STUDY OF RESOURCE ALLOCATION
IN COMPUTER SYSTEMS BY SIMULATION

A Thesis
Presented To
the Faculty of Graduate Studies and Research
The University of Manitoba

In Partial Fulfilment
of the requirements for the Degree
Master of Science
in the Institute for Computer Studies

by
Bruce H. McDonald
February 1969

ABSTRACT

Various job-scheduling algorithms based upon round-robin scheduling with a variable time-slice are studied by simulation under various conditions of system utility. System utility is varied by varying the inter-arrival time of jobs and by varying the quantity of main storage available to jobs. A measure of performance is based upon curves which represent the cost of delay to jobs. This measure of performance is very similar to a performance criterion previously used.

An empirical equation is developed which relates performance to the utilities of the central processor and main storage by means of a variable coefficient. For simulated conditions of utility less than 0.60 the empirical equation is found to agree fairly well with the simulated performance with a constant coefficient value which depends on the number of priority classes. For simulated conditions of higher utility, variability in the coefficient is required to fit the empirical equation to the simulated performance. As utility is increased beyond 0.80, simulated performance becomes a function of the job-scheduling algorithm and when utility reaches unity, simulated performance becomes a function of the number of jobs in the job stream as well.

Those algorithms which tend to select the shorter jobs from the waiting queue for processing during the simulation produce a somewhat better performance.

ACKNOWLEDGEMENTS

I wish to express my sincere appreciation to Professor T. A. Rourke, my thesis supervisor, for his guidance, direction, advice and criticism which have been so helpful to me in this research.

I wish to thank Professors S. R. Clark and A. Wexler for their time spent in reading this thesis, and for their helpful comments and criticisms.

I wish to thank Mr. Parker (Bud) Caufield and his computer operators for the assistance and cooperation given to me during the simulation studies.

Also, I wish to thank my typist, Mrs. Betty Kimery, for so accurately deciphering my hieroglyphics while preparing this thesis.

TABLE OF CONTENTS

PAGE

LIST OF TABLES

LIST OF FIGURES

FIGURE PAGE

NOMENCLATURE

The following notations are used in this thesis:

A_{jI} : actual interarrival time for job j of priority class I, (page 15).

A_{jI} : arrival time of job j in priority class I, (page 12).

A'_{jI} : earliest time when job j of priority class I can be loaded into storage, sum of arrival time (A_{jI}) and device time (d_j), (page 16).

A^*_{jI} : time when job is loaded into storage. Sum of earliest possible time (A'_{jI}) and time spent in waiting queue, (page 23).

A_s : input parameter determining which queue selection algorithm to be used, (page 45).

b : storage time constant, giving the rate at which mean storage requirement increases with central processor time request, (page 17).

B : maximum mean storage requirement for jobs in the job stream, (page 17).

C_o : coefficient used to fit empirical equation to observed simulated results, (page 52).

d : input parameter device time number, (page 19).

d_j : assumed device time of job j, (page 12).

D_o : time demand on central processor, (page 31).

D_1 : units·time demand on main storage, (page 32).

ε : small quantity used to represent variation from unity where lack of attention actually becomes infinite, (page 52).

g_j : augmented lack of attention used in quantum allocation routine, (page 42).

I_j : priority number of job j in priority class I, (page 12).

job j : the j^{th} job in the job stream, (page 12).

k : a priority constant, (page 13).

L : the number of priority classes I, (page 12).

m_I : mean central processor time request, jobs of priority class I, (page 14).

M : mean storage requirement for jobs j with central processor time request t_j, (page 16).

n_I : the number of jobs of priority class I, (page 13).

N : the number of jobs in the job stream, (page 11).

O_1 : storage occupancy, (page 33).

p_j : penalty of job j , (page 25).

p_j^* : penalty of job J at completion of job j, (page 25).

P : penalty function, (page 24).

Q : length of waiting queue, (page 35).

r_I : mean interarrival time, jobs of priority class I, (page 14).

R_j : storage requirement of job j , (page 12).

$\overline{R}_I$: mean storage requirement of jobs in priority class I, (page 33).

$\overline{R}_M$: sum of mean storage requirements of jobs in all priority classes ($\Sigma\overline{R}_I$), (page 33).

σ_N : standard deviation in system penalty, (page 34).

s : number of slots in the execution list filled by jobs, (page 42).

s_o : time supply of central processor (unity), (page 31).

S_1 : units·time supply of main storage (page 32).

τ_j : time devoted to job j by the computer, includes device time (d_j) and central processor time given, (page 25).

t_j : actual central processor time requires of job j, (page 12).

T : total time (simulated) for simulation run, (page 32).

T_j : time at completion of job j, (page 21).

ΔT_j : duration of job j, elapsed time between job arrival (A_{jI}) and job completion (T_j), (page 21).

T_j^1 : time between arrival of job j in storage and completion of job j, (page 24).

T_R : round robin cycle time, (page 41).

T_{Rj} : time slice allocated to job j by the variable time slicing algorithm, (page 41).

T_{RM} : minimum time-slice, (page 41).

T_{sc} : supervisor cycle time, (page 39).

μ_j : lack of attention of job j, (page 24).

μ_j^* : lack of attention of job j at completion of job j, (page 24).

μ_M : mean lack of attention for simulation run, equals the system penalty when $p_i = \mu_i$, (page 49).

U : utility, defined as time ratio of demand to supply in the competition concerned, (page 50).

U_o : central processor (C.P.U.) utility, (page 27).

U_1 : storage utility, (page 32).

W_o : number of slots in the execution list, (page 34).

W_1 : number of units of main storage initially available to jobs in the job stream, (page 32).

W_t : number of units of main storage available at any time t during the simulation run, (page 42).

CHAPTER I

INTRODUCTION

1.1 Purpose of the Research

In this study use is made of the computer system simulation model which has been used by Chai to investigate some of the effects of a particular time-slicing job-scheduling algorithm (1)[1]. This model is the first stage of a more comprehensive computer system simulation model which will include the operations of logging-in, loading into core, scheduling and outputting of the results. This first stage may be considered as a model of a hypothetical computer system which incurs no overheads to load jobs into core, and which processes to completion jobs which require no input or output during their execution. In the study conducted by Chai (1) it was assumed that the physical size of the main store was large enough to accomodate any three jobs. However, no attempt was made to consider the possibility that this same main store may be able to accomodate more or fewer than three jobs during certain periods of time.

The purpose of this research is to examine the service to computer users as the number of units of main

---

[1] Notation (n) refers to the n^{th} entry in the list of references.

storage available changes for a variety of different job-scheduling algorithms. A job-scheduling algorithm may be separated into two sections, the first is the logic which determines which of the waiting jobs is to be accepted next (queue selection algorithm) and the second is the logic which determines how a job is serviced once it is accepted (quantum allocation routine). More specifically, the purpose of this research is to examine various queue selection algorithms under conditions of varying storage.

1.2 Methods and Scope of the Research

A small modification of the model used by Chai, in order to keep account of how many units of main storage are occupied and how many units are free, makes the model suitable for the study of queue selection algorithms. The main memory is attributed with a size and each job within the job stream requests a particular amount of this main storage. Providing there is sufficient storage available for a particular job, the job can be accepted for execution. No account is taken of the spatial arrangement of jobs within the main storage and therefore the organization within the main memory is taken to be either non-contiguous, that is, the main storage is split into units of an arbitrary size, but jobs need not occupy these units in a contiguous manner, or alternatively, contiguous organization with negligible time of relocation of jobs within the main storage.

The basis of a measure of computer system performance has been suggested by Greenberger (2). This measure of performance involves the summation of the cost of delay to each job requiring service by the system. The measure of performance used in this study and in the previous study by Chai is based upon Greenberger's suggestion with a modification which makes it quite similar to a measure of performance used by Fife (10). Fife considered that the relative response, that is, the ratio of the response time to the amount of processing time required, was a quantity more fundamental than the actual response time to the measurement of performance.

The study of the performance of computer systems for a constant job stream and various job-scheduling algorithms can lead to the choice of a job-scheduling algorithm for a particular computer system and for the job stream considered. However, once selection of a computer system, or configuration, and a job-scheduling algorithm have been made, the job load generally grows rapidly until the system becomes heavily loaded. It is therefore often wise to select a job-scheduling algorithm which functions well under heavy loading, and it is an advantage to study the performance of an algorithm over a range of different job loads.

Many modern computers are modular in the sense that more main storage can be attached upon request, providing some upper limit is not exceeded. The size of main storage can be considered as a variable within a computer system to be selected according to some economic criterion, generally.

It is reasonable to suppose that, if a computer user is paying to receive a certain maximum relative response either directly, or indirectly by offering a certain proportion of a monthly computing capacity, for example, and that if this maximum relative response is exceeded, a discount amounting to a cost of delay to the user will be given. In this way the measure of performance would be related to the earning power of the computer system.

The decision whether or not to acquire more main storage in order to improve service can be related to economics by studying the improvement in performance as main storage is added to a computer system and balancing this improvement with the extra cost of acquiring the main storage. In this study no assumptions are made concerning the latter cost and the measure of performance is not related specifically to economics, but the effect upon system performance as the size of the main storage is varied is examined.

1.3 Previous Work in the Area of This Research

Most of the published research concerning the performance of time sharing systems has dealt with the performance of scheduling algorithms, and in particular, with the logic used for allocating execution times to jobs for which sufficient storage is already available. This logic is known as a time-slicing algorithm, or quantum allocation routine.

The primary reference for this study is the thesis of Chai (1) in which the computer system simulation model used in this study is introduced and described, in which a variable time-slice quantum allocation routine is presented, and in which the dynamic job penalty is used as a means of measuring user service. The Monte-Carlo techniques for job stream construction used in this study were also used in the previous study of Chai. In his study both central processor time requests and interarrival times were assumed to be normally distributed, while in this study arrival times are assumed to follow a Poisson distribution.

Kleinrock (3) presents an analytical study of time-shared computer systems in which facilities are treated as stochastic queuing systems under priority service disciplines. The performance measurement of these systems in taken to be the response time expected by the job under consideration. The results presented are for an ideal system. The Priority Processor-shared system analysed includes several priority classes with Poisson arrivals and exponentially distributed service requirements, with a known mean service requirement for jobs of each priority class. For this system Kleinrock states and proves a theorem which relates expected response time to mean central processor utilization in a fashion very similar to the empirical relation developed in the studies reported here. Other job scheduling algorithms are analysed, with theorems relating performance to utilization being stated and proved, by classical queuing theory, for each algorithm.

Schrage (4) presents an analytical study of the queuing and servicing discipline M/G/1 with feedback. This is a round-robin type of discipline. His performance measurement is the expected response time of the job under consideration. Arrival times are taken to be Poisson and execution times are taken to be exponentially distributed. Analytical relationships are given for the expected response time as a function of job arrival rate and job processing time. A graph is presented showing expected response time as a function of central processor utilization for various job processing times. For certain values of the job processing time, the graph is similar to some graphs presented by Kleinrock (3).

Penny (8) studies the effects of time-shared and non-time-shared computer facilities. He discusses improvements to be made in work load processing times that can be made by time and space sharing. The analysis produces a range of improvements as a function of processor utilization. Results obtained by simulation are shown to be in the range predicted, for the time sharing of two, three and four jobs.

Huesmann and Goldberg (6) present a survey article describing research in time-slicing algorithms with particular emphasis on Scherr's work (13) and the LOMUSS system (5,6) at Lockheed Corporation (Lockheed Multipurpose Simulation System). Huesmann and Goldberg make several very interesting statements. They suggest, for a successful simulation, that the details of the job-stream must be specified formally, that the constraints of the operation must be specified formally, and that the characteristics for judging system performance must be clearly defined. This is all in complete accord with the philosophy used by Chai and continued in this research. Further, Huesmann stresses the need for 'parameterization' of input to a simulation run and suggests that the simulation approach to time sharing system analysis is popular because there is lack of a viable alternative.

Some details of the LOMUSS system presented by Huesmann and Goldberg are important to this study. The LOMUSS system permits simulation of varying computer systems, or configurations with varying job streams.

For each simulation run two types of output are produced: the state of each resource at different points in time, and what is called an overall profile of each job, from which response times, processor idle time, memory utilization, throughput, and queue behavior may be determined. The specific nature of the scheduling algorithms used is not known, and no attempt has been made to relate user service to resource utilization - at least none has been published.

Nielsen (7) describes simulation studies made of an IBM 360/67 time sharing computer. He suggests that analytic studies (9, 13) are relatively inflexible and simulation studies allow more scope. The model described by Nielsen is responsive to changes in configuration, to changes in the scheduling and memory allocation algorithms, and to changes in the job stream. Memory is allocated by Nielsen in interchangeable sections rather than contiguously. Nielsen measures performance of his simulation model in terms of job response by priority and type and in terms of central processor utilization and equipment activity - in principle very simularly to the methods used in this research. He considers paging, that is, the rolling in and out of sections of jobs as required (17). Consideration was given to reserving storage for emergency, or heavily loaded operational conditions and found that any benefits to users were absorbed by the resultant increased idle time. Nielsen concluded that reducing the amount of paging, or rolling jobs in and out was the way to reduce overheads and improve user service. This conclusion may not be pertinent to these studies since paging is not used, but the intention of the statement - to reduce supervisor overheads-is one which is considered in this research.

Fife (10) examined the optimization of user service by using Markov model techniques in his study of the time slicing algorithm (quantum allocation routine.) The model be used included the complete rolling of jobs into and out of storage as required to have only the job being processed instantaneously in core. Three queues were used: the first, with jobs awaiting their first quantum of execution time; the second, with jobs awaiting their second quantum; and the third, with jobs awaiting their third