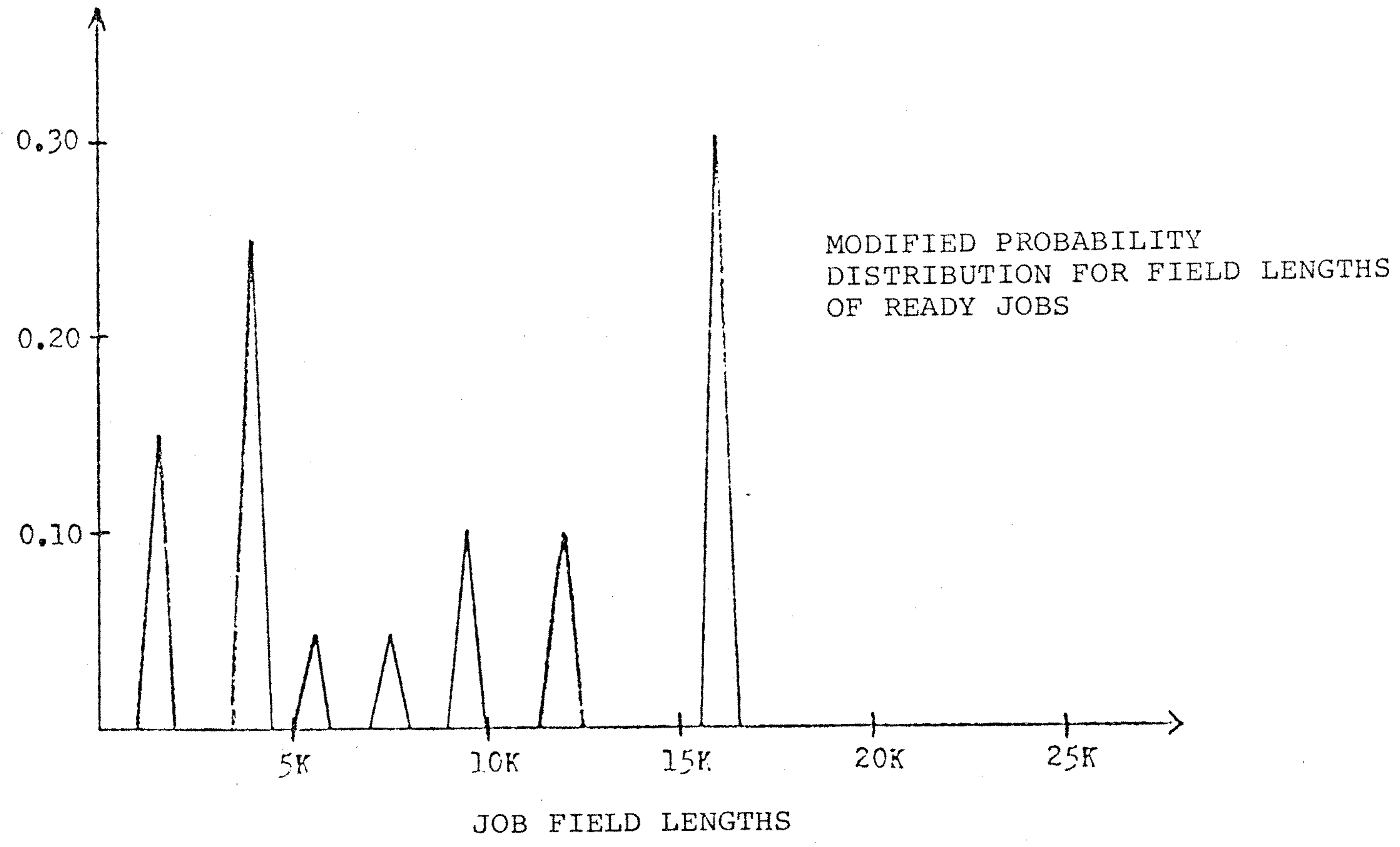


Figure 18. Modified probability distribution for Experiment 1.

shifted by 4K so that the mean was increased by 4K but the variance of the distribution remained the same. Experiment 3 used a job field length probability distribution of two 0.50 probability peaks at 3 units (1.5K) and 32 units (16K). This distribution has approximately the same mean as the distribution of Figure 18 but it has a greater variance. The model output for the three experiments is given in Table 3. Comparison of the results in the table indicate that changes in the variance of the distribution have only a slight effect on system performance, while changes in the mean of the distribution can have a strong effect on system performance. Experiment 3 illustrates a situation in which the system performance is influenced strongly by the limitation of finite main memory.

The next set of experiments on job load variations investigates the system capabilities for variations in the number of users on line. The number of users was varied from thirty to sixty-two. Figure 19 shows the mean response time as a function of the number of users. The exponentially increasing response time indicates that one or more of the system components is being heavily utilized, there by bottlenecking the system. In order to determine the guilty component it is necessary to determine the utilizations of the basic system resources. Figure 20 shows these utilizations as functions of the number of users. The

TABLE 3

RESULTS OF EXPERIMENTAL VARIATIONS OF
JOB FIELD LENGTH DISTRIBUTION

Data	EXP 1	EXP 2	EXP 3
Mean job field length (model units)	17.65	25.65	17.50
Standard deviation of job field length distribution	11.14	11.14	14.50
Response time (seconds)	1.238	2.622	1.354
Multiprogramming level	3.533	2.308	3.932
Throughput (jobs/second)	4.827	4.257	4.783
CPU utilization	0.805	0.710	0.797
Empty memory space (model units)	23.178	20.550	29.380
Overhead memory space (model units)	4.571	6.680	4.336
Number of ready jobs	6.425	11.341	6.838
Available main memory	66.0	66.0	66.0

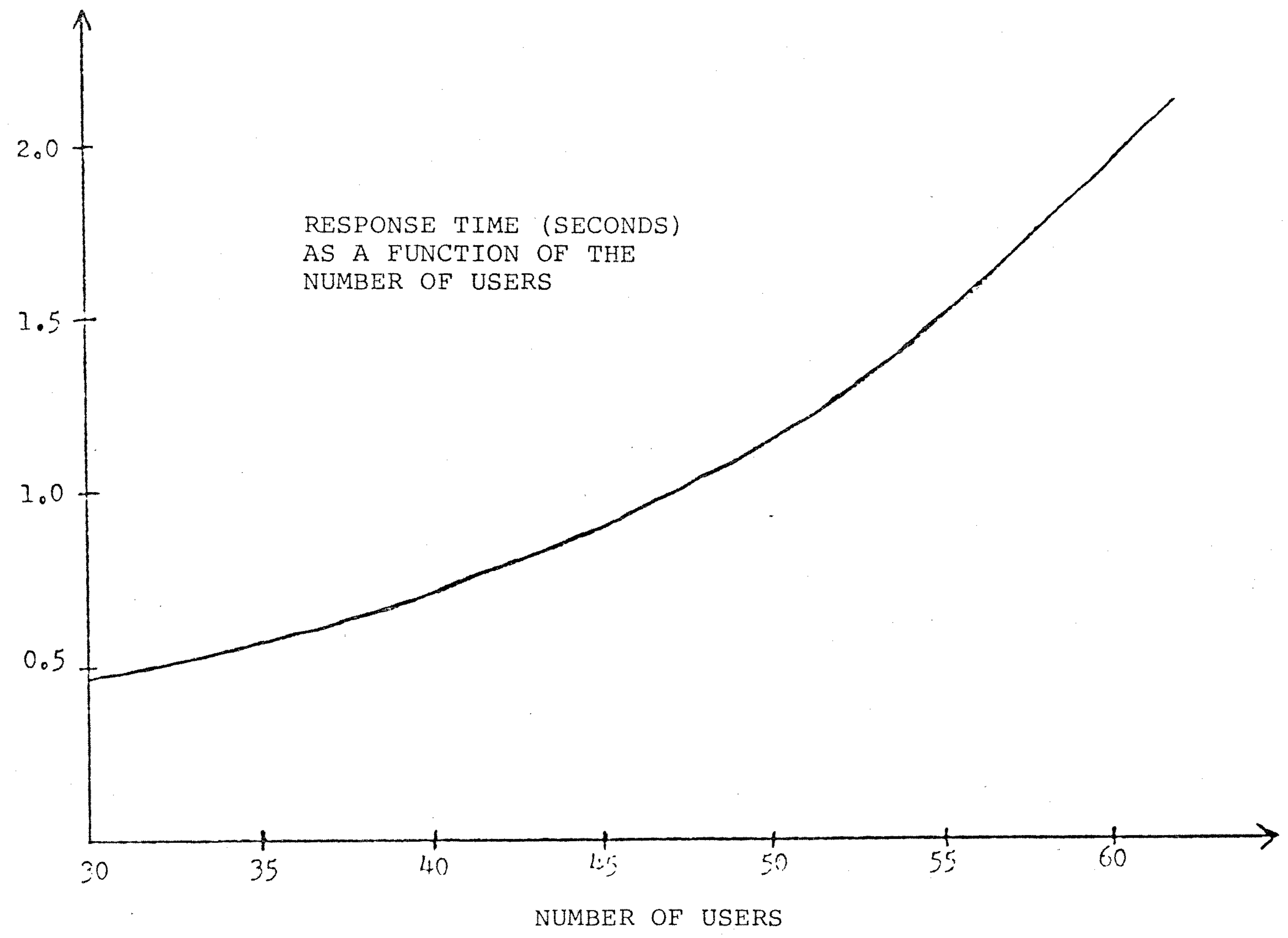


Figure 19. Curve showing response time as a function of the number of users.

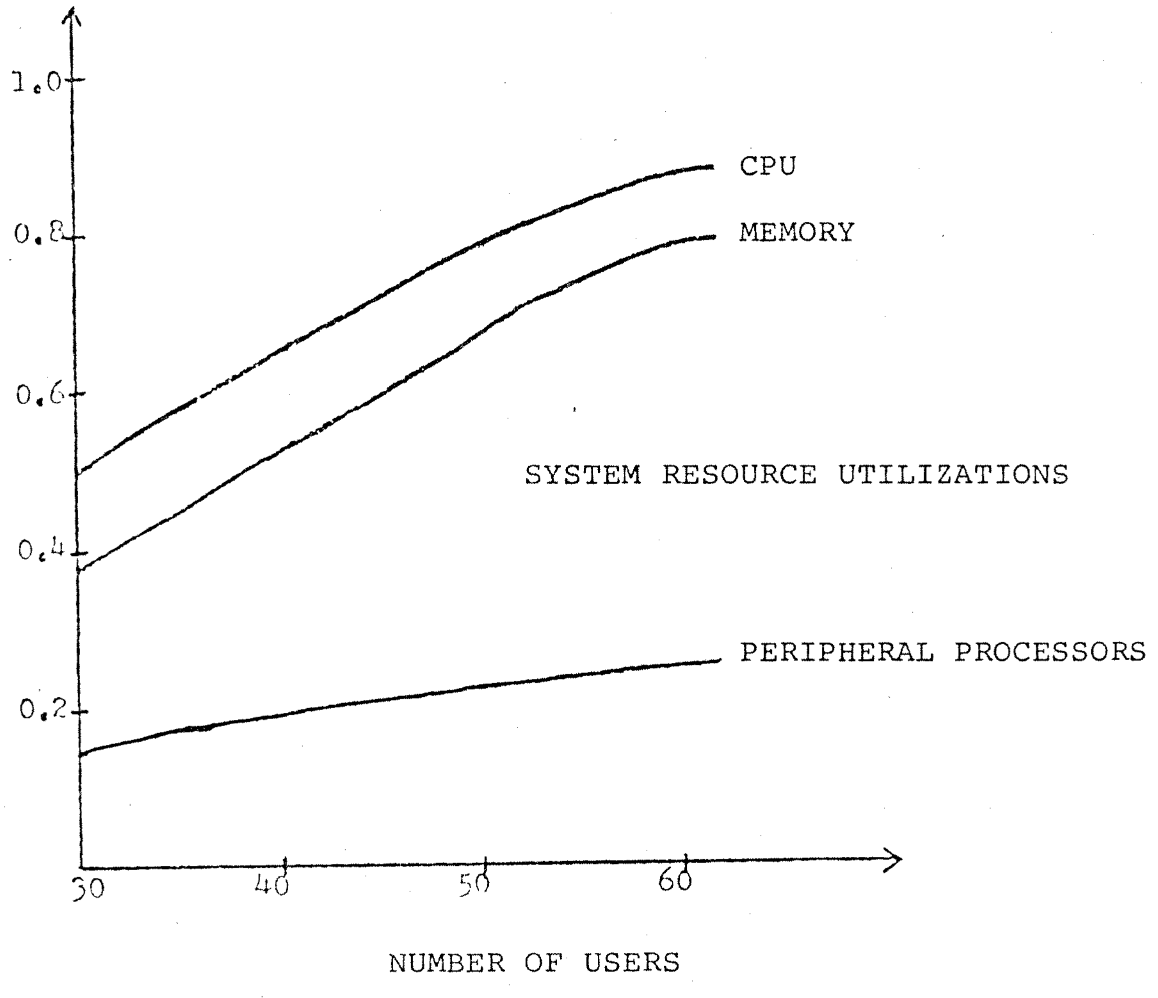


Figure 20. Utilization curves as functions of the number of users.

peripheral processor utilizations were determined by summing the utilizations of all components using a PP (swapping and I/O) and dividing this sum by five, the number of available PP's. From Figure 20 the cause of the bottlenecking can be determined. The CPU is so heavily utilized that jobs are spending much of their time in main memory waiting for CPU service. However, this data does not imply that the available main memory is a secondary bottleneck. If more main memory were available, there would be little improvement in system performance due to the bottlenecking occuring in the CPU service queue. Furthermore, if the effective power of the CPU were increased, the mean degree of multiprogramming would decrease assuming that the job load remains constant. Thus the utilization of main memory would decrease.

5.3 Experiments With Variations in System Characteristics

The purpose of this set of experiments was to approach the analysis of this system from the viewpoint of modifications in the capabilities of the system. Again the data from Tape 1 was used as the basis. Five experimental models were analysed and the system modifications used are described below:

EXP 4. The service rates of the swapping devices were doubled. This provides an upper bound on the increase in system performance resulting from the addition of two peripheral processors which would allow two simultaneous swaps in and out to occur.

EXP 5. The service rates of the swapping devices were set to the service rates of the disk I/O devices to approximate the condition of swapping to disk instead of ECS. This experiment was conducted primarily to determine if the model is reactive to changes in swapping device service times and if the results of Experiment 4 are meaningful.

EXP 6. The service rate of one of the I/O devices was doubled. This provides an upper bound on the performance increase resulting from the addition of one peripheral processor and one disk which would allow four I/O operations to occur simultaneously.

EXP 7. The amount of available main memory was increased by ten model units (5K), corresponding to a reduction of system job and buffer sizes.

EXP 8. The CPU degradation constant was reduced so that users received 25 percent of the CPU service. This corresponds to a reduction of the CPU time currently used by system jobs and the system central monitor.

The results of these experiments and of the validation model upon which they are based are shown in Table 4. The results of Experiment 5 clearly show that the model is

TABLE 4

RESULTS OF EXPERIMENTS ON VARIATIONS IN
THE SYSTEM CAPABILITIES

Data	MODEL #1	EXP 4	EXP 5	EXP 6	EXP 7	EXP 8
Response time (seconds)	1.273	1.069	2.240	1.196	1.135	0.606
Throughput (jobs/second)	4.812	4.900	4.429	4.845	4.870	5.126
Multiprogramming level	3.470	3.216	4.202	3.379	3.639	2.523
Number of ready jobs in the system	6.570	5.749	10.060	6.263	6.031	3.612
Memory space used for overhead (model units)	4.860	2.361	21.948	4.891	5.000	5.064
Empty memory space (model units)	22.917	20.980	22.849	23.934	27.603	33.271
CPU utilization	0.802	0.817	0.738	0.807	0.812	0.566

reactive to swapping device speeds. The results of the other experiments indicate that only slight improvement is observed for increased system capabilities in all respects except for increased effective CPU power. From this group of experiments it is possible to reach the same analytic conclusions that were shown in the first group of experiments, that is, that the CPU is the primary bottleneck in this system. It should also be noted that in Experiment 8 the amount of empty memory space did increase considerably as was suggested in the earlier experiments.

5.4 System Design Using the Model

The following design experiments were conducted using the job load characteristics observed on Tape 1 as the basis for the job loads on the proposed systems. The intent of these experiments is to illustrate the performance of the model as a design tool.

Suppose that a university wishes to obtain a computation facility capable of maintaining a good interactive system for a job load similar to that of Tape 1 with 52 users on line. The system is to have one CPU, three disk I/O devices, two swapping disks, and 64K of fast access central memory. There are also five pool peripheral processors to be used for swapping and I/O operations. The swaps are to disk and require 100 milliseconds with an additional 60 millisecond delay for memory allocation for swaps in. I/O operations are also to disks and they require 100 milliseconds. Due to system programs and buffer areas, there is only 25K of central memory available for user jobs. The planning staff for the university want to know how much CPU power is required for the system to have a mean response time of less than two seconds and if it would be worthwhile to buy another block of 16K of central memory. In order to answer these design questions, the model was tested using job characteristics from Tape 1, system characteristics as described, available central memory for users in a range from 50 model units (25K) to 90 units (45K), and CPU power in a range from that of the CDC 6400 to three times that amount. The results are shown in Figure 21. From these curves it is clear that the university has a choice. The two second mean response time limit can be obtained by either buying a CPU of the same order of power as the CDC 6400 and an additional 16K of main memory or buying a CPU having approximately 1.5 or more times the computing power of the 6400. The final decision, of course, depends upon the relative costs of central memory and computing power, the consideration of future and possibly heavier demands on the system, the financial budget for the installation, and the reliability of the system configurations.

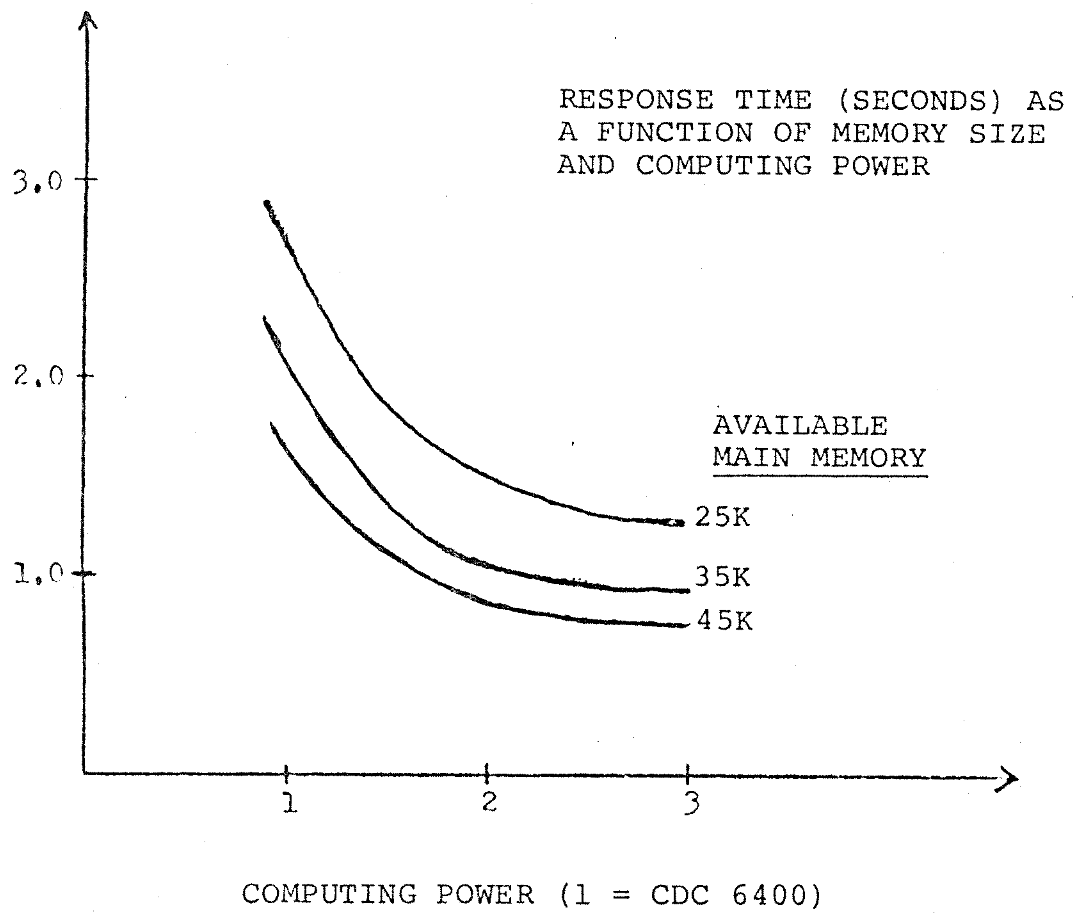


Figure 21. Response time curves as functions of the relative computing power of the CPU and the available amount of main memory.

The next set of experiments deal with system design for strongly I/O bound job loads. Suppose that a system like that at this installation is to be placed in an interactive environment where the jobs require approximately half the CPU service as those from Tape 1, but they require significantly more I/O operations per residence. If ECS is used as the swapping storage medium, then the system will have the same characteristics as the system for Model #1. If, however, the disks are used as the swapping medium and ECS is used for I/O buffering, then the system will perform like the system of Experiment 5 with the important exception that the I/O serivce time will be 15 milliseconds instead of 100 milliseconds on the average. The design problem then is to compare the two systems to discover which performs better for two to five I/O operations per residence. Figure 22 shows this comparison for 40 users on line and using response time as the principle measure of interactive system performance. This clearly indicates that for more than 2.1 I/O operations per residence the system using ECS for I/O buffering performs better. Note that if fewer than two I/O operations are required, the other system performs better especially for larger numbers of users when the swap rate becomes high.

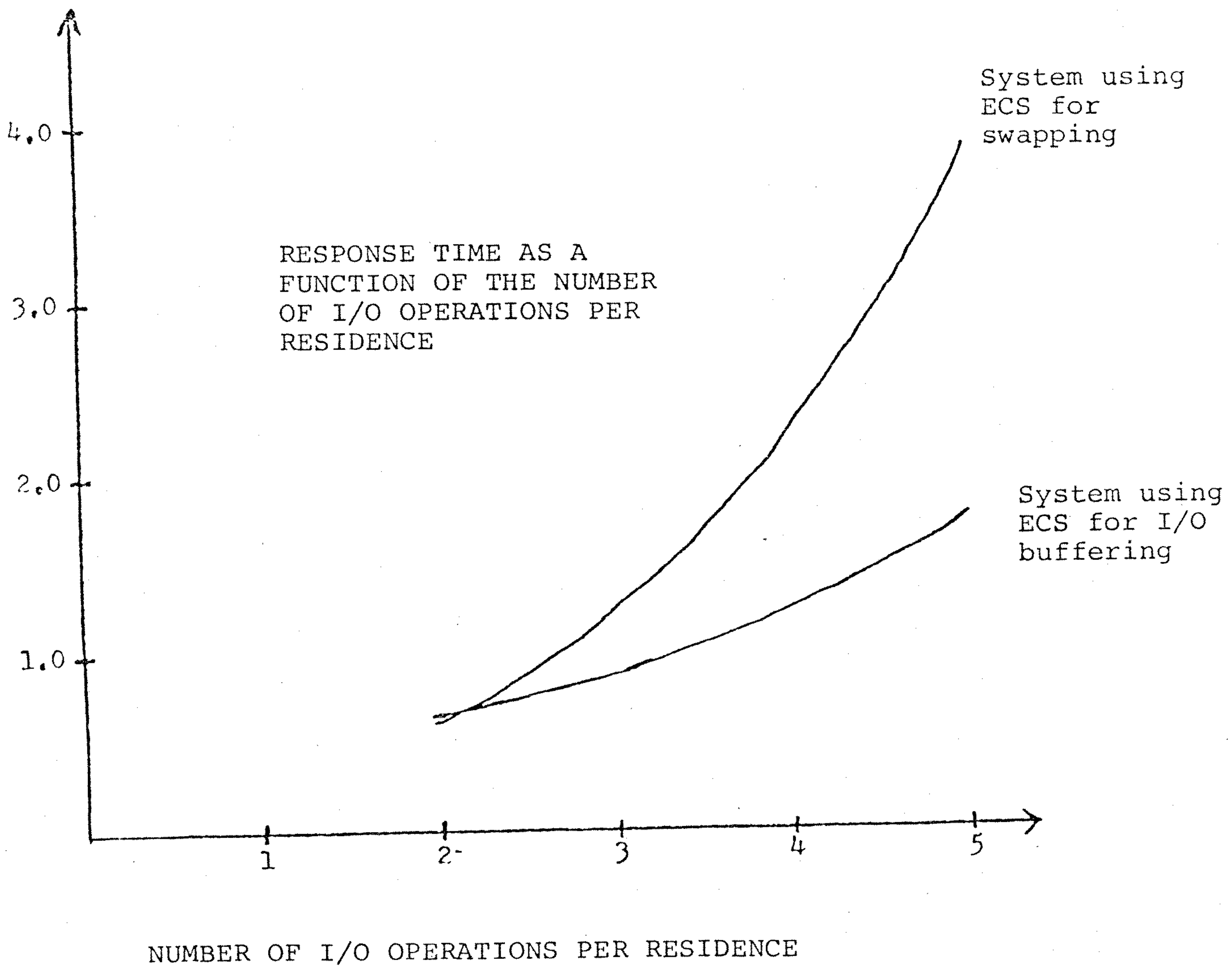
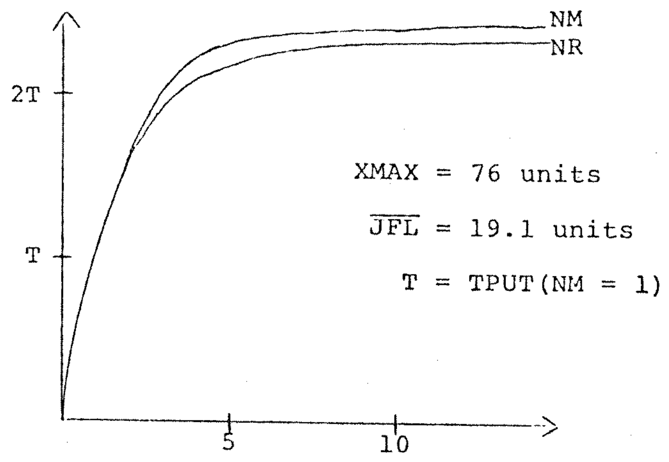


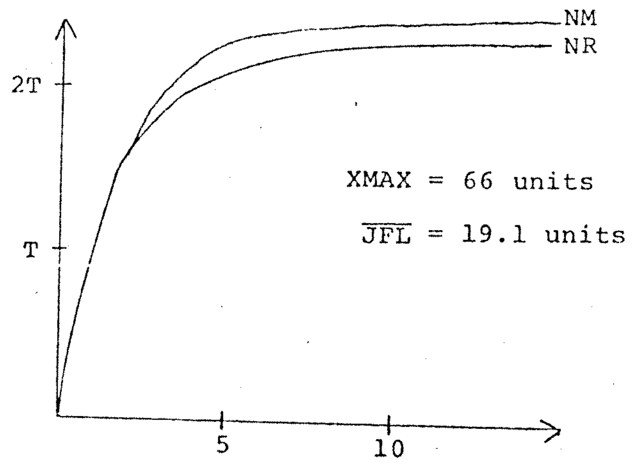
Figure 22. Comparison of two methods for using ECS.

5.5 Comments on the Experiments

The first two groups of experiments show that it is possible to use the model for system analysis. This may be done by suggesting changes in the job load characteristics and noting the effects on system performance or by proposing various system modifications and observing the effects. Furthermore, the applicability of the model to design problems has been illustrated in the third group of experiments. There are an infinite number of possible changes that can be proposed for the job load characteristics and system capabilities and no attempt has been made to provide an exhaustive set of examples. The intent of these experiments was to demonstrate the flexibility of the model by investigating some subset of the possible situations.

It is possible to reach several conclusions from an overview of all of the experiments with the model. First, the main memory of an interactive system limits system performance if the throughput of the system with infinite main memory is higher than the throughput of the system with the given, finite amount of main memory. Figure 23 displays three sets of throughput curves. One curve of each set (marked NM) is the graph of system throughput as a function of the number of jobs in main memory. If infinite main memory is assumed, these same curves represent the system throughput as a function of the number of ready jobs in the





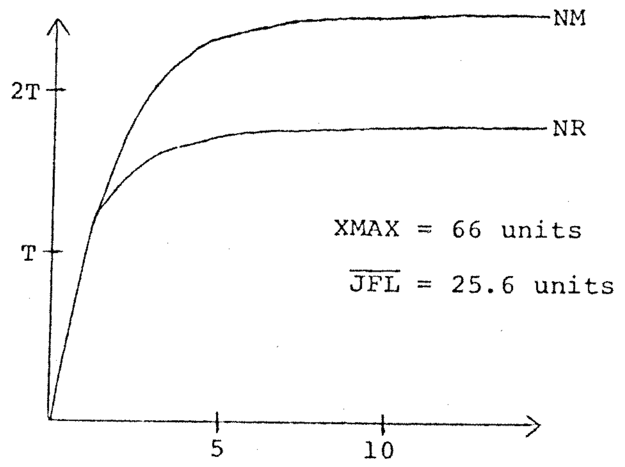


Figure 23. Comparison of throughput for systems with finite and infinite main memory.

system because all ready jobs will fit into an infinite memory. The second curve of each set (marked NR) is the graph of system throughput as a function of the number of ready jobs in the system if the finite memory size is considered. The top set of curves was taken from Experiment 7 which used Tape 1 data with an increased main memory size of 76 model units. Very slight memory limitation is indicated. The middle set of curves was constructed from Tape 1 data and slight system performance degradation is again indicated. The bottom set of curves was taken from Experiment 2 which used a modified job field length distribution having a higher mean. Substantial degradation of system performance is observed indicating that for this case memory size is severely limiting system performance for more than two ready jobs in the system.

A second conclusion, equivalent to the first, is that main memory size is a limitation of system performance if, when the system achieves its maximum throughput, no other system resource is saturated. If another resource is saturated, then an increase in the amount of available main memory will not significantly increase the system throughput although it may increase the level of multiprogramming. This characteristic is typical of all passive resources such as channels, peripheral processors, and memory.

The curves shown in Figure 24 were constructed from the set of experiments using the data from Tape 1 and varying the number of users. They show the number of units of main memory that are occupied or empty as functions of the mean number of ready jobs in the system. The dotted lines indicate estimated extensions of the curves. Note that if the mean job field length requirement is $\overline{\text{JFL}}$, then the amount of empty main memory space approaches 0.68 $\overline{\text{JFL}}$, the amount of occupied main memory (including overhead space) approaches 2.82 $\overline{\text{JFL}}$, and the level of multiprogramming approaches 4.30 jobs. The number of user jobs of average size that should fit into the mean occupied main memory is 2.82, but the number of user jobs actually located in that occupied space is 4.30. This indicates that in this job load environment, the smaller than average user jobs are being serviced and the larger user jobs are blocked for perhaps an indefinite period of time. This situation applies only to systems which are operating in regions of near maximum throughput because the memory saturation necessarily implies an upper bound on system throughput. It is hypothesized that any interactive system which uses a first fit type of memory scheduler can be driven to a saturation state which exhibits similar characteristics. This problem could, of course, be alleviated by introducing a wait time priority system into the scheduling algorithm.

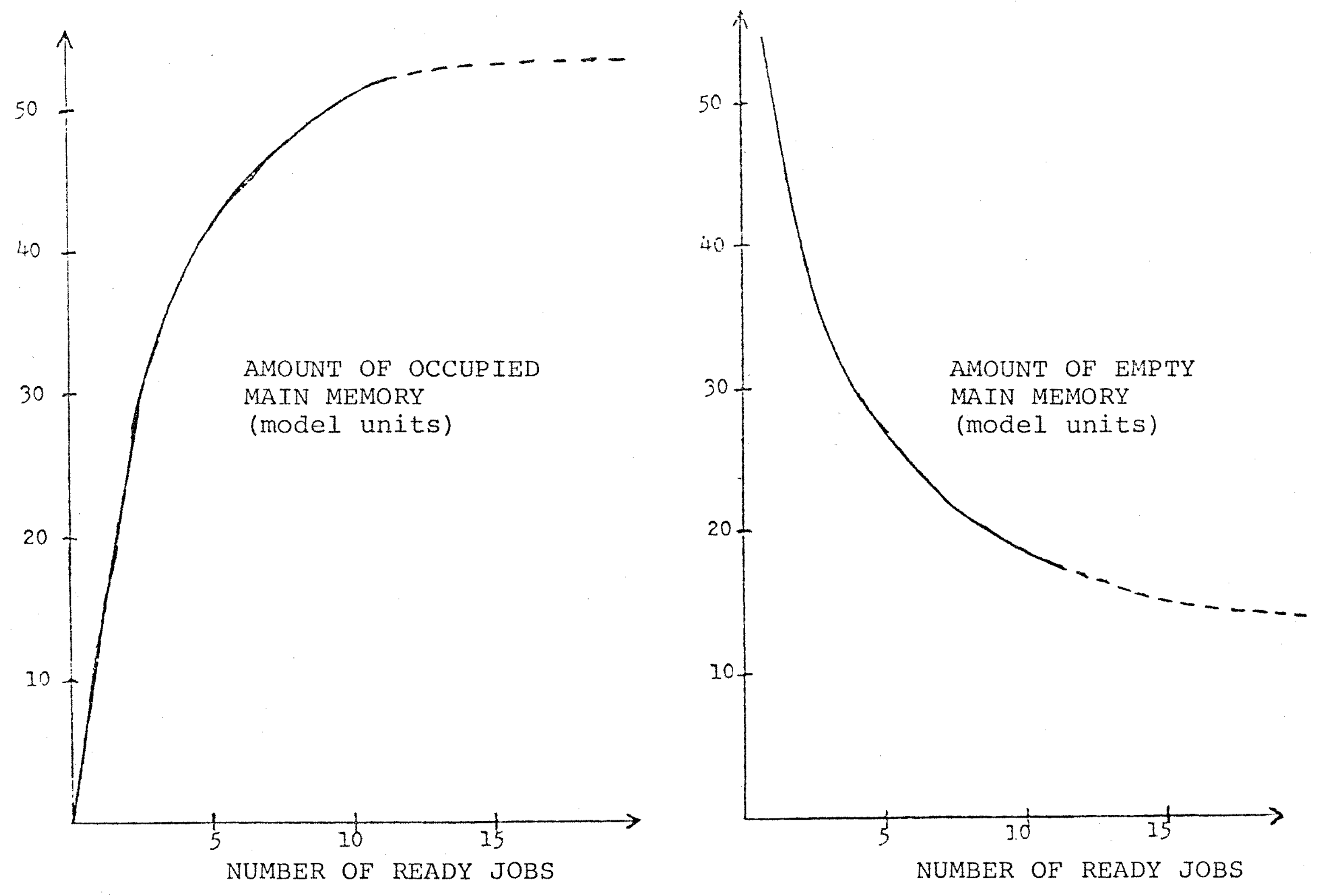


Figure 24. Memory utilization as a function of the number of ready jobs in the system.

It is, however, interesting to determine some measure which indicates that the indefinite wait problem is apparent. Clearly, as the number of users decreases, the difference between the level of multiprogramming and the occupied memory space divided by $\overline{\text{JFL}}$ decreases. For example, if there are 48 users on this system, the amount of empty main memory is 1.36 $\overline{\text{JFL}}$, the amount of occupied main memory is 2.14 $\overline{\text{JFL}}$, and the level of multiprogramming is 3.09 jobs. This model data leads to the following hypothesis: For a large interactive system using a first fit type of scheduling algorithm, the mean amount of empty main memory should be at least 1.5 times the mean job field length if consistent response time is desired for all users. This hypothesis assumes that the amount of empty main memory can be used as a measure of large job blockage independent of the total amount of available main memory and of the system configuration. It is possible that this hypothesis can be extended to other scheduling algorithms such as shortest job first, longest job first, and best fit.

SECTION VI

CONCLUSIONS

6.1 Summary of Results

An analytic model has been proposed for the analysis of the effects of finite main memory on a class of large scale interactive systems. This model uses a four phase method of analysis which includes division of the overall system into an in-memory portion and an out-of-memory portion. The concept of superposition of these two systems based on a given job field length probability distribution and a given amount of available main memory has been introduced and discussed. The model has been validated for three sets of data taken from the UT-2D interactive system which uses a CDC 6400. The capability of the model to produce results pertinent to systems analysis has been demonstrated. The model has also been shown to be an effective design tool. It is possible to conclude from an overview of the experiments with the model that memory utilization can be used as an effective measure of the performance region in which an interactive system is operating.

6.2 Possible Extensions of this Research

This model has been applied only to a non-paged memory system. The basic structure of the model, however, can be used to model many memory architectures. It should be possible with appropriate parameterization to analyse a virtual memory system using these techniques. This modeling analysis can also be applied to interactive systems which have different memory scheduling algorithms if a probabilistic analysis similar to that of Phase 2 can be developed for the scheduling algorithm. Another possible extension of this work would be to add phases of analysis to include in the model the limitations of finite sized ECS. The in-memory portion of the model (Phase 1) can be extended to a more detailed level. It is, however, not clear that any increased accuracy obtained would justify the additional complexity.

BIBLIOGRAPHY

1. Baskett, F., "Mathematical Models of Multiprogrammed Computer Systems", Computation Center TSN-17, (Ph.D. Dissertation), The University of Texas at Austin, 1971.

2. Baskett, F., Browne, J. C., and Lan, J., "The Interaction of Multiprogramming, Job Scheduling, and CPU Scheduling", Proceedings FJCC AFIPS, December 1972, 13-21.

3. Baskett, F., Browne, J. C., and Raike, W., "The Management of a Multi-level Non-paged Memory System", Proceedings SPJCC, May 1970, 459-465.

4. Brice, R. S., "A Study of Feedback Coupled Resource Allocation Policies in a Multiprocessing Computer Environment", Computation Center TSN-35, (Ph.D. Dissertation), The University of Texas at Austin, 1973.

5. Buzen, J., "Queueing Network Models of Multiprogramming Systems", (Ph.D. Dissertation), Harvard University, 1971.

6. Chandy, K. M., "The Analysis and Solutions for General Queueing Networks", Proceedings Sixth Annual Princeton Conference on Information Sciences, Princeton University, March 1972.

7. Chandy, K. M., Howard, J. H., Keller, T. W., and Townsley, D. J., "Local Balance, Robustness, Poisson Departures, and the Product Form in Queueing Networks", Department of Computer Sciences TR-15, The University of Texas at Austin, 1973.

8. Chandy, K. M., Keller, T. W., and Browne, J. C., "Design Automation and Queueing Networks", Proceedings Ninth Annual Design Automation Conference, June 1967, 357-367.

9. Denning, P. J., "Virtual Memory", Computing Surveys Vol. 2 No. 3, September 1970, 153-190.

10. Fuchel, K., and Heller, S., "Considerations in the Design of a Multiple Computer System with Extended Core Storage", CACM Vol. 11, 1968, 334-340.

11. Howard, J. H., "A Large Scale Dual Operating System", Proceedings ACM 73, Atlanta, Georgia, August 1973, 242-248.

12. Johnson, D. S., "A Process-oriented Model of Resource Demands in Large Multiprocessing Computer Utilities", Computation Center TSN-29, The University of Texas at Austin, 1972.

13. Knight, D. C., "An Algorithm for Scheduling Storage on a Non-paged Computer", Computer Journal Vol. 11, 1968, 17-22.

14. Krishramoorthi, B., and Wood, R. C., "Time-shared Computer Operations with Both Interarrival and Service Time Exponential", Journal of ACM Vol. 13, 1966.

15. Martin, J., Design of Real-time Computer Systems, Prentice-Hall, 1967.

16. Pewitt, T. C., and Su, S. Y. W., "Resource Demanded Paging and Dispatching to Optimize Resource Utilization in an Operating System", First Annual SIGME Symposium on Measurement and Evaluation, February 1973, 29-36.

17. Sherman, S. W., "Trace-driven Modeling Studies of the Performance of Computer Systems", Computation Center TSN-30, The University of Texas at Austin, 1972.

18. Thornton, J. E., Design of a Computer--the Control Data 6600, Scott, Foresman, and Company, 1970.

VITA

Robert McBurnie Brown was born in Pulaski, Virginia on July 9, 1950, the son of Bruce Robert Brown and Alma June Brown. After completing his high school education at William Fleming High School in Roanoke, Virginia, he attended Florida State University. He received a Bachelor of Science degree from the School of Engineering Science at Florida State in June, 1972. At that time he also received a commission as a second lieutenant in the U. S. Army through the R.O.T.C. program. In September of 1972 he entered The Graduate School of The University of Texas at Austin in the Department of Computer Sciences. He married Rhonda Gaffney in December of 1972.

Permanent address: 2703 Cedarhurst Avenue, NW
Roanoke, Virginia

This thesis was typed by Terry's Typing Service.