AD 739 268

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
Springfield, Va. 22151

Security Classification

# DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)


1. ORIGINATING ACTIVITY (Corporate author)

Carnegie-Mellon University
Department of Computer Science
Pittsburgh, Pennsylvania 15213


2a. REPORT SECURITY CLASSIFICATION

UNCLASSIFIED

2b. GROUP

3. REPORT TITLE

ON THE NEAR-COMPLETE-DECOMPOSABILITY OF NETWORKS OF QUEUES AND OF STOCHASTIC MODELS OF MULTIPROGRAMMING COMPUTING SYSTEMS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Scientific Interim


5. AUTHOR(S) (First name, middle initial, last name)

P. J. Courtois



6. REPORT DATE

November 1971

7a. TOTAL NO. OF PAGES

129

7b. NO. OF REFS

49

8a. CONTRACT OR GRANT NO.

F44620-70-C-0107

b. PROJECT NO.

AO 827

c. 61102F

d. 681304

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

AFOSR - TR - 72 - 0632


10. DISTRIBUTION STATEMENT

Approved for public release;
distribution unlimited.

11. SUPPLEMENTARY NOTES

TECH, OTHER

12. SPONSORING MILITARY ACTIVITY

Air Force Office of Scientific Research
1400 Wilson Boulevard (NM)
Arlington, Virginia 22209

13. ABSTRACT


Sufficient conditions under which a closed network of interconnected queues is nearly completely decomposable [Si61], [An63], are defined in terms of the resource service rates and the probabilities of transfer between queues. It is shown that when such conditions hold, the network may be organized as a hierarchy of 'aggregate resources', the equilibrium equations of which may be obtained separately as those of a finite single server queuing system. The interest of this approach in the analysis of multiqueue systems is discussed.

This approach is used to define and evaluate performance criteria for multiprogramming storage hierarchies which are shown to be nearly completely decomposable systems. We discuss also the relation between the concept of aggregate resource and the concept of abstract machine proposed by E. W. Dijkstra to structure the software of multiprogramming operating systems [Di69/1].

Finally, the use of an aggregative model is illustrated by the queuing analysis of a given paging time-sharing computing system. This analysis reveals that the regime of operations in such a system may be either stable or unstable, owing to the fluctuations of processor efficiency with the degree of multiprogramming. This notion of instability leads to a more complete definition of the circumstances under which thrashing may occur.


DD FORM 1 NOV 65 1473

Security Classification

CMU-CS-72-111

# ON THE NEAR-COMPLETE-DECOMPOSABILITY OF NETWORKS OF QUEUES AND OF STOCHASTIC MODELS OF MULTIPROGRAMMING COMPUTING SYSTEMS*

P. J. Courtois

November 1971

D D C
RECEIVED
MAR 27 1972
B

*This work was supported in part by the Manufacture Belge de Lampes et de Materiel Electronique S.A., Brussels, and in part by the Advanced Research Projects Agency of the Office of the Secretary of Defense (F44620-70-C-0107) (monitored by the Air Force Office of Scientific Research).

Approved for public release;
distribution unlimited.

ACKNOWLEDGMENTS

I am indebted to Professor H. A. Simon, Carnegie-Mellon University, for several conversations on the concept of near-decomposability and for his many helpful suggestions which contributed to the improvement of different parts of this paper.

I am also indebted to Professor D. L. Parnas, Carnegie-Mellon University, for several discussions and valuable suggestions, particularly with regard to the examination in section 6.2 of the relation between the concept of resource aggregation and that of level of abstraction in multiprogramming computer systems.

Mr. J. Georges, MBLE Research Laboratory, deserves my special gratitude. A great part of this research is the continuation of the work we did earlier together [Cou70], [Cou71]. The discussions I have had with him and his useful suggestions concerning this paper are numerous. He also wrote the computer program by which the numerical results discussed in the last section have been obtained.

I want also to thank Mrs. D. Josephson for her diligence in typing the manuscript.

CONTENTS

Acknowledgments

Introduction and Survey

I. Nearly Completely Decomposable Systems

1.1 The Simon-Ando Theorems

1.2 Interpretation of the Theorems

1.3 Fused States

1.4 Multi-level Systems

1.5 Cohesiveness

1.6 Remark

II. A Stochastic Model for Multiprogramming Systems

III. Near-Complete-Decomposability of Multiprogramming Stochastic Models

IV. A Hierarchy of Aggregate Resources

4.1 Decomposition into Levels

4.2 Inter-level Relationship

4.3 Concluding Remark

V. Closed Multi-queues System Analysis

VI. Near-Complete-Decomposability in Computing Systems

6.1 Storage Hierarchies

6.1.1 Computations

6.1.2 Single Process Storage Hierarchy

6.1.3 Multiprocess Storage Hierarchy

6.1.4 Near-Complete-Decomposability

6.1.5 Memory Level Aggregation

6.1.6 Dynamic Space Sharing

6.1.7 Remark

6.2 Hierarchical Structure of Multiprogramming Computer Operating Systems

6.2.1 Levels of Abstraction

6.2.2 Aggregation and Ordering of Abstractions

VII. Short- and Long-run Equilibria in a Time-Sharing Paging Multiprogramming System

7.1 The Hypothetical System

7.2 The User Programs

7.3 Simplifying Assumptions

7.4 Actualizing the Hypothetical System

7.5 The Page Demand Rate

7.6 The Rate of Page Transfer Completions

7.7 Conditions for Aggregation

7.8 Analysis of the Aggregates $\mathfrak{M}_1(J)$, $\mathfrak{M}^*(J_{max})$

7.8.1 Optimal Maximum Degree of Multiprogramming

7.8.2 Thrashing

7.8.3 Processor Efficiency versus Execution Intervals

7.9 The Long-Run Equilibrium

7.10 The System Congestion. Avalanche-like Effect

7.11 The System Saturation Points

7.12 The Congestion Stability

7.13 The System Response Time

7.14 Long-Run and Unconditional Distributions

Figures of Chapter VII

References

# INTRODUCTION AND SURVEY

"The concept of hierarchic order occupies a central place in this book, and lest the reader should think that I am riding a private hobby-horse, let me reassure him that this concept has a long and respectable ancestry. So much so, that defenders of orthodoxy are inclined to dismiss it as 'old hat' - and often in the same breath to deny its validity. Yet I hope to show as we go along that this old hat, handled with some affection, can produce lively rabbits."

A. Koestler
(The Ghost in the Machine)

It is in economic theory that aggregation of variables has been most explicitly used as a technique to study and evaluate the dynamics of systems of great size and complexity. This technique is founded on the idea that in many large systems all variables can somehow be clustered into a small number of groups so that: (i) the interactions among the variables of each single group may be studied as if interactions among groups did not exist and (ii) interactions among groups may be studied without reference to the interactions within groups. This is trivially correct when variables are functions of the values of variables of the same group but not of the values of any variable in any different group. The system in this case can be said *completely decomposable:** it truly consists of independent subsystems, each one of which can be analyzed separately, without reference to the others.

---

*It is worth making the terminology more precise at the outset: such a system may be represented by a *completely decomposable* matrix, i.e., a square matrix such that an identical permutation of rows and columns leaves a set of square submatrices on the principal diagonal and zeros everywhere else; a *decomposable* matrix, as opposed to completely decomposable, is a matrix with zeros everywhere below the principal submatrices but not necessarily also above. *Near-complete-decomposability* and *near-decomposability* are defined by replacing the zeros in these definitions by small non-zero numbers.

H. A. Simon and A. Ando [Si61] investigated circumstances under which variable aggregation still yields satisfactory approximations when the variables of a group do depend on the values of the variables of the other groups, but only weakly compared with intra-group dependency. Several examples taken from economics [Si61], physics [Si61] [Si62][Si69], and social sciences [Fi62/1], indicate that systems of that kind are likely to be more frequently encountered in reality than systems verifying the assumption of complete decomposability. The authors of [Si61] show that in these systems, qualified in [An63] as *Nearly Completely Decomposable Systems*, aggregation of variables separates the analysis of the short-run from that of the long-run dynamics. They proved two theorems. The first one states that, provided inter-group dependencies are sufficiently weak as compared to intra-group ones, in the short-run the system will behave approximately, and can therefore be analyzed, as if it were completely decomposable; whatever standard of approximation is required, there will always exist a non-zero degree of weakness of dependency such that the analysis will meet this standard of approximation. The second theorem states that even in the long-run, when neglected inter-group dependencies have had time to influence the system behavior, the values of the variables within any group will remain approximately in the same ratio as if those inter-group influences had never existed. The results obtained in the short-run will therefore remain approximately valid in the long-run, as far as the *relative* behavior of the variables of a same group is concerned.

These two theorems are formally introduced in the first section in the case of stochastic systems; our approach and our notation only deviate in a few details from [Si61]. Furthermore, we indicate how aggregative

variables representing *fused* states [Ve69] may be used at an arbitrary number of levels of aggregation to evaluate the limiting equilibrium probability distribution of stochastic systems with a large number of states.

In the following four sections we exploit the concept of near-complete-decomposability to analyze stochastic networks of interconnected queues. In Section II we define a model of a network which is a particular case of W. J. Gordon's and G. F. Newell's model [Go67]. An arbitrary fixed number of customers make use of an arbitrary fixed number (L+1) of resources, each of which provides a certain type of service. When service is completed by resource $\ell$, a customer proceeds directly to resource m with probability $p_{\ell m}$. The conditions under which such a system is nearly completely decomposable into L levels of aggregation are established in Section III in terms of the resource service rates and the transfer probabilities $p_{\ell m}$. We show in Section IV how, when these conditions are fulfilled, the set of resources may be organized in a hierarchy of *aggregate resources*, each aggregate resource being analyzable merely as a single server queuing system.

This approach presents several advantages which are discussed in Section V. First, the determination of the equilibrium marginal probabilities does not require, as in [Go67], the inversion of the transfer matrix $\|p_{\ell m}\|$. Explicit and closed form solutions are obtained for these probabilities, as well as for the average time a customer waits at each resource. These expressions facilitate the estimation of the equilibrium probabilities when the number of states is large, i.e., in the case of a

large number of customers and/or resources. Next, these solutions apply to queuing networks which are more general than the model considered in [ Go67]; they may be used to evaluate networks in which, as in the general model of J. R. Jackson [Ja63], the service rates are at each service stage almost arbitrary functions of the congestion at this stage; transfer probabilities dependent on the congestions at the stage of departure may be taken into account as well. Finally, this hierarchical model of aggregate resources appears to be a good approximation to multi-queue systems in which, instead of being exponentially distributed, the service times are random variables with arbitrary distribution functions; in other words, near-complete decomposability can under suitable circumstances dispense with the Poissonian service times assumption, which is classically made in networks of queues by sheer necessity of overcoming analytic difficulties.

Networks of interconnected queues are in several respects adequate models for studying congestion problems in multiprogramming computing systems [Wa66] [Sm66] [Ar71] [Bu71]. We show in Section VI that the characteristics inherent in memory hierarchies are of such a nature as to make near-complete decomposability an intrinsic property of their models. Hence, considering a computer memory as a hierarchy of *aggregate memory levels*, we use the model set up in Section IV to define and evaluate performance criteria for multiprogramming computer memory hierarchies. We indicate also how its ability to cope rather simply with state-dependent transfer probabilities may render this model sensitive to allocation policies which adjust dynamically the space allotted per program in each memory level to the demand made by the programs in the course of their execution.

In the second part of this same section, we discuss the similitude existing between the hierarchical model of aggregate resources defined in section IV and the structure of levels of abstraction advocated by E. W. Dijkstra for the software of multiprogramming computer systems [Di69/1]. We conjecture that the criteria that indicate what resources to aggregate are, in many cases, the same as those which specify what the levels of abstraction should be. The conditions for aggregation in such systems are expressed in terms of parameters closely related to the physical characteristics and the usage of the hardware resources. Hence, provided these parameters may be assessed, resource aggregation conditions might help the designer choose the ordering of the levels of abstraction. Variable aggregation appears in these two models as a technique that enables the level-by-level evaluation of the system to be interlaced with its level-by-level design. This stems from the fact that the conditions for aggregation are the same as those necessary to provide sufficiently approximate knowledge of the performances of a still incomplete system on which further design decisions may be based. Aggregative models of queue networks are, in the sphere of computing system analytical models, a counterpart of the level-by-level simulation techniques recommended in [Pa67] [Pa69] [Zu68] [Ra69].

The last section is intended to illustrate the use of an aggregative model in analyzing a given computing system. This concrete case study reveals that aggregation is not only adequate in obtaining numerical results when the number of parameters involved is large, but also helps to gain insight and conceptual clarity on the respective parts played by

these parameters. We study the model of a time-sharing system in which memory space and processor time are respectively allocated on a demand paging and a multiprogramming basis. We first carry out the analysis of the short-run equilibrium attained by the page traffic between primary and secondary memory. Several interesting phenomena are then revealed by relating these results to the long-run behavior of the flow of transactions between the set of user consoles and the entire computing system. We show that, owing to the fluctuations of processor usage with the degree of multiprogramming, the working conditions of a paging time-sharing system may be either *stable* or *unstable*. When the regime of operations is stable, the fluctuations of the congestion, measured by the current number of pending user transactions, damp out as a result of the alterations they cause to the ratio of the transaction input rate to the processor usage factor; in the unstable regime, these congestion fluctuations are reinforced by the way they alter the input rate to processor usage ratio. We indicate how these stable and unstable values of the congestion may be calculated from a set of parameters defining the computing load on the system and the page traffic between primary and secondary memory. Moreover, the analysis reveals that below a certain computing load the average congestion and the mean system response time increase less than linearly with the load, while beyond this computing load they increase more than linearly. This critical computing load is an extension to service systems with congestion-dependent service rate of the *saturation* point defined in [Kl68]. These notions of instability and saturation lead to a more complete definition of the circumstances under which the severe system performance degradation known as thrashing [De68/2] may occur.

# I. NEARLY COMPLETELY DECOMPOSABLE SYSTEMS

We introduce in this first section two theorems on nearly completely decomposable systems proved by H. A. Simon and A. Ando. We limit ourselves to the case of stochastic systems, which is the only case relevant to our study. Except for the subscripts, we have kept the same notation as in [Si61]. We show afterwards how aggregative variables may be used to represent *fused* states [Ve69] and to evaluate the limiting state probability distribution in stochastic multi-level nearly completely decomposable systems.

## 1.1 The Simon-Ando Theorems

We are interested in stochastic models of the form

$$x(t+1) = x(t)Q \tag{1.1}$$

where $x(t)$ is a row probability vector and $Q$ a stochastic matrix. The system having n possible state, $x_\ell(t)$ is the unconditional probability of the system being in the $\ell^{th}$ state at time t; $q_{k\ell}$ is the conditional probability that the system is in the $\ell^{th}$ state at time (t+1) given that it is in state k at time t.

We consider next a matrix $Q^*$, that can be arranged in the following form after an appropriate identical permutation of rows and columns:

$$Q^* = \left\| \begin{matrix} Q_1^* & & & \\ & \ddots & & \\ & & Q_I^* & \\ & & & \ddots \\ & & & & Q_N^* \end{matrix} \right\|$$

where the $Q_I^*$'s are square matrices and the remaining elements, not displayed, are all zero. Let $n_I$ be the order of $Q_I^*$; then $n = \sum_{I=1}^{N} n_I$. The following notation is adopted to refer to the elements of the vector $x^*(t)$:

$$x^*(t) = \{x_k^*(t)\} = \{[x_{i_1}^*(t)],\ldots,[x_{i_I}^*(t)],\ldots,[x_{i_N}^*(t)]\}$$

where $[x_{i_I}^*(t)]$ is a vector of elements of $\{x_k^*(t)\}$ so that if

$$x_{i_I}^*(t) = x_k^*(t),$$

then

$$k = \sum_{J=1}^{I-1} n_J + i, \qquad i=1,\ldots,n_I; \quad I=1,\ldots,N.$$

The matrix $Q^*$ is said to be *completely decomposable*. It is clear that in the system

$$x^*(t) = x^*(0)\, Q^{*t}$$

the subset $[x_{i_I}^*(t)]$ depends, for any t, only on $[x_{i_I}^*(0)]$ and $Q_I^*$, and is independent from $[x_{i_J}^*(t)]$ and $Q_J^*$, $J \neq I$.

Let us now consider the slightly altered matrix Q defined by

$$Q = Q^* + \epsilon\, C \tag{1.2}$$

where C is a matrix of the same size as $Q^*$ which has the property of keeping both Q and $Q^*$ stochastic (i.e., elements of C are at most equal to unity in absolute value and their rowsums amount to zero), and where $\epsilon$ is a very small real number to be specified later in §1.5. Matrices of the form of Q are defined in [An63] as being *nearly completely decomposable matrices*.

The two theorems to be introduced apply to the dynamic behavior of systems defined by (1.1) with Q defined by (1.2). Let the roots of the submatrix $Q_I^*$ be designated by $\lambda_{i_I}$, $i=1,\ldots,n_I$. We assume that these roots are distinct and so arranged that $\lambda_{1_I}^* > \lambda_{2_I}^* > \ldots > \lambda_{n_I I}^*$. $Q^*$ being completely decomposable and stochastic, we have

$$\lambda_{1_I}^* = 1, \quad I=1,2,\ldots,N.$$

Let us also define $\delta^*$ as being the minimum of the absolute values of the differences among all roots of $Q^*$ whose values are not unity and of their differences from unity; we have in particular:

$$|1-\lambda_{i_I}^*| \geq \delta^*, \quad i=2,\ldots,n_I; \; I=1,\ldots,N. \tag{1.3}$$

The roots of a matrix being continuous functions of its elements (see [Web98] §44, or [Bo07], e.g.), we can define for any positive real number $\delta$, however small, a small enough $\epsilon$ so that, for every root of $Q^*$, $\lambda_{i_I}^*$, there exists a root of Q, $\lambda_{i_I}$, such that

$$|\lambda_{i_I} - \lambda_{i_I}^*| < \delta, \quad i=1,\ldots,n_I; \; I=1,\ldots,N. \tag{1.4}$$

Hence, we may classify the roots of Q so that

$$\begin{aligned} &|1-\lambda_{1_I}| < \delta, && I=1,\ldots,N, \\ &|1-\lambda_{i_I}| > \delta^*-\delta, && i=2,\ldots,n_I; \; I=1,\ldots,N, \end{aligned} \tag{1.5}$$

where $\delta$ approaches zero with $\epsilon$.

Now, assuming all roots of Q to be distinct, the Sylvester expansion of Q into itself gives (see e.g., [Fr52] page 78):

$$Q = \sum_{I=1}^{N} \sum_{i=1}^{n_I} \lambda_{i_I} Z(\lambda_{i_I}) \tag{1.6}$$

where $$Z(\lambda_{i_I}) = \prod_{\substack{j,J \\ j_J \neq i_I}} (Q-\lambda_{j_J} I_n) / \prod_{\substack{j,J \\ j_J \neq i_I}} (\lambda_{i_I} - \lambda_{j_J}), \quad \begin{matrix} j=1,\ldots,n_J; \\ J=1,\ldots,N. \end{matrix}$$

The matrices $Z(\lambda_{i_I})$ have the following properties (see e.g., [Wed34] page 25);

$$[Z(\lambda_{i_I})]^2 = Z(\lambda_{i_I}),$$

$$Z(\lambda_{i_I}) \times Z(\lambda_{j_J}) = O_n, \quad i_I \neq j_J,$$

$$\sum_{I=1}^{N} \sum_{i=1}^{n_I} Z(\lambda_{i_I}) = I_n,$$

where $O_n$ is an $n \times n$ matrix of zeros and $I_n$ the $n \times n$ unit matrix.

On the other hand Q may also be written

$$\sum_{I=1}^{N} \sum_{i=1}^{n_I} \lambda_{i_I} \upsilon(i_I) \tilde{\nu}(i_I),$$

where the symbol ~ denotes the transpose and the column vectors $\upsilon(i_I)$ and $\nu(i_I)$ are respectively the left and right normalized eigenvectors of Q associated with $\lambda_{i_I}$ (see [Ta60], e.g.). So that

$$Z_{k,\ell}(\lambda_{i_I}) = \upsilon_k(i_I)\, \nu_\ell(i_I), \quad 1 \le k,\ \ell \le n,$$

and $$\sum_{k=1}^{n} Z_{k,k}(\lambda_{i_I}) = 1, \quad i=1,\ldots,n_I.$$

Moreover as Q is stochastic, $\upsilon_\ell(1_1)=1$ $(\ell=1,\ldots,n)$ and $\tilde{\nu}(1_1)$ is the limiting probability vector $\lim_{t\to\infty} x(t)$.

It follows from (1.6) and from the properties of idempotency and orthogonality that

$$Q^t = \sum_{I=1}^{N} \sum_{i=1}^{n_I} \lambda_{i_I}^t \; Z(\lambda_{i_I}). \tag{1.7}$$

Following the classification of roots defined by (1.5), we may divide the righthand side of (1.7) into three terms:

$$Q^t = Z(\lambda_{1_1}) + \sum_{I=2}^{N} \lambda_{1_I}^t \; Z(\lambda_{1_I}) + \sum_{I=1}^{N} \sum_{i=2}^{n_I} \lambda_{i_I}^t \; Z(\lambda_{i_I}). \tag{1.8}$$

$Q^*$ may not be expanded directly as Q since the N largest roots of $Q^*$ are all unity. However, any non decomposable submatrix $Q_I^*$ of $Q^*$ may be so expanded:

$$Q_I^{*t} = Z^*(\lambda_{1_I}^*) + \sum_{i=2}^{n_I} \lambda_{i_I}^{*t} \; Z^*(\lambda_{i_I}^*).$$

Bordering each matrix $Q_I^*$ with the appropriate number of rows and columns of zeros and designating those $n \times n$ matrices by the same name, we have

$$Q^* = \sum_{I=1}^{N} Q_I^*$$

and thereby $$Q^{*t} = \sum_{I=1}^{N} Z^*(\lambda_{1_I}^*) + \sum_{I=1}^{N} \sum_{i=2}^{n_I} \lambda_{i_I}^{*t} \; Z^*(\lambda_{i_I}^*). \tag{1.9}$$

The dynamic time behaviors of $x(t)$ and $x^*(t)$ are specified respectively by (1.8) and (1.9). To compare those behaviors, Simon and Ando [Si61] have proven the following propositions:

*Theorem 1.1. For an arbitrary positive real number* $\xi_2$, *there exists a number* $\epsilon_2$ *such that for* $\epsilon < \epsilon_2$,

$$\max_{k,\ell} \left| z_{k\ell}(\lambda_{i_I}) - z^*_{k\ell}(\lambda^*_{i_I}) \right| < \xi_2$$

*for* $i=2,\ldots,n_I$; $I=1,\ldots,N$; $1 \le k,\ \ell \le n$.

And

*Theorem 1.2. For an arbitrary positive real number* $\omega$, *there exists a number* $\epsilon_\omega$ *such that for* $\epsilon < \epsilon_\omega$,

$$\max_{i,j} \left| z_{i_I j_J}(\lambda_{1_K}) - \bar{\nu}^*_{j_J}(1_J) \quad \alpha_{IJ}(\lambda_{1_K}) \right| < \omega$$

*for* $K=1,\ldots,N$; $I=1,\ldots,N$; $J=1,\ldots,N$; $1 \le i \le n_I$; $1 \le j \le n_J$; *where* $[\bar{\nu}^*_{j_J}(1_J)]$ *is the right row eigenvector of* $Q^*_J$ *associated with the root of unity*

$$\bar{\nu}^*_{j_J}(1_J) = \frac{\nu^*_{j_J}(1_J)}{\sum_{j=1}^{n_J} \nu^*_{j_J}(1_J)}, \qquad j=1,\ldots,n_J,$$

*and where* $\alpha_{IJ}(\lambda_{1_K})$ *is given by*

$$\alpha_{IJ}(\lambda_{1_K}) = \sum_{i=1}^{n_I} \sum_{j=1}^{n_J} \bar{\nu}^*_{i_I}(1_I)\, z_{i_I j_J}(\lambda_{1_K}).$$

## 1.2 Interpretation of the Theorems

The implications of these two theorems may be discussed in more concrete terms.

Since by (1.5), the $\lambda_{1_I}$, $I=1,\ldots,N$ are almost unity, for any small $t, t < T_2$, $\lambda_{1_I}^t$, $I=1,\ldots,N$ will stay close to unity. Thereby, the first two terms of the righthand side of (1.8) will not vary very much while the first term of the righthand side of (1.9) will not vary at all. Thus, for $t < T_2$, the time behavior of $x(t)$ and $x^*(t)$ are defined by the last terms of (1.8) and (1.9), respectively. But, as $\epsilon \to 0$, it results from (1.4) that

$$\lambda_{i_I} \to \lambda^*_{i_I}$$

and from theorem 1.1 that

$$Z(\lambda_{i_I}) \to Z^*(\lambda_{i_I})$$

$$\text{for } i=2,\ldots,n_I \text{ and } I=1,\ldots,N.$$

Hence, for $t < T_2$, the time path of $x(t)$ must be very close to that of $x^*(t)$.

Now, since the $\lambda^*_{i_I}$, $i=2,\ldots,n_I$; $I=1,\ldots,N$, are less than unity (1.3), for any positive real $\xi_1$, we can define $T_1^*$ such that

$$\max_{\substack{k,\ell \\ 1 \le k,\ \ell \le n}} \left| \sum_{I=1}^{N} \sum_{i=2}^{n_I} \lambda_{i_I}^{*t} z^*_{k\ell}(\lambda^*_{i_I}) \right| < \xi_1, \text{ for } t > T_1^*.$$

Likewise, we can define $T_1$ such that

$$\max_{\substack{k,\ell \\ 1 \le k,\ \ell \le n}} \left| \sum_{I=1}^{N} \sum_{i=2}^{n_I} \lambda_{i_I}^{t} z_{k\ell}(\lambda_{i_I}) \right| < \xi_1, \text{ for } t > T_1.$$

Moreover, theorem 1.1 and (1.4) ensure that

$$T_1 \to T_1^* \text{ as } \epsilon \to 0,$$

$T_1^*$ being independent of $\epsilon$. Since $T_2$ can be made as large as we want by taking $\epsilon$ sufficiently small, while $T_1^*$ remains independent of $\epsilon$, *we shall take* $\epsilon$ *so that* $T_2$ *is very much larger than* $T_1$.

Finally, provided that $\epsilon$ is not identically zero so that, except for $\lambda_{1_1}$, $\lambda_{1_I}$, $I=2,\ldots,N$, is not identically unity, we may define $T_3$ so that for an arbitrary positive real number $\xi_3$ and $t > T_3$

$$\max_{\substack{k,\ell \\ 1 \le k,\ \ell \le n}} \left| \sum_{I=2}^{N} \lambda_{1_I}^t z_{k\ell}(\lambda_{1_I}) \right| < \xi_3.$$

Owing to the classification of roots defined above, $T_3$ is greater than $T_2$; it increases without limit as $\epsilon \to 0$. For $T_2 < t < T_3$, the last summation term of (1.8) is negligible and the time path of x is determined by the two first summation terms. But theorem 1.2 specifies that for any I and J, the elements of $Z(\lambda_{1_K})$

$$z_{i_I 1_J}(\lambda_{1_K}),\ldots,z_{i_I j_J}(\lambda_{1_K}),\ldots,z_{i_I n_J J}(\lambda_{1_K})$$

depend only upon I,J and j, and are almost independent of i. That is, for any I and J they are proportional to the elements of the characteristic vector of $Q_J^*$ associated with the root of unity

$$\bar{\nu}_{1_J}^*(1_J),\ldots,\bar{\nu}_{j_J}^*(1_J),\ldots,\bar{\nu}_{n_J J}^*(1_J) \tag{1.10}$$

and are approximately the same for $i=1,\ldots,n_I$.

Thereby, since for $T_2 < t < T_3$ $Q^t$ is mainly determined by the two first terms of (1.8), the vector $[x_{j_J}(t)]$ will vary with t during that period keeping for a given J an approximately constant *ratio* among distinct elements which is identical to that of the elements of (1.10). In other words a local equilibrium is reached within each subsystem J. We may at this stage use such an equilibrium probability distribution as (1.10) to cluster all states of a subsystem I, that is to replace the n-dimensional vector x(t) by an N-dimensional vector of aggregative variables $x_I(t)$, $I=1,\ldots,N$, and the $n \times n$ matrix Q by a $N \times N$ matrix $||q_{IJ}||$.

Finally, for $t > T_3$, all terms of the righthand side of (1.8) except the first one become negligible and the behavior of x(t) will be dominated by the largest root of Q, as in any linear dynamic system.

We may summarize the above discussion by saying that the dynamic behavior of a system representable by a nearly completely decomposable matrix may be analyzed in four stages called respectively by Simon and Ando: (i) short-run dynamics, (ii) short-run equilibrium, (iii) long-run dynamics, (iv) long-run equilibrium. In more precise terms, these stages are:

(i) Short-run dynamics: $t < T_1 < T_2$; the preponderantly varying term of (1.8) is the last one and this term is close to the last one of (1.9); x(t) and $x^*(t)$ evolve similarly.

(ii) Short-run equilibrium: $T_1 < t < T_2$; the last terms of (1.8) and (1.9) have vanished while the time powers of the N largest roots $\lambda_{1_I}^t$, $I=1,\ldots,N$, remain close to unity. A similar equilibrium is being reached within each subsystem of x(t) and $x^*(t)$.

(iii) Long-run dynamics: $T_2 < t < T_3$; the preponderantly varying term of (1.8) is the second one. The whole nearly completely decomposable system moves toward equilibrium, an equilibrium among relative values of variables within each subsystem being approximately maintained.

(iv) Long-run equilibrium: $t > T_3$; the first term of (1.8) dominates all the others. A global equilibrium is attained since in this case $\lambda_{1_1} = 1$.

*We conclude that in the short run, i.e., stages (i) and (ii), a system which enjoys the property of near-complete-decomposability may be considered as a set of independent subsystems which may approximately be analyzed separately from one another. In the long run the whole system appears to evolve keeping roughly the state of equilibrium within each subsystem. Each subsystem may be replaced by an aggregative variable, and the whole system analyzed as a set of interactions among those variables, the interactions within each subsystem being ignored.*

## 1.3 Fused States

In principle, the aggregative variables $x_I(t)$, $I=1,\ldots,N$, can be any function of the equilibrium state distribution $[\bar{\nu}^*_{i_I}(1_I)]$ of the system $Q^*_I$ provided that appropriate transition probabilities $q_{IJ}$ may be derived accordingly. A straightforward approach which has been discussed from another point of view in the literature (see e.g., [Ve69]), is that of "fusing" the states of each subsystem I. In this case $x_I(t)$ is taken as the sum of the probabilities of being in any one state $i_I$, $i=1,\ldots,n_I$ of subsystem $Q_I$ at time t:

$$x_I(t) = \sum_{i=1}^{n_I} x_{i_I}(t);$$

then, assuming the system defined by (1.1) to be at time t in any one state $i_I$, $i=1,\ldots,n_I$ of subsystem I, the probability that it will be in any one state $j_J$, $j=1,\ldots,n_J$ of subsystem J at time (t+1) is given by

$$q_{IJ}(t) = (x_I(t))^{-1} \sum_{i=1}^{n_I} x_{i_I}(t) \sum_{j=1}^{n_J} q_{i_I j_J}.$$

For $T_2 < t < T_3$, we may write

$$\frac{x_{i_I}(t)}{x_I(t)} = \bar{\nu}^*_{i_I}(1_I)$$

and thereby analyze the whole system as a set of subsystems $I=1,\ldots,N$ whose interactions among subsystems are *independent of time* and given by $||q_{IJ}||$ where

$$q_{IJ} = \sum_{i=1}^{n_I} \bar{\nu}^*_{i_I}(1_I) \sum_{j=1}^{n_J} q_{i_I j_J}. \tag{1.11}$$

Since, at this stage, only the roots $\lambda_{1_K}$, $K=1,\ldots,N$, are to be taken into account, we have

$$q_{i_I j_J} = \sum_{K=1}^{N} \lambda_{1_K} z_{i_I j_J}(\lambda_{1_K}). \tag{1.12}$$

Introducing (1.12) into (1.11) we obtain

$$q_{IJ} = \sum_{K=1}^{N} \lambda_{1_K} \sum_{i=1}^{n_I} \sum_{j=1}^{n_J} \bar{\nu}^*_{i_I}(1_I)\, z_{i_I j_J}(\lambda_{1_K})$$

which yields the relation between elements $q_{IJ}$ and the constants $\alpha_{IJ}(\lambda_1)$ which were introduced by theorem 1.2:

$$q_{IJ} = \sum_{K=1}^{N} \lambda_{1_K} \alpha_{IJ} (\lambda_{1_K}).$$

Let $x_I$, $I=1,\ldots,N$, be the long-run equilibrium probability of being at some time $t > T_3$ in subsystem I which is obtained by studying the whole system as a set of subsystems whose interactions are defined by (1.11). From those probabilities one can approximate the long-run equilibrium probabilities of being in any one elementary state $i_I$ by:

$$x_{i_I} = x_I \quad \bar{\nu}^*_{i_I}(1_I), \tag{1.13}$$

with

$$\sum_{i=1}^{n_I} x_{i_I} = x_I,$$

and

$$\sum_{I=1}^{N} x_I = 1, \qquad i=1,\ldots,n_I;\ I=1,\ldots,N.$$

## 1.4 Multi-level Systems

The system of variables discussed in the foregoing paragraph can be represented as a two-level hierarchy with the aggregative variables at the higher level. It is clear, and Simon and Ando do not fail to observe it, that such an hierarchy may be extended to more than two levels, each variable at a certain level being an aggregate of variables of the immediately lower level. An example of a two level nearly decomposable matrix (corresponding to a three-level hierarchy) would be

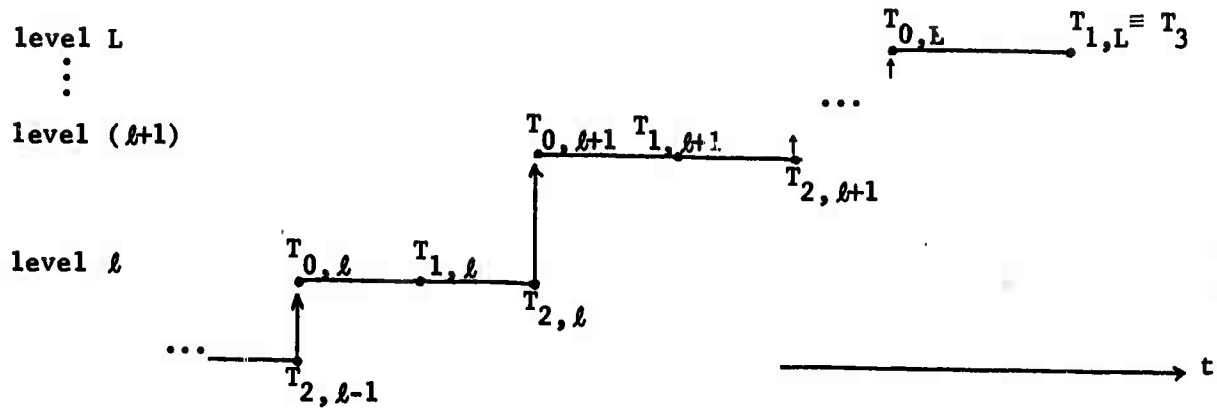
$$\left\| \begin{array}{cc|cc} P_1 & Q_1 & S_1 & S_2 \\ Q_2 & P_2 & S_3 & S_4 \\ \hline S_5 & S_6 & P_3 & Q_3 \\ S_7 & S_8 & Q_4 & P_4 \end{array} \right\|$$

where the order of magnitude of the elements of submatrices P is larger than that of matrices Q, this latter being larger than that of matrices S. At the first level of aggregation there could be four aggregative variables, one for each matrix P; at the second level of aggregation two aggregative variables would correspond to the partitions indicated by the dotted lines.

More generally, we will say that to the (L+1) levels $\ell=0,\ldots,L$ of a nearly completely decomposable hierarchy may correspond L levels of aggregation $\ell=1,\ldots,L$ and that such a hierarchy is representable by an L-*level nearly completely decomposable matrix.* In such a system the time instant $T_{2,\ell}$ beyond which the whole system ceases behaving as a set of nearly independent subsystems at the $\ell^{th}$ level of aggregation is also the time instant $T_{0,\ell+1}$ at which subsystems of the adjacent upper level ($\ell$+1) of aggregation start moving towards their internal equilibrium (which they reach at time $T_{1,\ell+1}$):

$$T_{2,\ell} \equiv T_{0,\ell+1}, \qquad \ell=1,\ldots,L-1$$

The time instant $T_3$ at which a global equilibrium is reached throughout the whole system is the time instant $T_{1,L}$ at which equilibrium is reached among the aggregative variables in terms of which the whole system is described at the uppermost level L. These time relations among levels are schematized in the following diagram:




Each level $\ell$ is a level of aggregation provided there exists an $\epsilon_\ell$ small enough so that

$$T_{1,\ell-1} < (T_{2,\ell-1} \equiv T_{0,\ell}) \quad \text{for } \ell=2,\ldots,L.$$

The analysis of the entire structure proceeds then from the lowest level of aggregation through each adjacent level up to the highest one: at each level of aggregation the system is viewed as a set of independent subsystems whose short-run equilibrium is analyzed in terms of aggregative variables descriptive of the short-run equilibria reached by the subcomponents which were considered at the adjacent lower level. The equilibrium obtained ultimately in this way at the most upper level defines *in terms of aggregative variables* the long-run equilibrium of the whole system.

From this long-run equilibrium at the uppermost level it is possible to deduce the long-run equilibrium probabilities for each lower level. In the case of "state fusing" discussed above, we may for instance generalize the relation (1.13) as follows. Let $x_I^{(L)*}$ be the long-run equilibrium probability of being in some aggregate subsystem I of the upper level L at any time t, $t > T_3$. Then the long-run equilibrium probability $x_J^{(L-1)}$ of being in subsystem J of I at the (L-1)th level of aggregation is given by

$$x_J^{(L-1)} = x_I^{(L)} \quad \bar{\nu}_{J_I}^{*(L-1)} \; (1_I). \tag{1.14}$$

Using recurrently this relation successively with L, L-1,...,1 and for all subsystems at each level, yields ultimately the long run equilibrium probability $x_i^{(0)}$ of being in the elementary state i, for all i.

## 1.5 Cohesiveness

It is easy to show [Fi62/2] that the largest characteristic root $\lambda_{1_I}$ of an $n_I \times n_I$ non negative matrix $Q_I$ is equal to

$$\lambda_{1_I} = \frac{\sum_{i=1}^{n_I} x_{i_I} S_{i_I}}{\sum_{i=1}^{n_I} x_{i_I}} , \tag{1.15}$$

where $S_{i_I}$ designates the ith row sum of $Q_I$ and $\{x_{i_I} \geq 0\}_{i=1}^{n_I}$ is the characteristic vector associated with $\lambda_{1_I}$, the strict inequality holding true if and only if $Q_I$ is not decomposable. Now, if $Q_I$ is a principal submatrix of a stochastic matrix Q, $S_{i_I}$ may be interpreted as the probability, when in state $i_I$, of remaining in the subsystem defined by $Q_I$ after the next transition; moreover, the distribution of the probabilities of being in state $i_I$, $i=1,\ldots,n_I$, at time t on condition to be in $Q_I$ at time t=0, approaches the distribution of the $x_{i_I}$ as $t \to \infty$. Thus, the weighted average (1.15) is the long-run probability, having been in subsystem $Q_I$ during the interval [0,t], of remaining in $Q_I$ at time (t+1), $t \to \infty$. Following [Fi62/1] we will refer to the probability $\lambda_{1_I}$ as being the *cohesiveness* of the subsystem $Q_I$.

One can deduce from (1.15) that a necessary and sufficient condition for Q to be nearly completely decomposable into the indecomposable matrices $Q_I$'s is that the roots $\lambda_{1_I}$, $I=1,\ldots,N$ be sufficiently close to unity. This condition is necessary to have the two first terms of (1.8) sufficiently close to the first term of (1.9). It is a sufficient condition since if the $x_{i_I}$ are strictly greater than zero, (1.15) requires that the $S_{i_I}$, just as $\lambda_{1_I}$, be close to unity, with the result that

$$\epsilon < 1 - \max_{i_I}(S_{i_I})$$

will be sufficiently small to satisfy the conditions of theorems 1.1 and 1.2.

Now, the Frobenius theorem (see e.g. [Ma64], page 152) states that

$$s_I \leq \lambda_{1_I} \leq S_I$$

where $s_I = \min_i s_{i_I}$ and $S_I = \max_i S_{i_I}$. Thus, $s_I$ is a lower bound of the cohesiveness of $Q_I$; a sufficient condition of the near-complete-decomposability of Q is thereby that $s_I$ be sufficiently close to unity for all subsystems $Q_I$, or that

$$\min_{\substack{I \\ I=1,\ldots,N}}(s_I) \gg 1-\min_I(s_I). \tag{1.16}$$

Relation (1.16), merely expressed in terms of the entries of a matrix, will be used in section III as a criterion of the near-complete-decomposability of matrices defining the dynamic behavior of networks of queues.

## 1.6 Remark

Ando and Fischer [An63] proved that the Simon-Ando theorems could be extended to the case of *nearly decomposable* matrices, defined by replacing the zeros in a decomposable matrix by small numbers. They showed that, mutatis mutandis, the conclusions of paragraph 1.2 remain true for such systems: the short-run dynamics may be analyzed as if the system were decomposable, thus ignoring the weak 'feedbacks' between subsystems; the long-run influence of these feedbacks may be analyzed in terms of aggregative variables representative of the short-run equilibrium attained by each subsystem. The subsystems which are aggregated and which, in the Simon-Ando case, are associated with the submatrices $P_I^*$, are in this case those corresponding to submatrices

$$\left\| \begin{matrix} P_I^* & G_I^* & & & \\ 0 & P_{I+1}^* & G_{I+1}^* & & \\ \cdot & \cdot & \cdot & \cdot & \\ 0 & \cdot & \cdot & 0 & P_N^* \end{matrix} \right\|$$

where the $G_I^*$ are submatrices of dimensions $(n_I, N - \sum_{J \leq I} n_J)$. Such systems are referred to in [An63] as *nearly decomposable block triangular* systems.

## II. A STOCHASTIC MODEL FOR MULTIPROGRAMMING SYSTEMS

We consider a system which consists of a set of independent resources $R_0, R_1, \ldots, R_L$. Each of those resources (*) is capable of providing a different type of service. We define a *sequential process* (*) in this system as being the sequential execution of an ordered sequence of requests for those resources, every single request applying to a single resource, every resource completing one request at a time.

Let the service time of resource $R_\ell$ be a random variable exponentially distributed with a mean value equal to $1/\mu_\ell$; $\mu_\ell$ measures the average number of requests which may be completed by $R_\ell$ per time unit.

A sequential process is stochastically defined by a set of transfer probabilities $p_{m\ell}$, $0 \le m, \ell \le L$; $p_{m\ell}$ is the probability of a request for resource $R_\ell$ following immediately the completion of a request for resource $R_m$ in the same sequential process. We have

$$\sum_{\ell=0}^{L} p_{m\ell} = 1, \qquad m=0,\ldots,L.$$

The fact that $p_{mm}$ may be nonzero means that a sequential process that has just completed service at resource $R_m$ may need to be reserviced immediately by the same resource.

(*) In queuing theory, the terms server and customer would be common.

Let N, $N < \infty$, be the total number of sequential processes concurrently executed in the system, these sequential processes being mutually independent and identically defined by the same probability set $\{p_{m\ell}\}$. As resources may at the most service one request at a time, sequential processes will eventually queue up for resources. Let $i_\ell$ be the number of sequential processes in service or in queue at resource $R_\ell$. Since sequential processes may at most be waiting for one resource at a time, we have

$$\sum_{\ell=0}^{L} i_\ell = N. \tag{2.0}$$

N is supposed to remain constant in our system. That is, each process is to be considered as lasting for an infinite period of time. This is not so important a restriction as might appear at first glance. We could indeed easily construct an equivalent system in which the total number of processes $\sum_{\ell=0}^{L} i_\ell$ would vary in time without exceeding N by simply considering a resource $R_{L+1}$ with service rate

$$\mu_{L+1} = \lambda \times i_{L+1} \tag{2.01}$$

and where

$$i_{L+1} = N - \sum_{\ell=0}^{L} i_\ell,$$

which would model the creation of processes in our system. The probability of a request to $R_\ell$ being the last request of a sequential process, i.e., the probability of a sequential process "dying" on completion of a request to $R_\ell$, would be $p_{\ell(L+1)}$; $p_{(L+1)\ell}$ would be the probability that $R_\ell$ is the first resource requested by a sequential process.

The state of the system is uniquely defined by the (L+1)-tuple $(i_0,\ldots,i_L)$ and there are $\binom{L+N}{L}$ distinguishable such states, i.e., the number of partitions of N processes among (L+1) sets.

Let $P(i_0,\ldots,i_L,t)$ be the joint probability that at epoch t the system is in state $(i_0,\ldots,i_L)$. These probabilities satisfy the system of difference equations of a generalized time-homogeneous birth and death process:

$$P(i_0,i_1,\ldots,i_L,\ t+h) =$$

$$P(i_0,i_1,\ldots,i_L,\ t) \quad (1-h \sum_{\ell=0}^{L} \varkappa(i_\ell)\ \mu_\ell(1-p_{\ell\ell}))$$

$$+ \sum_{\ell=0}^{L} \sum_{\substack{m=0 \\ m\neq \ell}}^{L} \varkappa(i_m)\ P(i_0,\ldots,i_\ell+1,\ldots,i_m-1,\ldots,i_L,t)h\mu_\ell\ p_{\ell m}$$

$$+\ 0(h), \tag{2.1}$$

where 0(h) is the probability, negligible for small values of h, of more than one request being completed during the time interval (t,t+h], and the binary function

$$\varkappa(i_\ell) = \begin{cases} 0 & \text{if } i_\ell = 0\ , \\ 1 & \text{if } i_\ell \neq 0\ , \end{cases}$$

accounts for the impossibility of any $i_\ell$ taking negative values.

This system of difference equations may be represented by the matrix equation

$$P(t+h) = P(t)\ (h\ A+I)\ \ + 0(h), \tag{2.2}$$

where P(t) is a vector whose each element is the probability for the system

being at time t in one of the $\binom{L+N}{L}$ distinguishable states. A is a matrix of constants called the transition intensity matrix of the process, with the properties that

$$\sum_j a_{ij} = 0$$

for all values of i, and

$$a_{ij} \geq 0$$

for all $i \neq j$. I is the matrix unity.

Equation (2.2) may be rewritten in the form

$$(P(t+h) - P(t))h^{-1} = P(t)\ A + \frac{0(h)}{h}\ .$$

As $h \to 0$, the last term tends to zero; hence the limit of the lefthand side exists and

$$P'(t) = P(t)A$$

where $P'(t)$ is the vector of the time derivatives of the elements of $P(t)$. An equilibrium probability distribution

$$P = \lim_{t \to \infty} P(t)$$

is, by definition, a solution of the system

$$P\ \ A = 0 \tag{2.3}$$

satisfying the condition that the sum of the elements of P equal 1. For a birth and death process, this limiting distribution always exists and is independent of the initial conditions P(0).

We shall not pursue the investigation of this equilibrium distribution which has been studied in [Ja63], [Go67]; we shall instead concentrate on the structure of the matrix A which defines the dynamic behavior of the model.

## III. NEAR-COMPLETE-DECOMPOSABILITY OF MULTIPROGRAMMING STOCHASTIC MODELS

We establish in this section the conditions under which the system defined by equations (2.1) enjoys the property of near-complete-decomposability.

Henceforth we will neglect $0(h)$, take $h=1$ and $Q=A+I$, so that equation (2.2) may be rewritten

$$P(t+1) = P(t)\, Q.$$

The time unit is chosen small enough so that $\sum_{\ell=0}^{L} \mu_\ell \leq 1$.

Each index value $j$, $j=1,\ldots,\binom{L+N}{L}$ of the stochastic matrix $Q$ refers to a distinct state $(i_0,i_1,\ldots,i_L)$ of the system. We choose an arrangement of the rows and columns of $Q$ such that the $(L+1)$-tuple $i_0,i_1,\ldots,i_L$ which is associated with the index value $j$, yields for the function

$$f(x) = i_0 + i_1 x + i_2 x^2 + \ldots + i_L x^L$$

a value $f_j(x)$ such that

$$f_j(x) > f_{j-1}(x) \text{ for } \forall j > 1 \text{ and } \forall x > N.$$

An example of a matrix whose rows and columns are arranged in this order is given in figure 3.1.

It results from equations (2.1) that any non diagonal entry $q_{jk}$ of $Q$, $j \neq k$, is non zero and equal to $\mu_\ell\, p_{\ell m}$, $\ell \neq m$, if and only if there exists a pair $(\ell,m)$ so that

$$\begin{aligned} i_{j\ell} &= i_{k\ell} + 1 \\ i_{jm} &= i_{km} - 1 \\ i_{jp} &= i_{kp} \quad \text{for } \forall p \neq \ell,m, \end{aligned} \tag{3.1}$$

where $i_{j\ell}$ is the value of the coefficient of $x^{\ell}$ in $f_j(x)$.

It results also from (2.1) that any diagonal entry $q_{jj}$, j referring to some state $(i_0,\ldots,i_\ell,\ldots,i_m,\ldots i_L)$ is given by

$$q_{jj} = 1 - \sum_{k\neq j} q_{jk} = 1 - \sum_{\ell=0}^{L} \sum_{\substack{m=0\\ m\neq \ell}}^{L} \varkappa(i_\ell)\, \mu_\ell\, p_{\ell m} = 1 - \sum_{\ell=0}^{L} \varkappa(i_\ell)\mu_\ell(1-p_{\ell\ell}) \tag{3.2}$$

Denoting Q(N,L), $N > 0$, $L > 1$, the matrix of order $\binom{L+N}{L}$ defined by equations (2.0), (2.1) and this numbering of states, we can prove the following

*Lemma 3.1. The entries of the stochastic matrix* Q(N,L), $N>0$, $L>1$, *may be partitioned among* (N+1) *principal submatrices* $Q'(i_L)$, $i_L = 0,\ldots,N$ *so that*

(i) *the set of all non diagonal entries of each matrix* $Q'(i_L)$, $i_L=n$, *is the set of all transition probabilities between any two distinct states of the* $\binom{L-1+N-n}{L-1}$ *states with* $i_L=n$;

(ii) *Any non zero entry outside the main diagonal of* Q(N,L) *takes only the value* $\mu_\ell p_{\ell m}$, $\ell\neq m$, *with* $\ell, m<L$ *if it is located within one of these submatrices, and with* $\max(\ell,m) = L$ *if it is located outside these submatrices.*

*Proof.* 1) Owing to the numbering of system states according to increasing values of f(x), the smaller index values of Q(N,L) refer to *all* states with $i_L=0$; there are $\binom{L-1+N}{L-1}$ such states, i.e., the number of distinct integer solutions of equation

$$\sum_{\ell=0}^{L-1} i_\ell = N.$$

Likewise, the $\binom{L+N-2}{L-1}$ following index values refer to all states with $i_L = 1$, and so on up to $i_L = N$. All states may therefore be partitioned among (N+1) sets, $i_L = 0,\ldots,N$, the $(i_L+1)$th grouping all the $\binom{L-1+N-i_L}{L-1}$ states for which

$$\sum_{\ell=0}^{L-1} i_\ell = N-i_L,$$

in agreement with the identity

$$\sum_{i_L=0}^{N} \binom{L-1+N-i_L}{L-1} = \binom{L+N}{L}$$

These (N+1) sets define as many principal submatrices denoted $Q'(i_L)$ all of whose non diagonal entries are the transition probabilities between any two states of the set of all states for which $i_L$ has a given value.

2) The second proposition of the lemma follows immediately. Any off diagonal entry $q_{jk}(N,L)$, $j\neq k$, of $Q(N,L)$ is non zero and equal to $\mu_\ell p_{\ell m}$, $0 \le \ell, m \le L$, provided there exist a pair $(\ell,m)$, $\ell\neq m$, which verifies equalities (3.1).

For any entry $q_{jk}(N,L)$, $j\neq k$, located within a submatrix $Q'(i_L)$, $i_{jL} = i_{kL} = i_L$; it is thereby necessary that

$$(\ell\neq L) \text{ and } (m\neq L)$$

for $q_{jk}(N,L)$ to be non zero.

Alternatively, for any entry $q_{jk}(N,L)$ $j\neq k$, located outside all the principal matrices $Q'(i_L)$, $i_L=0,\ldots,N$, by definition $i_{jL} \neq i_{kL}$; that is

$$(\ell=L) \text{ or } (m=L),$$

which completes the proof.

In order to help visualize the structure of matrix $Q(N,L)$, $Q(2,3)$ is displayed in figure 3.1. Plain lines isolate submatrices $Q'(0)$, $Q'(1)$, $Q'(2)$; $\Sigma_i$ denotes the ith rowsum of non diagonal entries.

Using the preceding lemma, we may now prove the following proposition:

*Lemma 3.2. If,*

$$\min_{\substack{i_0,\ldots,i_L \\ \sum_{k=0}^{L} i_k = N}} \left[ 1 - \sum_{k=0}^{L} \varkappa(i_k)\mu_k + \sum_{k=0}^{L-1} \varkappa(i_k)\mu_k \left( \sum_{m=0}^{L-1} p_{km} - p_{kL} \right) - \varkappa(i_L)\mu_L \left( \sum_{m=0}^{L-1} p_{Lm} - p_{LL} \right) \right] \gg 0, \quad (3.3)$$

*then the sparse stochastic matrix* $Q(N,L), N > 0,\ L > 1$, *defines a system nearly completely decomposable into* $(N+1)$ *systems which may be represented by stochastic matrices* $Q(N-n,L-1)$, $n=0,\ldots,N$.

*Proof.* (1) Lemma 3.1 allows to write:

$$Q(N,L) = Q^*(N,L) + \epsilon_L\, C(N,L) \quad (3.4)$$

with

$$Q^*(N,L) = \left\| \begin{matrix} Q'(0) & & & \\ & Q'(1) & & \\ & & \ddots & \\ & & & Q'(N) \end{matrix} \right\|$$

where all entries of $Q^*(N,L)$ not displayed are zero, non diagonal entries of $Q^*(N,L)$ have only values $\{0,\mu_k p_{km}\}$, $k \neq m$, with $k,m < L$, and non diagonal entries of $C(N,L)$ have only values $\{0,\epsilon^{-1}\mu_L p_{Lm}, \epsilon^{-1}\mu_m p_{mL}\}$ with $m < L$.

On the other hand, the jth rowsum of $Q(N,L)$, say $S_j$, j referring to some state $(i_0,\ldots,i_k,\ldots,i_m,\ldots,i_{L-1},i_L)$, may be expressed as

$$S_j = q_{jj} + \sum_{k=0}^{L-1} \sum_{\substack{m=0 \\ m\neq k}}^{L-1} \varkappa(i_k)\mu_k p_{km} + \sum_{k=0}^{L-1} \varkappa(i_k)\mu_k p_{kL} + \varkappa(i_L) \sum_{m=0}^{L-1} \mu_L p_{Lm}. \qquad (3.5)$$

It results from lemma 3.1 that the sum of the two first terms of $S_j$, is the sum $S'_j$ of the entries of the corresponding row in $Q'(i_L)$, $i_L=0,\ldots,N$. We know from § 1.5 that a sufficient condition for $Q(N,L)$ to be nearly completely decomposable into $Q'(0),\ldots,Q'(N)$, is that the lower bound of the *cohesiveness* of any of these submatrices be sufficiently close to unity; or that (cfr. inequality 1.16):

$$\min_j(S'_j) \gg 1 - \min_j(S'_j).$$

Since $S_j=1$, this condition is equivalent to

$$\min_j \left[ q_{jj} + \sum_{k=0}^{L-1} \sum_{\substack{m=0 \\ m\neq k}}^{L-1} \varkappa(i_k)\mu_k p_{km} \right] \gg \sum_{k=0}^{L-1} \varkappa(i_k)\mu_k p_{kL} + \varkappa(i_L) \sum_{m=0}^{L-1} \mu_L p_{Lm}.$$

Replacing $q_{jj}$ by its value given by relation 3.2, yields inequality (3.3).

Inequality (3.3) expresses the condition that the probability of an individual sequential process moving between the subsystem of resources $R_0,\ldots,R_{L-1}$ and the subsystem $R_L$ is small relative to the minimum probability of his remaining in the same subsystem or of no sequential process moving at all.

(2) All states referred by the index values of a matrix $Q'(n)$, $n=0,\ldots,N$, are arranged according to increasing values taken by the function $(i_0+\ldots+i_{L-1}x^{L-1} + nx^L)$ with $\sum_{\ell=0}^{L-1} i_\ell = N-n$. Hence, it results from proposition (i) of lemma 3.1 that each non diagonal entry of $Q'(n)$ is identical to the

corresponding entry of a matrix $Q(N-n,L-1)$. $Q'(n)$ is however not stochastic; its jth rowsum $S'_j$ is equal to the two first terms of $S_j$ in (3.5) and is therefore strictly smaller than unity. But the stronger the inequality (3.3), the more negligible will be the two last terms of (3.5) relative to the two first ones, so that

$$S'_j \to S_j \text{ with } S_j = 1,$$

and

$$Q'(n) \to Q(N-n,L-1),\ n=0,\ldots,N,$$

which completes the proof.

Applying recurrently lemma 3.2 on the matrices $Q(n,\ell)$, $n > 0$, $\ell > 1$, produced by the recurrence matrix relation (3.4), one obtains the following.

*Theorem 3.1: The system defined by matrix* $Q(N,L)$, $N > 0$, $L > 1$, *is* (L-1)-*level nearly completely decomposable and may be represented at each level of aggregation* $\ell$, $\ell=L-1,\ldots,1$, *by a set of* (N+1) *aggregation variables, each one defining the short-run equilibrium of a system* $Q(n_\ell,\ell)$, $n_\ell=0,\ldots,N$ *if, for* $\ell=L-1,\ldots,1$:

$$\min_{\substack{i_0,\ldots,i_{\ell+1} \\ 1 \le \sum_{k=0}^{\ell+1} i_k \le N}} \left[1 - \sum_{k=0}^{\ell+1} \varkappa(i_k)\mu_k + \sum_{k=0}^{\ell} \varkappa(i_k)\mu_k\Big(\sum_{m=0}^{\ell} p_{km} - p_{k(\ell+1)}\Big) - \varkappa(i_{\ell+1})\mu_{\ell+1}\Big(\sum_{m=0}^{\ell} p_{(\ell+1)m} - p_{(L+1)(L+1)}\Big)\right] \gg 0. \tag{3.5.1}$$

A sufficient condition for near-complete-decomposability which is stronger than the above condition but also easier to verify is given by

*Corollary 3.1: A sufficient condition for* $Q(N,L)$ *to be* (L-1)-*level nearly completely decomposable into systems* $Q(n_\ell,\ell)$, $n_\ell=0,\ldots,N$; $\ell=L-1,\ldots,1$, *is:*

$$\textit{for } \ell = L-1,\ldots,1: \quad \min_{0\le k\le \ell} [\mu_k (\sum_{m=0}^{\ell} p_{km} - p_{k(\ell+1)})] \gg \mu_{\ell+1} (\sum_{m=0}^{\ell} p_{(\ell+1)m} - p_{(\ell+1)(\ell+1)}). \quad (3.6)$$

This condition is obtained by ignoring in (3.5.1) the term $[1 - \sum_{k=0}^{\ell+1} \varkappa(i_k)\mu_k]$ which is always non-negative.

*Remarks.* (1) The inequalities specified in theorem 3.1 are sufficient but not necessary conditions for near-complete-decomposability since they refer to the lower bound of the subsystems' cohesiveness. On the other hand, let us recall that theorem 1.1 guarantees that whatever standard of approximation is required, a degree of inequality exists which is sufficient to produce results satisfying that standard.

(2) We will be more particularly concerned in the sections VI and VII with the special multi-queue model in which $\forall k$: $p_{kk} = 0$; for $k \neq 0$, $p_{k0} = 1$; and for k and $m \neq 0$, $p_{km} = 0$. This model is referred to in [Bu71] as the central server model. In this case conditions (3.6) reduce to:

$$\text{for } \ell = L-1,\ldots,1: \min_{0<k\le \ell} (\mu_k) \gg \mu_{\ell+1} \quad (3.7.1)$$

$$\text{and} \quad \mu_0 (\sum_{m=1}^{\ell} p_{0m} - p_{0(\ell+1)}) \gg \mu_{\ell+1}. \quad (3.7.2)$$

on the other hand, it is easy to verify that a sufficient (but

not necessary) condition for (3.7.2) to hold, is $p_{01} \gg p_{02} \gg \ldots \gg p_{0L}$.

(3) Each submatrix $Q'(i_L)$, $i_L=1,\ldots,N$ may itself be partitioned into principal submatrices $Q'(i_L,i_{L-1})$, $0 \le i_{L-1} \le N-i_L$, whose set of non diagonal entries is the set of all transition probabilities between any two distinct states of the set of all states which have the same pair $(i_{L-1},i_L)$. Following this scheme recurrently, $Q(N,L)$ may eventually be decomposed into principal submatrices $Q'(i_L,i_{L-1},\ldots,i_2)$. In figure 3.1 for example, dotted lines isolate submatrices $Q'(0,0)$, $Q'(0,2)$, $Q'(1,0)$, $Q'(1,1)$, $Q'(2,0)$. Let $x_{\ell+1,n}$ be the number of submatrices $Q'(i_L,\ldots,i_{L-\ell})$, $\ell=0,\ldots,L-2$; $x_{\ell+1,n}$ obeys the recurrence relation

$$x_{\ell+1,n} = x_{\ell,n} + x_{\ell+1,n-1}$$

where $n = \sum_{m=0}^{\ell} i_{L-m}$ and with the boundary conditions $x_{\ell+1,1} = \ell+1$, $x_{1,n} = 1$. Therefore $x_{\ell+1,n} = \binom{\ell+n}{\ell}$ and the total number of submatrices $Q'(i_L,\ldots,i_{L-\ell})$ is given by

$$\sum_{n=0}^{N} x_{\ell+1,n} = \binom{N+\ell+1}{N}.$$

Using identity (12.16) of [Fe68] page 65 one may verify that the sum of the orders of matrices $Q'(i_L,\ldots,i_{L-\ell})$ amounts to the order of $Q(N,L)$ or that

$$\sum_{n=0}^{N} \binom{\ell+n}{n} \binom{L-\ell-1+N-n}{N-n} = \binom{L+N}{L}.$$

(4) One can observe in figure 3.1 that by the mere virtue of the resource service rate inequalities, Q(N,L) is a nearly decomposable block triangular matrix if all transfer probabilities have approximately the same value.

| $i_0 i_1 i_2 i_3$ | 2 0 0 0 | 1 1 0 0 | 0 2 0 0 | 1 0 1 0 | 0 1 1 0 | 0 0 2 0 | 1 0 0 1 | 0 1 0 1 | 0 0 1 1 | 0 0 0 2 |
|---|---|---|---|---|---|---|---|---|---|---|
| 2 0 0 0 | $1-\Sigma_1$ | $\mu_0 P_{01}$ | 0 | $\mu_0 P_{02}$ | 0 | 0 | $\mu_0 P_{03}$ | 0 | 0 | 0 |
| 1 1 0 0 | $\mu_1 P_{10}$ | $1-\Sigma_2$ | $\mu_0 P_{01}$ | $\mu_1 P_{12}$ | $\mu_0 P_{02}$ | 0 | $\mu_1 P_{13}$ | $\mu_0 P_{03}$ | 0 | 0 |
| 0 2 0 0 | 0 | $\mu_1 P_{10}$ | $1-\Sigma_3$ | 0 | $\mu_1 P_{12}$ | 0 | 0 | $\mu_1 P_{13}$ | 0 | 0 |
| 1 0 1 0 | $\mu_2 P_{20}$ | $\mu_2 P_{21}$ | 0 | $1-\Sigma_4$ | $\mu_0 P_{01}$ | $\mu_0 P_{02}$ | $\mu_2 P_{23}$ | 0 | $\mu_0 P_{03}$ | 0 |
| 0 1 1 0 | 0 | $\mu_2 P_{20}$ | $\mu_2 P_{21}$ | $\mu_1 P_{10}$ | $1-\Sigma_5$ | $\mu_1 P_{12}$ | 0 | $\mu_2 P_{23}$ | $\mu_1 P_{13}$ | 0 |
| 0 0 2 0 | 0 | 0 | 0 | $\mu_2 P_{20}$ | $\mu_2 P_{21}$ | $1-\Sigma_6$ | 0 | 0 | $\mu_2 P_{23}$ | 0 |
| 1 0 0 1 | $\mu_3 P_{30}$ | $\mu_3 P_{32}$ | 0 | $\mu_3 P_{32}$ | 0 | 0 | $1-\Sigma_7$ | $\mu_0 P_{01}$ | $\mu_0 P_{02}$ | $\mu_0 P_{03}$ |
| 0 1 0 1 | 0 | $\mu_3 P_{30}$ | $\mu_3 P_{31}$ | 0 | $\mu_3 P_{32}$ | 0 | $\mu_1 P_{10}$ | $1-\Sigma_8$ | $\mu_1 P_{12}$ | $\mu_1 P_{13}$ |
| 0 0 1 1 | 0 | 0 | 0 | $\mu_3 P_{30}$ | $\mu_3 P_{31}$ | $\mu_3 P_{32}$ | $\mu_2 P_{20}$ | $\mu_2 P_{21}$ | $1-\Sigma_9$ | $\mu_2 P_{23}$ |
| 0 0 0 2 | 0 | 0 | 0 | 0 | 0 | 0 | $\mu_3 P_{30}$ | $\mu_3 P_{31}$ | $\mu_3 P_{32}$ | $1-\Sigma_{10}$ |

Q(2,3)

**Figure 3.1**

## IV. A HIERARCHY OF AGGREGATE RESOURCES

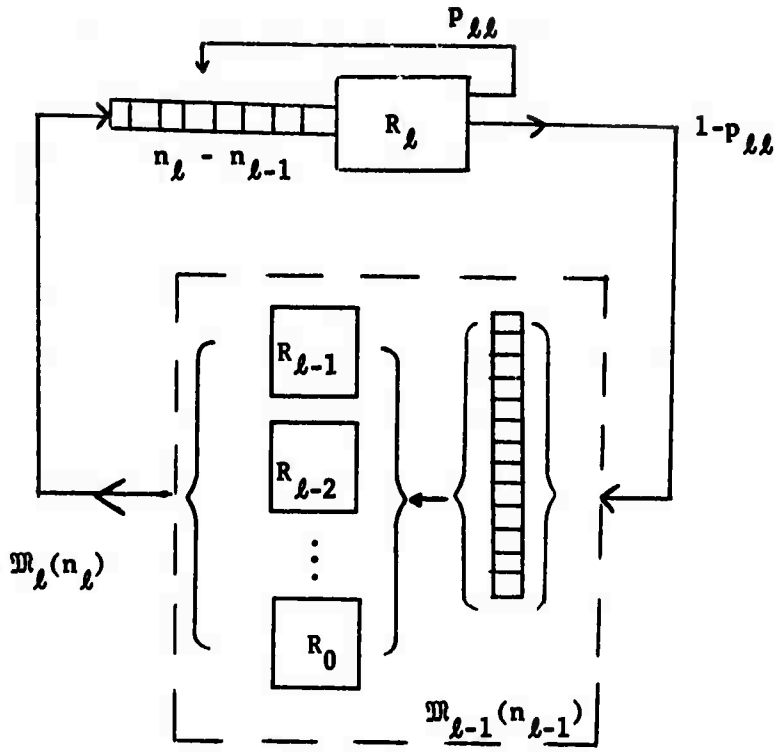
We assume henceforth that the stochastic multiprogramming model defined by matrix $Q(N,L)$, $N > 0$, $L > 1$, is a $(L-1)$ level nearly-completely-decomposable system. This amounts to postulating that there exists a numbering of the resources $R_0, R_1, \ldots, R_\ell, \ldots, R_L$ so that the service rates $\mu_\ell$ and the stochastic behavior $\{p_{ij}\}$, $0 \le i,j \le L$, of the sequential processes obey the $(L-1)$ inequalities of theorem 3.1.

These inequalities imply that L subsystems s, $s=1,\ldots,L$, consisting of the $(s+1)$ resources $R_0,\ldots,R_s$ may be distinguished so that, on the average, the rates at which resources of a same subsystem interact upon each other are higher than rates of interaction between any resource within that subsystem and any resource outside it. In other words, although the values of variables $i_0,\ldots,i_s$ defining the state of the sequential process population in any subsystem s depend on the past values of variables $i_{s+1},\ldots,i_L$ outside this subsystem, these dependencies are weak compared to intra-subsystem dependencies. In this case the first Ando-Simon theorem (theor. 1.1) guarantees the following: provided that inter-subsystem dependencies are weak compared to intra-subsystem dependencies, the former may be neglected when carrying out the analysis of each subsystem; the results of this analysis will remain approximately valid *in the short run*; the weaker inter-subsystems influences are, the better the degree of approximation. Now, as pointed out by M. Fisher and A. Ando [Fi62/1]:"...if this were all, it would be useful but not very remarkable for it would merely mean that if neglected influences are sufficiently weak they take a long time to matter much;...".

But the second Ando-Simon theorem asserts that, even after a time period long enough for inter-subsystem influences to make themselves felt, the *relative* values of variables within each subsystem will remain approximately the same as those yielded by the short-run analysis. This justifies an analysis of the long-run behavior of the system in its entirety carried out strictly in terms of aggregative variables representative of these intra-subsystem short-run equilibria and in terms of inter-subsystem dependencies. On the basis of those general concepts, the recurrent decomposition scheme defined by theorem 3.1 enables a hierarchical model of the multiprogramming system defined in section 2 to be set up. This model is approximately equivalent to Q(N,L), the degree of approximation depending on the relative degree of weakness of inter-subsystem dependencies. We now proceed to describe more precisely this "nearly equivalent" hierarchical model, which will turn out to be a generalization of the model described in [Cg70].

## 4.1 Decomposition into Levels

Any matrix $Q(n_\ell, \ell)$, $1 \le n_\ell \le N$, $1 \le \ell \le L$, defines a system in which $n_\ell$ sequential processes compete for the $(\ell+1)$ resources $R_0, R_1, \ldots, R_\ell$. Let us associate with each such matrix a queueing system which will be denoted $\mathfrak{M}_\ell(n_\ell)$ and which may be schematically represented as follows:




A fixed and finite population of $n_\ell$ sequential processes is considered to be cycling in this system. These sequential processes request alternatively two resources: resource $R_\ell$ and what we call an *aggregate resource* which will appear to be a synthesis of resources $R_0,\ldots,R_{\ell-1}$. We say that the system $\mathfrak{M}_\ell(n_\ell)$ is in state $E_\ell(n_{\ell-1}|n_\ell)$ whenever $n_{\ell-1}$ among $n_\ell$ sequential processes are either waiting for or being executed by that aggregate resource, i.e., whenever $n_{\ell-1}$ among $n_\ell$ sequential processes are either waiting for or being executed by *any one* of the resources $R_{\ell-1},\ldots,R_0$, and $(n_\ell - n_{\ell-1})$ sequential processes are either waiting for or being executed by the resource $R_\ell$.

Assuming Q(N,L) to be a (L-1) level nearly-completely-decomposable system, we may analyze the time behavior of $\mathfrak{M}_\ell(n_\ell)$ in the following way. We regard $\mathfrak{M}_\ell(n_\ell)$

as a closed system that no sequential process may leave or enter. In other words, we disregard all interactions between $\mathfrak{M}_\ell(n_\ell)$ on the one hand and $R_{\ell+1},\ldots,R_L$ on the other hand; we will see hereafter that the rates of those interactions are sufficiently low to be neglected in the analysis of the time behavior of $\mathfrak{M}_\ell(n_\ell)$. We further suppose that all conditions are fulfilled for the existence of a probabilistic equilibrium in $\mathfrak{M}_\ell(n_\ell)$. This means that it may be taken for granted that each state $E_\ell(n_{\ell-1}|n_\ell), n_{\ell-1}=0,\ldots,n_\ell$, may be reached from any other state $E_\ell(j\,|n_\ell), j\neq n_{\ell-1}$, within a finite time period (irreducibility and non-null recurrence).

These conditions are necessary for the stochastic matrix $Q(n_\ell,\ell)$ to have a unique root equal to unity and will in fact be fulfilled in most cases when resources have a non-zero and finite service rate since the population of $\mathfrak{M}_\ell(n_\ell)$ is finite. $\mathfrak{M}_\ell(n_\ell)$ thereby enjoys the property of ergodicity: there exists a stationary distribution of the probabilities $\pi_\ell(n_{\ell-1}|n_\ell)$ of finding $\mathfrak{M}_\ell(n_\ell)$ in state $E_\ell(n_{\ell-1}|n_\ell), n_{\ell-1}=0,\ldots,n_\ell$, which is independent of time and of the initial conditions of $\mathfrak{M}_\ell(n_\ell)$.

An immediate consequence of theorem 3.1 is that this probabilistic equilibrium of $\mathfrak{M}_\ell(n_\ell)$ which is defined by $Q(n_\ell,\ell)$ will establish itself by some time $T_{1,\ell}$ (see section 1.4), that is, before some time $T_{0,\ell+1}$, when resources $R_{\ell+1},\ldots,R_L$ have had time to influence $\mathfrak{M}_\ell(n_\ell)$. Interactions between those resources and $\mathfrak{M}_\ell(n_\ell)$ may therefore be ignored when studying the evolution of $\mathfrak{M}_\ell(n_\ell)$ towards its probabilistic equilibrium $\{\pi_\ell(n_{\ell-1}|n_\ell)\}_{n_{\ell-1}=0}^{n_\ell}$.

Similarly, just as $Q(n_\ell, \ell)$ is itself decomposable into subsystems $Q(n_{\ell-1}, \ell-1), n_{\ell-1}=0,\ldots,n_\ell$, $\mathfrak{M}_\ell(n_\ell)$ is nearly decomposable into subsystems $\mathfrak{M}_{\ell-1}(n_{\ell-1})$. By definition, these subsystems have attained a state of internal equilibrium $\{\pi_{\ell-1}(n_{\ell-2}|n_{\ell-1})\}_{n_{\ell-2}=0}^{n_{\ell-1}}$ at some time $T_{1,\ell-1}$, before time $T_{0,\ell}$ when $\mathfrak{M}_\ell(n_\ell)$ starts moving towards equilibrium; those internal equilibria are maintained during the period from $T_{0,\ell}$ to $T_{1,\ell}$ when $\mathfrak{M}_\ell(n_\ell)$ settles down to equilibrium. In the study of the time behavior of $\mathfrak{M}_\ell(n_\ell)$ during that period, we may therefore assume that the aggregate resource of $\mathfrak{M}_\ell(n_\ell)$, *which by virtue of our decomposition scheme is nothing but* $\mathfrak{M}_{\ell-1}(n_{\ell-1})$ *when* $\mathfrak{M}_\ell(n_\ell)$ *is in state* $E_\ell(n_{\ell-1}|n_\ell)$, is already in a state of equilibrium, conditioned on $n_{\ell-1}$ and defined by the distribution $\{\pi_{\ell-1}(n_{\ell-2}|n_{\ell-1})\}_{n_{\ell-2}=0}^{n_{\ell-1}}$.

More precisely, let $\sigma_\ell(n_\ell)$ be the mean number of requests completed per time unit in $\mathfrak{M}_\ell(n_\ell)$; this number is the sum of the mean number of requests completed by $R_\ell$ and the mean number of requests completed by the aggregate resource per time unit. As our decomposition scheme implies that the aggregate resource of $\mathfrak{M}_\ell(n_\ell)$ is $\mathfrak{M}_{\ell-1}(n_{\ell-1})$ whenever $\mathfrak{M}_\ell(n_\ell)$ is in state $E_\ell(n_{\ell-1}|n_\ell)$, we have

$$\sigma_\ell(n_\ell) = \mu_\ell(1-\pi_\ell(n_\ell|n_\ell)) + \sum_{n_{\ell-1}=1}^{n_\ell} \pi_\ell(n_{\ell-1}|n_\ell)\sigma_{\ell-1}(n_{\ell-1}). \tag{4.1}$$

$$(\ell=2,\ldots,L),(n_\ell=1,\ldots,N)$$

For $\ell=1$, (4.1) reduces to

$$\sigma_1(n_1) = \mu_1(1-\pi_1(n_1|n_1)) + \mu_0(1-\pi_1(0|n_1)), \quad (n_1=1,\ldots,N). \tag{4.2}$$

We will refer to $\sigma_\ell(n_\ell)$ as being the service rate of $\mathfrak{M}_\ell(n_\ell)$; clearly, requests to $R_0, R_1, \ldots, R_\ell$ are completed in $\mathfrak{M}_\ell(n_\ell)$, when $\mathfrak{M}_\ell(n_\ell)$ is in equilibrium, at a rate of $\sigma_\ell(n_\ell)$ requests per time unit; $\sigma_\ell(n_\ell)$ being deduced from the rates $\mu_0, \mu_1, \ldots, \mu_\ell$ and the equilibrium distributions $\{\pi_{\ell-m}(n_{\ell-m-1}|n_{\ell-m})\}_{n_{\ell-m-1}=0}^{n_{\ell-m}}$, for $n_{\ell-m}=1,\ldots,n_\ell$, and $m=0,\ldots,\ell-1$.

*Remark.* Relations (4.1) and (4.2) seem to imply that the constituents of $\mathfrak{M}_\ell(n_\ell)$, $R_\ell$ and the aggregate resource, work independently of each other, while in reality these resources interfere with each other. In fact $\sigma_\ell(n_\ell)$ is an aggregative variable describing only the equilibrium of $\mathfrak{M}_\ell(n_\ell)$ in which these interferences have reached a steady state defined by the distribution $\{\pi_\ell(n_{\ell-1}|n_\ell)\}_{n_{\ell-1}=0}^{n_\ell}$.

Let $[h \times \psi_{\ell,k}(n_\ell) + 0(h)]$ designate the probability that during any interval $(t,t+h]$, $t > T_{1,\ell}$, a request is completed by $\mathfrak{M}_\ell(n_\ell)$ and the sequential process being serviced applies to resource $R_k$, $k > \ell$. As $\sigma_\ell(n_\ell)$, $\psi_{\ell,k}(n_\ell)$ obeys a recurrence relation:

$$\psi_{\ell,k}(n_\ell) = \mu_\ell(1-\pi_\ell(n_\ell|n_\ell))p_{\ell k} + \sum_{n_{\ell-1}=1}^{n_\ell} \pi_\ell(n_{\ell-1}|n_\ell)\psi_{\ell-1,k}(n_{\ell-1}). \qquad (4.3)$$

$$(k > \ell),(\ell=2,\ldots,L-1),(n_\ell=1,\ldots,N);$$

For $\ell=1$, (4.3) reduces to

$$\psi_{1,k}(n_1) = \mu_1 p_{1k} \quad (1-\pi_1(n_1|n_1)) + \mu_0 p_{0k}(1-\pi_1(0|n_1)), \quad (k=2,\ldots,L)(n_1=1,\ldots,N) \quad (4.4)$$

$\psi_{\ell,k}(n_\ell)$ will be referred to as the interaction rate of $\mathfrak{M}_\ell(n_\ell)$ upon resource $R_k$, $k > \ell$.

Probabilities $\{\pi_1(n_0|n_1)\}_{n_0=0}^{n_1}, n_0=1,\ldots,N$ appearing in (4.2) and (4.4) may easily be expressed as functions of $n_0$ and $n_1$. They define the equilibrium state of $\mathfrak{M}_1(n_1)$ whose constituents are merely $R_0$ and $R_1$. The probability that a state transition $E_1(n_0|n_1) \to E_1(n_0+1|n_1), n_0=0,\ldots,n_1-1$, occurs during a time interval $(t,t+h]$, $t< T_{0,2}$, is equal to $(\mu_1 p_{10} h + 0(h))$ where $0(h)$ is negligible for small values of h. Likewise, the probability of a transition $E_1(n_0|n_1) \to E_1(n_0-1|n_1)$, $n_0=1,\ldots,n_1$, during a same time interval is equal to $(\mu_0 p_{01} h + 0(h))$. In such a system, equilibrium equations reduce to (see e.g., [Fe68], pp. 460 ff.):

$$\mu_1 p_{10}\, \pi_1(0|n_1) = \mu_0 p_{01}\, \pi_1(1|n_1),$$

$$(\mu_1 p_{10} + \mu_0 p_{01})\pi_1(n_0|n_1) = \mu_1 p_{10}\, \pi_1(n_0-1|n_1) + \mu_0 p_{01}\, \pi_1(n_0+1|n_1),$$

$$n_0=1,\ldots,n_1-1,$$

$$\mu_0 p_{01}\, \pi_1(n_1|n_1) = \mu_1 p_{10}\, \pi_1(n_1-1|n_1).$$

With the additional condition that the $\pi_1(n_0|n_1)$, $n_0=0,\ldots,n_1$, add to unity, these relations yield

$$\pi_1(n_0|n_1) = \left(\frac{\mu_1 p_{10}}{\mu_0 p_{01}}\right)^{n_0} \pi_1(0|n_1),\ n_0=1,\ldots,n_1 \tag{4.5}$$

with $$\pi_1(0|n_1) = \left\{1 + \sum_{n_0=1}^{n_1}\left(\frac{\mu_1 p_{10}}{\mu_0 p_{01}}\right)^{n_0}\right\}^{-1}. \tag{4.6}$$

Introducing (4.5) and (4.6) into (4.4) yields the values of $\psi_{1,k}(n_1), n_1=1,\ldots,N$; $k=2,\ldots,L$, given the service rates of $R_0$ and $R_1$ and the transfer probabilities $\{p_{0k}\}_{k=0}^{L}, \{p_{1k}\}_{k=0}^{L}$.

Using the values obtained for $\psi_{1,2}(n_1)$, $n_1=1,\ldots,N$, one may easily obtain the distributions $\{\pi_2(n_1|n_2)\}_{n_1=0}^{n_2}$ of the equilibria which will be attained by systems $\mathfrak{M}_2(n_2)$, $n_2=1,\ldots N$, at some time $T_{1,2}$.

The probability of a state transition $E_2(n_1|n_2) \to E_2(n_1-1|n_2)$, $n_1=n_2,\ldots,1$, occurring in $\mathfrak{M}_2(n_2)$ during any time interval $(t,t+h]$, $t < T_{0,3}$ is equal to $(\psi_{1,2}(n_1)h + 0(h))$ while the probability of a transition $E_2(n_1|n_2) \to E_2(n_1+1|n_2)$, $n_1=0,\ldots,n_2-1$, is equal to $[\mu_2(p_{20}+p_{21})h + 0(h)]$. The equilibrium equations are therefore:

$$[\psi_{1,2}(n_1) + \mu_2(p_{20}+p_{21})]\,\pi_2(n_1|n_2) = \pi_2(n_1-1|n_2)[\mu_2(p_{20}+p_{21})] + \pi_2(n_1+1|n_2)\,\psi_{1,2}(n_1+1),$$

$$(n_1=1,\ldots,n_2-1),$$

$$[\mu_2(p_{20}+p_{21})]\,\pi_2(0|n_2) = \pi_2(1|n_2)\,\psi_{1,2}(1),$$

$$\psi_{1,2}(n_2)\,\pi_2(n_2|n_2) = \pi_2(n_2-1|n_2)[\mu_2(p_{20}+p_{21})].$$

The additional condition that probabilities $\{\pi_2(n_1|n_2)\}_{n_1=0}^{n_2}$ add to unity yields

$$\pi_2(n_1|n_2) = \frac{[\mu_2(p_{20}+p_{21})]^{n_1}}{\prod_{k=1}^{n_1}\psi_{1,2}(k)}\,\pi_2(0|n_2),\quad n_1=1,\ldots,n_2$$

$$\text{with } \pi_2(0|n_2) = \left\{1 + \sum_{n_1=1}^{n_2}\frac{[\mu_2(p_{20}+p_{21}]^{n_1}}{\prod_{k=1}^{n_1}\psi_{1,2}(k)}\right\}^{-1}. \tag{4.7.1}$$

Introducing these probabilities into (4.3) yields the values of the interaction rates $\psi_{2,k}(n_2)$, $n_2=1,\ldots,N$; $k = 3,\ldots,L$, given the values of the rates $\psi_{1,k}(n_1)$, $n_1=1,\ldots,n_2$. Proceeding reourrently in this way up to level $\ell$, the probabilistic equilibrium of a system $\mathfrak{M}_\ell(n_\ell)$, $n_\ell=1,\ldots,N$, is obtained as

$$\pi_\ell(n_{\ell-1}|n_\ell)= \frac{(\mu_\ell \sum_{k=0}^{\ell-1} p_{\ell k})^{n_{\ell-1}}}{\prod_{k=1}^{n_{\ell-1}} \psi_{\ell-1,\ell}(k)} \pi_\ell(0|n_\ell)$$

$$\pi_\ell(0|n_\ell) = \left\{1 + \sum_{n_{\ell-1}=1}^{n_\ell} \frac{(\mu_\ell \sum_{k=0}^{\ell-1} p_{\ell k})^{n_{\ell-1}}}{\prod_{k=1}^{n_{\ell-1}} \psi_{\ell-1,\ell}(k)}\right\}^{-1} . \tag{4.7.2}$$

$$\ell=2,\ldots,L;\ n=1,\ldots,N.$$

Starting with the systems of the bottom level $\ell=1$, the probability distributions $\{\pi_\ell(n_{\ell-1}|n_\ell)\}_{n_{\ell-1}=0}^{n_\ell}$, for $n_\ell=1,\ldots,N$, are obtained from (4.7.1) and (4.7.2) successively for each level up to the uppermost one. At each level $\ell$ the values of the interaction rates $\psi_{\ell,k}(n_\ell)$, $k = \ell+1,\ldots,L$ necessary for the analysis of the levels above, must be deduced by (4.3) from the values of $\psi_{\ell-1,k}(n_{\ell-1})$.

## 4.2 Inter-level Relationship

So far, each subsystem $\mathfrak{M}_\ell(n_\ell)$, $\ell=1,\ldots,L-1$, has been analyzed as a closed system inaccessible to entry or exit from the upper levels. For this reason the equilibrium distributions obtained are conditioned on a

fixed number $n_\ell$ of sequential processes supposed to be permanently cycling in $\mathfrak{M}_\ell(n_\ell)$. These conditional internal equilibria are what Simon and Ando call the short run equilibria which are successively attained at each level of aggregation. We have seen in section 1.4 of Chapter I how the long-run equilibrium probabilities for each lower level of aggregation could be deduced from the long-run equilibrium at the uppermost level. If $a_\ell(n_{\ell-1}),(n_{\ell-1}=0,\ldots,N)$, $(\ell=1,\ldots,L)$, is the long-run and unconditional probability of $n_{\ell-1}$ sequential processes being in service or in queue at any one resource $R_m$, $m=0,\ldots,\ell-1$, and $s_\ell(i_\ell)$, $i_\ell=0,\ldots,N$ is the long-run and unconditional probability of $i_\ell$ sequential processes being in service or in queue at resource $R_\ell$, the relations (1.14) yield:

$$a_L(n_{L-1}) = \pi_L(n_{L-1}|N),\ s_L(i_L) = \pi_L(N-i_L|N),(n_{L-1},i_L=0,\ldots,N);$$

and for each level $\ell=L-1,\ldots,1$:

$$a_\ell(n_{\ell-1}) = \sum_{n_\ell=n_{\ell-1}}^{N} a_{\ell+1}(n_\ell)\pi_\ell(n_{\ell-1}|n_\ell), \tag{4.8}$$

$$s_\ell(i_\ell) = \sum_{n_\ell=i_\ell}^{N} a_{\ell+1}(n_\ell)\pi_\ell(n_\ell-i_\ell|n_\ell).$$

At level 1, whose only constituents are resources $R_1$ and $R_0$, we have

$$s_0(i_0) \equiv a_1(i_0), \quad i_0=0,\ldots,N.$$

These relations express simply that the probability of having a population of $n_\ell$ sequential processes cycling at level $\ell$ is equal to the probability

of being in one of the states $E_{\ell+1}(n_\ell|n_{\ell+1})$, $n_{\ell+1}=n_\ell,\ldots,N$ at level $(\ell+1)$.

The long-run fraction of time a resource $R_\ell$, $\ell=0,\ldots,L$ is busy is equal to $(1-s_\ell(0))$. The *mean response time* $W_\ell$ of resource $R_\ell$, defined as being the mean time spent by a sequential process in queue or the service at resource $R_\ell$ may be easily deduced using Little's formula [Li61]:

$$W_\ell = [\mu_\ell(1-s_\ell(0))]^{-1} \sum_{i_\ell=1}^{N} i_\ell s_\ell(i_\ell), \quad \ell=0,\ldots,L. \tag{4.9}$$

Moreover, if the uppermost level resource $R_L$ is the model of an input mechanism of sequential processes whose input rate $\mu_L$ obeys equation (2.01), one may deduce from the probabilities $\{a_L(n_{L-1})\}_{n_{L-1}=0}^{N}$, using Little's formula, the mean life time of a sequential process:

$$W = [\mu_L(1-s_L(0))]^{-1} \sum_{n_{L-1}=1}^{N} n_{L-1}a_L(n_{L-1}); \tag{4.10}$$

W is the mean time spent by a sequential process in queue or in service at the aggregate resource of level (L-1). We will refer to W as being the *mean response time of this aggregate of resources* $R_0,\ldots,R_{L-1}$.

## 4.3 Concluding Remark

Near-Complete-Decomposability allows the time analysis of a system to be broken up into distinct stages at each of which only a subspace of the system state space needs to be taken into consideration. Thanks to this property we were able to substitute the closed multiqueues system Q(N,L) whose state space is of size $\binom{N+L}{L}$, with L x N subsystems $\mathfrak{M}_\ell(n_\ell)$, $(\ell=1,\ldots,L;\ n_\ell=1,\ldots,N)$,

each having no more than $(n_\ell+1)$ distinct states. Each of these subsystems is an equivalent representation of the $x_{\ell,N-n_\ell}$ subsystems, $x_{\ell,N-n_\ell} = \binom{\ell-1+N-n_\ell}{\ell-1}$, defined by the $x_{\ell,N-n_\ell}$ principal submatrices $Q'(i_L,\ldots,i_{L-\ell+1})$ of $Q(N,L)$, with $\sum_{m=0}^{\ell-1} i_{L-m}=N-n_\ell$ (cfr. remark 3 at the end of section 3). An interesting consequence of this state space partitioning is that larger state spaces of more refined models than the one in section 2 may be analyzed. Examples are given in the following section.

## V. CLOSED MULTI-QUEUES SYSTEM ANALYSIS

In this section we discuss some of the advantages of the hierarchical approach of resource aggregation in the analysis of multi-queues systems enjoying the property of near-complete decomposability. We are introducing two generalizations of the model defined in section 2 which may be approached by this method.

5.1 In [Go67] W. J. Gordon and G. F. Newell solve very elegantly the system of equilibrium equations (2.3) by a separation of variables technique. They obtain an exact expression of the equilibrium joint probability $P(i_0,\ldots i_L)$. However, the form of this expression is such as to make the determination of the marginal probabilities tedious for large values of N or L. So, for the case of systems with non-uniform service-rates, assuming that one resource, say $R_L$, has an effective slower service rate than all the other resources, they show that, at the limit for $N \to \infty$, the distribution of sequential processes within the system is regulated by this slower resource; they are able to give in this case asymptotical expressions for the marginal probability of $i_\ell$ sequential processes in queue or in service at a resource $R_\ell$, $\ell=0,\ldots,L-1$. This method may in certain cases present two types of disadvantages: Firstly, the values obtained are only asymptotical values for $N \to \infty$, and secondly, the method requires the inversion of the $(L+1) \times (L+1)$ transfer probability matrix $\| p_{ij} \|$. Both disadvantages may be avoided if the service rates and the transfer probabilities obey the conditions of near-complete-decomposability of theorem 3.1 so that expressions (4.8) may be used to calculate the marginal probability distributions.

5.2 Equation (2.1) may be generalized so as to allow, as in [Ja63], the service rates of any resource $R_\ell$, $\ell=0,\ldots,L$, to be a function $\mu_\ell(i_\ell) \geq 0$ of the number $i_\ell$ of sequential processes currently in queue or in service at this resource ($\forall_\ell : \mu_\ell(0)=0$).

The conditions under which this model is nearly-completely-decomposable are obtained by substituting $\varkappa(i_\ell)\mu_\ell$ with $\mu(i_\ell)$ in the inequalities of theorem 3.1. Simpler but stronger conditions are given by corollary 3.1:

For $\ell=L-1,\ldots,1$:

$$\min_{k,i_k}[\mu_k(i_k)(\sum_{m=0}^{\ell} p_{km} - p_{k(\ell+1)})] \gg$$

$$\mu_{\ell+1}(i_{\ell+1})(\sum_{m=0}^{\ell} p_{(\ell+1)m} - p_{(\ell+1)(\ell+1)}). \tag{5.1}$$

where $0 \leq k \leq \ell$, $0 \leq i_{\ell+1} \leq N-1$, $1 \leq i_k \leq N-1_{\ell+1}$.

The interaction rate of the aggregate $\mathfrak{M}_\ell(n_\ell)$, $n_\ell=1,\ldots,N$, upon resource $R_k$, $k > \ell$, may then be obtained as

$$\psi_{\ell,k}(n_\ell) = \sum_{n_{\ell-1}=0}^{n_\ell-1} \pi_\ell(n_{\ell-1}|n_\ell)[p_{\ell k}\mu_\ell(n_\ell-n_{\ell-1}) + \sum_{n_{\ell-1}=1}^{n_\ell} \pi_\ell(n_{\ell-1}|n_\ell)\psi_{\ell-1,k}(n_{\ell-1}). \tag{5.2}$$

And the equilibrium equations of this aggregate are:

$$[\psi_{\ell-1,\ell}(n_{\ell-1})+\mu_\ell(n_\ell-n_{\ell-1})\times\sum_{k=0}^{\ell-1} p_{\ell k}]\pi_\ell(n_{\ell-1}|n_\ell)=\pi_\ell(n_{\ell-1}-1|n_\ell)[\mu_\ell(n_\ell-n_{\ell-1})\times\sum_{k=0}^{\ell-1} p_{\ell k}]$$

$$+\pi_\ell(n_{\ell-1}+1|n_\ell)\psi_{\ell-1,\ell}(n_{\ell-1}+1),$$

$$n_{\ell-1}=1,\ldots,n_\ell-1,$$

$$[\mu_\ell(n_\ell)\times\sum_{k=0}^{\ell-1} p_{\ell k}]\pi_\ell(0|n_\ell)=\pi_\ell(1|n_\ell)\psi_{\ell-1,\ell}(1),$$

$$[\psi_{\ell-1,\ell}(n_\ell)]\pi_\ell(n_\ell|n_\ell)=\pi_\ell(n_\ell-1|n_\ell)[\mu_\ell(1)\times\sum_{k=0}^{\ell-1} p_{\ell k}].$$

The additional condition that probabilities $\{\pi_\ell(n_{\ell-1}|n_\ell)\}_{n_{\ell-1}=0}^{n_\ell}$ add to unity yields

$$\pi_\ell(n_{\ell-1}|n_\ell)=\frac{[\sum_{k=0}^{\ell-1} p_{\ell k}]^{n_{\ell-1}}\prod_{k=0}^{n_{\ell-1}-1}\mu_\ell(n_\ell-k)}{\prod_{k=1}^{n_{\ell-1}}\psi_{\ell-1,\ell}(k)}\pi_\ell(0|n_\ell) \tag{5.3}$$

with
$$\pi_\ell(0|n_\ell)=\left\{1+\sum_{n_{\ell-1}=1}^{n_\ell}\frac{[\sum_{k=0}^{\ell-1} p_{\ell,k}]^{n_{\ell-1}}\prod_{k=0}^{n_{\ell-1}-1}\mu_\ell(n_\ell-k)}{\prod_{k=1}^{n_{\ell-1}}\psi_{\ell-1,\ell}(k)}\right\}^{-1}$$

which extend expressions (4.7.2). Using (4.8) one obtains the unconditional probabilities $s_\ell(i_\ell)$, $i_\ell=0,\ldots,N$, of $i_\ell$ sequential processes in service or in queue at any resource $R_\ell$, $\ell=0,\ldots,L$.

Applications of this generalization are mentioned at the end of [Cg70] and use is made of it later on in paragraph 7.9. The special case of this

generalization studied in [Go67] is the case in which $\mu_\ell(i_\ell)$ is given by

$$\mu_\ell(i_\ell) = \alpha_\ell(i_\ell)\mu_\ell, \quad \ell=0,\ldots,L,$$

$$\text{where } \alpha_\ell(i_\ell) = \begin{cases} i_\ell, & \text{if } i_\ell \le v_\ell, \\ v_\ell, & \text{if } i_\ell \ge v_\ell; \end{cases} \tag{5.4}$$

at each level of this system there are $v_\ell$ parallel exponential servers, each with mean service rate $\mu_\ell$. Sufficient conditions for inequalities (5.1) to be satisfied are then, for $\ell=L-1,\ldots,1$:

$$\min_{\substack{k \\ 0\le k\le \ell}}[\mu_k(\sum_{m=0}^{\ell} p_{km} - p_{k(\ell+1)})] \gg v_{\ell+1}\mu_{\ell+1}(\sum_{m=0}^{\ell} p_{(\ell+1)m} - p_{(\ell+1)(\ell+1)}). \tag{5.5}$$

Finally, by a very similar generalization, transfer probabilities $p_{\ell k}(i_\ell)$ depending on the congestion at the stage of departure may also be coped with. Conditions (5.1) as well as relations (5.2) and (5.3) are easily rewritten in this case. An application of state-dependent probabilities will be considered in §6.1.4.

5.3 The hierarchical model of aggregate resources may also approximate multiqueues closed systems in which, instead of being exponentially distributed, the service times of the upper level resources $R_\ell$, $\ell=L,L-1,\ldots$ are random variables identically and independently distributed with *arbitrary* distribution function, say $B_\ell(x)$. This conjecture is based on

the following argument. After time $T_{1,\ell}$, when aggregate resources $\mathfrak{M}_\ell(n_\ell)$, $n_\ell=1,\ldots N$, are in equilibrium, the time sequence of the occurrences of a transfer of a sequential process by $\mathfrak{M}_\ell(n_\ell)$ to a resource $R_k$, $k > \ell$, on a request completion, may be regarded as a stationary point process of intensity $\psi_{\ell,k}(n_\ell)$, (cf. [Kh55]). If we expand $\psi_{\ell,k}(n_\ell)$ in terms of the transfer rates $\mu_{\ell-j} \times p_{(\ell-j)k}$, $j=0,\ldots\ell$, we have:

$$\psi_{\ell,k}(n_\ell) = \sum_{j=0}^{\ell} A_\ell(\ell-j;\ n_\ell)[1 - \pi_{\ell-j}(n_{\ell-j}|n_{\ell-j})]\mu_{\ell-j}p_{(\ell-j)k}$$

where $\pi_0(n_0|n_0) = 0$, $A_\ell(\ell;\ n_\ell) = 1$; and for $j=1,\ldots,\ell$,

$$A_\ell(\ell-j;\ n_\ell) = \sum_{n_{\ell-1}=1}^{n_\ell} \cdots \sum_{n_{\ell-j}=1}^{n_{\ell-j+1}} \pi_\ell(n_{\ell-1}|n_\ell)\cdots\pi_{\ell-j+1}(n_{\ell-j}|n_{\ell-j+1}).$$

$\psi_{\ell,k}(n_\ell)$ may be considered therefore as the intensity of a stationary point process resulting from the superposition of $(\ell+1)$ *independent* stationary renewal processes, each of intensity $[A_\ell(\ell-j;\ n_\ell)(1-\pi_{\ell-j}(n_{\ell-j}|n_{\ell-j}))\mu_{\ell-j}p_{(\ell-j)k}]$, $j=0,\ldots,\ell$. It has been proved (cf. Chapter V, in [Kh55]) that such a superposed process rapidly approaches a Poisson Process, as the number $(\ell+1)$ of individual renewal processes increases. This result holds true whatever the distribution of the time intervals between successive renewals in each individual process is, and thereby, in this particular case, whatever the distribution functions $B_j(x)$, $j=0,\ldots,\ell$, with respective means

$$\mu_j^{-1} = \int_0^\infty x\, d\, B_j(x) < \infty,$$

of the service times of resource $R_j$ are. For $\ell$ large, one could therefore envisage analyzing each subsystem $\mathfrak{M}_\ell(n_\ell)$, $n_\ell=N,\ldots,1$, as a finite queuing system $G|M|1|n_\ell$ with an arbitrary inter-arrival time distribution $B_\ell(x)$ and a Poisson service process of parameter $\psi_{\ell-1,\ell}(n_{\ell-1})$, dependent on the congestion $n_{\ell-1}$, $n_{\ell-1}=0,\ldots,n_\ell$.

Under appropriate circumstances, more adequate assumptions than the classical Poissonian service times assumption may therefore be made if necessary, when an aggregative closed queuing model as defined in section IV is used.

## VI. NEAR-COMPLETE-DECOMPOSABILITY IN COMPUTING SYSTEMS

We deal in this section with two near-complete-decomposable models of computer system operations. The first one which is more especially hardware oriented, is the model of a computer memory organized as a hierarchy of storage devices of increasingly slower access speed in which computations are executed on a multiprogramming basis. Near-complete-decomposability conditions of this model turn out to conform with those a computation must satisfy to minimize the frequencies at which it needs to access the slower levels of the hierarchy. Given a statistical definition of the computations, the model set up in Section IV is then used to define and evaluate performance criteria for multiprogramming storage hierarchies; an optimal degree of multiprogramming minimizing the average hierarchy access time is evaluated. We finally show how state-dependent transfer probabilities (cfr. §5.2) may render the model sensitive to allocation policies which adjust the space allotted to each computation at every memory level as a function of the state of the multiprogrammed computations.

In the second part, which is more software oriented, the model of aggregate resources is interpreted as a simplified model of the resource control and allocation function of a multiprogramming operating system. We conjecture that conditions for resource aggregation provide in many cases rational criteria to order the levels of abstraction [Di68] [Di69/1] of such a system. These conditions turn out to be closely related to the physical characteristics and the usage of the hardware resources and may therefore help the designer choose the ordering of these levels.

Aggregation of variables appears in these two models as a technique which allows the level by level evaluation of the system to be interlaced with its level by level design. This results from the fact that the conditions for aggregation are the same as those necessary to provide sufficiently approximate knowledge of the performances of a still incomplete system on which further design decisions may be based. As such, the aggregative model of a network of queues defined in Section IV is a counterpart of the level-by-level simulation techniques recommended in [Pa67] [Pa69] and [Zu68] [Ra69], in the sphere of computer system analytical models.

## 6.1 Storage Hierarchies

### 6.1.1 Computations

Although the conclusions of this section apply to a wider class of hierarchies, we shall make use of the concept of linear storage hierarchy, defined and investigated in [Mg70], to introduce a stochastic definition of program execution.

Let $M_0, M_1, \ldots, M_L$ be a linear storage hierarchy of memory levels $M_\ell$, $\ell=0,\ldots L$; the higher the number of the level is, the larger its capacity tends to be and the lower its access speed. We define the capacity $c_\ell$ of $M_\ell$ as the maximum number of distinct information elements $M_\ell$ may contain.

A *computation* $\rho$ in this hierarchy may be identified in machine-independent terms by a *reference string* [De70], i.e., a sequence of references $\rho = (r_1^\rho, \ldots, r_y^\rho, \ldots, r_{d(\rho)}^\rho)$ where it is understood that, if $r_y^\rho = \mathscr{P}$, computation $\rho$ references a (not necessarily distinct) element of information

of name $\mathcal{A}^{\rho}$ at the yth reference. We define $\tau(r_y^{\rho})$ as the time instant at which computation $\rho$ makes its yth reference; $\tau(r_{y+1}^{\rho}) > \tau(r_y^{\rho})$.

We assume, as is generally the case, that a reference may only be serviced from the fastest level $M_0$; and we denote by $(\mu_0)^{-1}$ the expectation of the amount of time needed by a computation $\rho$ to complete a reference to an element $r_y^{\rho}$ which is located in $M_0$, $(r_y^{\rho} \in M_0)$, at time instant $\tau(r_y^{\rho})$. An element $r_y^{\rho}$ located in $M_{\ell \neq 0}$, at the time $\tau(r_y^{\rho})$ at which it is referenced, must be accessed and retrieved in $M_{\ell}$, and transferred from $M_{\ell}$ to $M_0$ where the reference may be serviced. Instead of being transferred one by one, elements are transferred by blocks, called pages, between levels. We denote $(\mu_{\ell})^{-1}$ the expectation of the amount of time needed to access and retrieve in $M_{\ell \neq 0}$ a page containing a given element $r_y^{\rho} \in M_{\ell}$, and to transfer this page from $M_{\ell}$ to $M_0$. The major component of $(\mu_{\ell})^{-1}$ is the time needed to retrieve and access the page containing the requested element. This is a characteristic of each memory level. We assume that memory levels are identified by this characteristic and ordered in such a way that

$$(\mu_1)^{-1} \ll (\mu_2)^{-1} \ll \ldots \ll (\mu_L)^{-1}. \tag{6.1}$$

A linear storage hierarchy is a hierarchy in which the only paths to move pages up the hierarchy are direct ones from each level $M_{\ell}$ to level $M_{\ell+1}$, $\ell=0,\ldots,L-1$. Paths to move pages down are unrestricted. Let $f_{\ell}^{\rho}$, $\ell=0,\ldots,L$, denote the number of elements $r_y$ which, in a given reference string $\rho$ of length $d(\rho)$ are located in $M_{\ell}$ at time instant $\tau(r_y)$. A procedure which determines these numbers according to page size, replacement rule, and capacity of each level of a linear demand paging hierarchy may be

found in [Mg70]; for a large class of replacement rules it is shown that $f_\ell^\rho$ may be obtained as a function $f_\ell^\rho(C_{\ell-1},c_\ell)$ of $c_\ell$ and of the total lower levels' capacity $C_{\ell-1} = \sum_{k=0}^{\ell-1} c_k$, $\ell > 1$; for $\ell=0$, we denote this function $f_0^\rho(c_0)$.

The *relative access frequency* to $M_\ell$ of a given reference string $\rho$ is then given by

$$F_\ell^\rho(C_{\ell-1},c_\ell) = d(\rho)^{-1} \times f_\ell^\rho(C_{\ell-1},c_\ell) \qquad \ell=1,\ldots,L,$$

with

$$F_0^\rho(c_0) = 1 - \sum_{\ell=1}^{L} F_\ell^\rho(C_{\ell-1},c_\ell).$$

Letting n be an arbitrary number of distinct reference strings $\rho$, $\rho=1,\ldots,n$, each with access frequencies $F_\ell^\rho(C_{\ell-1},c_\ell)$, $\ell=1,\ldots,L$, we may define an *access probability* $p_\ell(C_{\ell-1},c_\ell)$ as

$$p_\ell(C_{\ell-1},c_\ell) = \lim_{n\to\infty} \frac{\sum_{\rho=1}^{n} f_\ell^\rho(C_{\ell-1},c_\ell)}{\sum_{\rho=1}^{n} d(\rho)}, \quad \ell=1,\ldots,L$$

with $p_0(c_0) = 1 - \sum_{\ell=1}^{L} p_\ell(C_{\ell-1},c_\ell)$.

The set of probabilities $\{p_\ell(C_{\ell-1},c_\ell)\}_{\ell=0}^{L}$ is taken as the definition of the stochastic behavior of a computation in the hierarchy $M_0,\ldots,M_L$; the remainder of this section 6.1 is applicable to any memory hierarchy provided such a set of probabilities may be defined.

### 6.1.2 Single Process Storage Hierarchy

Let us assume that one computation at a time is executed in the hierarchy. Then, we may suppose that the time instant $\tau(r^{\rho}_{y+1})$ at which a computation makes its (y+1)th reference is also the time instant at which the yth reference is completed. The mean time interval between two successive references of any computation is in this case:

$$E[\tau(r^{\rho}_{y+1}) - \tau(r^{\rho}_{y})] = \begin{cases} (\mu_0)^{-1}, \text{ if } r_y \in M_0 \text{ at time } \tau(r^{\rho}_y) \\ (\mu_0)^{-1} + (\mu_\ell)^{-1}, \text{ if } r_y \in M_{\ell \neq 0} \text{ at time } \tau(r^{\rho}_y) \end{cases} \quad (6.2)$$

An *average hierarchy access time* $\bar{T}$ may then be defined as

$$\bar{T} = p_0(c_0) \times (\mu_0)^{-1} + \sum_{\ell=1}^{L} p_\ell(C_{\ell-1}, c_\ell) \times [(\mu_0)^{-1} + (\mu_\ell)^{-1}]$$

$$\bar{T} = (\mu_0)^{-1} + \sum_{\ell=1}^{L} p_\ell(C_{\ell-1}, c_\ell) \times (\mu_\ell)^{-1}, \quad (6.3)$$

as well as an average fraction of time $\eta$ lost in transferring pages between any one level $M_\ell$, $\ell > 1$, and $M_0$:

$$\eta = \frac{\bar{T} - (\mu_0)^{-1}}{\bar{T}} \quad (6.4)$$

### 6.1.3 Multiprocess Storage Hierarchy

Suppose now that transferring pages from $M_{\ell \neq 0}$ to $M_0$ and servicing references in $M_0$ may be performed simultaneously. Suppose also that memory space at the various levels is equally, statically and permanently shared among N independent computations $\rho$, $\rho=1,\ldots,N$, executed in a multiprogramming

basis. This means that the execution of the N reference strings is interleaved in the following way: iff $r_y^\rho \in M_{\ell \neq 0}$ at instant $\tau(r_y^\rho)$ and iff $\exists$ a computation $\rho'$ of which $r_{y'}^{\rho'}$ is the last reference completed and for which $r_{y'+1}^{\rho'} \in M_0$ at some time $t \geq \tau(r_y^\rho)$ before the transfer of the page containing $r_y$ is completed, then $r_{y'+1}^{\rho'}$ is serviced at time t.

This scheme provides for a better utilization of $M_0$ in which references may now be serviced while pages are being transferred. As a consequence of this however, the performance measures defined by (6.3) and (6.4) are no longer valid: they require indeed that only one reference to any one level $M_\ell$, $\ell=0,\ldots,L$, be made and completed at a time, whereas under multiprogramming time intervals $[\tau(r_{y+1}^\rho) - \tau(r_y^\rho)]$ of distinct computations $\rho$ may overlap each other.

In a multiprogramming storage hierarchy, the average time during which $M_0$ is not being referenced within a time interval $[\tau(r_{y+1}^\rho) - \tau(r_y^\rho)]$ may be less than $(\mu_\ell)^{-1}$ when $r_y^\rho \in M_{\ell \neq 0}$ at time $\tau(r_y^\rho)$. On the other hand, as one page only may be transferred from a level $M_{\ell \neq 0}$ to $M_0$ at a time, requests for page transfers will eventually queue up at each level and time intervals $[\tau(r_{y+1}^\rho) - \tau(r_y^\rho)]$ will be prolonged by these queuing times.

Performance criteria corresponding to (6.3) and (6.4) which allow the benefit of multiprogramming to be assessed will be introduced in paragraph 6.1.5.

### 6.1.4 Near-Complete-Decomposability

Let us identify each memory level $M_\ell$ with a resource $R_\ell$ with exponentially distributed service times of mean $\mu_\ell^{-1}$, and each of the N

multiprogrammed computations to a sequential process stochastically defined by the same probability set $\{p_\ell(\frac{C_{\ell-1}}{N}, \frac{c_\ell}{N})\}_{\ell=0}^{L}$. Let $i_\ell$, $\ell=1,\ldots,L$, $i_\ell=0,\ldots,N$, be the number of computations which at some time t are waiting for a page to be transferred from $M_{\ell\neq 0}$ to $M_0$, and let

$$i_0 = N - \sum_{\ell=1}^{L} i_\ell.$$

Defined in this way, this model of a multiprogramming storage hierarchy is a closed multiqueue system whose state may be defined by the (L+1)-plet $(i_0,\ldots,i_L)$ and whose transition stochastic matrix is Q(N,L). Conditions for near-complete-decomposability of this system into subsystem $Q(n_\ell, \ell)$, $n_\ell=1,\ldots,N$, $\ell=L-1,\ldots,1$, reduce to (see remark 2 , end of section III):

$$\text{for } \ell=L-1,\ldots,1: \quad \min_{0<k\leq\ell} \mu_k \gg \mu_{\ell+1},$$

$$\text{and} \quad \mu_0[\sum_{m=1}^{\ell} p_m(\frac{C_{m-1}}{N}, \frac{c_m}{N}) - p_{\ell+1}(\frac{C_\ell}{N}, \frac{c_{\ell+1}}{N})] \gg \mu_{\ell+1}, \tag{6.5}$$

since transfer probabilities $\{p_{\ell 0}\}_{\ell=1}^{L}$ are all equal to unity.

Conditions (6.5) are expressed in terms of rates at which a page may be retrieved, accessed and transferred from each level, and in terms of rates at which levels are referenced by an individual computation. More precisely, the set of probabilities $\{p_\ell(\frac{C_{\ell-1}}{N}, \frac{c_\ell}{N})\}_{\ell=0}^{L}$ defines the stochastic behavior of a computation in a single process storage hierarchy of capacities $\{\frac{c_\ell}{N}\}_{\ell=0}^{L}$. These probabilities follow from the statistical properties of the reference strings, but also from the choice of storage management parameters such as the memory level capacities, the page size, the replacement rule. The near-complete decomposability of the multiprogramming storage hierarchy model may be assessed by using e.g., a technique as the one described in

[Mg70], provided that these parameters be the same for each multiprogrammed computation.

One may expect inequalities (6.5) to hold in most storage hierarchies for various reasons. First, the lower bound for the average time needed to complete a single reference in a N-process hierarchy, which corresponds to the case of no queuing of page transfer requests, is equal to

$$(\mu_0)^{-1} + \sum_{\ell=1}^{L} p_\ell \left(\frac{C_{\ell-1}}{N}, \frac{c_\ell}{N}\right) \times (\mu_\ell)^{-1}. \tag{6.6}$$

Minimizing (6.6) is mandatory to optimize the average execution time of a computation. Both the user and the system storage management policies will aim at this insofar as they have control over the placing of information elements in the various levels, i.e., over the values of the access probabilities. But, as a result of inequalities (6.1), the only possible way to minimize (6.6), is to achieve the set of inequalities

$$p_0\left(\frac{c_0}{N}\right) \gg p_1\left(\frac{C_0}{N}, \frac{c_1}{N}\right) \gg \ldots \gg p_L\left(\frac{C_{L-1}}{N}, \frac{c_\ell}{N}\right), \tag{6.7}$$

which, together with (6.1), are (see remark (2), end of section III) sufficient conditions for (6.5) to hold.

Secondly, it has been observed that computations enjoy what has been called the *property of locality* [Be66, Va67, Be68, De68/1, De70]; that is they favor at each instant of their execution a subset of their information instead of scattering their references uniformly over their total set of information, this subset changing membership relatively slowly in the course of the computation execution.

As a consequence of this, every page transfer which is occasioned by a single reference to one upper level is likely to generate several references to $M_0$. On the other hand, page replacement policies like LRU (see e.g., [De68/2]) take advantage of this property to accentuate inequalities (6.7). In most cases, therefore, conditions for L-level near-complete-decomposability will be satisfied.

*Remark.* It is probably worth saying a word about what happens when inequalities (6.1) only are verified and not inequalities (6.6). One could show that in this case stochastic matrices $Q(n_\ell, \ell)$, $\ell=1,\ldots,L$, are (see remark 1.6) near-decomposable *block-triangular* matrices in the sense of Ando and Fisher [An63]. They proved that the Simon-Ando theorem could be modified to the case of such matrices. A resource aggregate model analogous to the model defined in Section IV could therefore be set up to cope with such systems.

### 6.1.5 Memory Level Aggregation

If conditions (6.5) are verified, the aggregative model defined in Section IV may be used. We proceed now by demonstrating how performance criteria comparable to (6.3) and (6.4) may be defined within that model.

We regard the aggregate $\mathfrak{M}_\ell(n_\ell)$, $n_\ell=1,\ldots,N$; $\ell=1,\ldots,L$, as a storage hierarchy of levels $M_0,\ldots,M_\ell$, in which $n_\ell$ computations defined by a same probability set $\{p_k(\frac{c_{k-1}}{N},\frac{c_k}{N})\}_{k=0}^{\ell}$ are being multiprogrammed. $\sigma_\ell(n_\ell)$ is defined as the mean number of references serviced in $M_0$ per time unit within this aggregate, and is given by (4.1), (4.2) which reduce to

$$\sigma_\ell(n_\ell) = \sum_{n_{\ell-1}=1}^{n_\ell} \pi_\ell(n_{\ell-1}|n_\ell)\sigma_{\ell-1}(n_{\ell-1}) \tag{6.8}$$

$$\sigma_1(n_1) = \mu_0(1-\pi_1(0|n_1)), \quad \ell=2,\ldots,L \tag{6.9}$$

$$n_1,\ n_\ell=1,\ldots,N.$$

In the same way, $\psi_{\ell,k}(n_\ell)$ defined as the rate at which aggregate $\mathfrak{M}_\ell(n_\ell)$ references memory level $M_k$, $k > \ell$, is given by relations (4.3) and (4.4) in which

$$p_{\ell k} = 0, \quad \ell=1,\ldots,L\ ,\ k > \ell,$$

$$p_{0k} = p_k(\frac{C_{k-1}}{N},\frac{c_k}{N}).$$

Conditional probabilities $\pi_\ell(n_{\ell-1}|n_\ell)$ are yielded by (4.7.2) in which

$$p_{\ell 0} = 1, \quad \ell=1,\ldots,L$$

$$p_{\ell i} = 0, \quad 0 < i \le \ell-1.$$

An average hierarchy access time $\bar{T}_N$ is then obtained as *the inverse of the average number of references serviced per time unit* in $\mathfrak{M}_L(N)$:

$$\bar{T}_N = [\sigma_L(N)]^{-1} = [\sum_{n_{L-1}=1}^{N} a_L(n_{L-1})\sigma_{L-1}(n_{L-1})]^{-1}, \tag{6.10}$$

the long-run equilibrium probability $a_L(n_{L-1})$, $n_{L-1}= 0,\ldots,N$, being given by (4.8).

Moreover, an average fraction of time during which all N computations are waiting for a page transfer from some memory level $M_\ell$, $\ell > 1$, to $M_0$ is given by

$$\eta_N = s_0(0)$$

It may be proved that $\eta_N$ and $\bar{T}_N$ are bound by the same relation (6.4) as $\eta$ and $\bar{T}$:

*Lemma* 6.1 $\eta_N = 1 - [\mu_0 \times \bar{T}_N]^{-1}$.

*Proof*: $s_0(0) \equiv a_1(0) = 1 - \sum_{n_0=1}^{N} a_1(n_0)$,

$$= 1 - \sum_{n_{L-1}=1}^{N} a_L(n_{L-1}) \sum_{n_{L-2}=1}^{n_{L-1}} \pi_{L-1}(n_{L-2}|n_{L-1}) \cdots \sum_{n_0=1}^{n_1} \pi_1(n_0|n_1).$$

Since by virtue of (6.9),

$$\sum_{n_0=1}^{n_1} \pi_1(n_0|n_1) = \mu_0^{-1} \times \sigma_1(n_1)$$

and by virtue of (6.8)

$$\sigma_\ell(n_\ell) = \sum_{n_{\ell-1}=1}^{n_\ell} \pi_\ell(n_{\ell-1}|n_\ell)\, \sigma_{\ell-1}(n_{\ell-1}),$$

we obtain

$$a_1(0) = 1 - \mu_0^{-1} \sum_{n_{L-1}=1}^{N} a_L(n_{L-1})\, \sigma_{L-1}(n_{L-1}),$$

which completes the proof.

$\eta$ and $\bar{T}$ being defined by (6.4) and (6.3) respectively, the following relations may also be established:

*Lemma* 6.2 $\bar{T}_1 = \bar{T}$; $\eta_1 = \eta$.

*Proof*: From the definition (6.10) of $\bar{T}_N$, it results that for N=1:

$$\bar{T}_1 = [a_L(1)\ \sigma_{L-1}(1)]^{-1} = [\pi_L(1|1)\ \sigma_{L-1}(1)]^{-1}. \tag{6.11}$$

In the queuing system $\mathfrak{M}_\ell(1)$, $\ell=2,\ldots,L$, $\pi_\ell(1|1)$ may be expressed as (see (4.7.2)):

$$\pi_\ell(1|1) = \frac{\psi^{-1}_{\ell-1,\ell}(1)}{\mu^{-1}_\ell + \psi^{-1}_{\ell-1,\ell}(1)} \tag{6.12}$$

In this particular case where transfer probabilities $p_{\ell k}$, $k > \ell$, $\ell > 0$ are all zero, (4.3) and (4.4) yields, using (6.8) and (6.9):

$$\psi^{-1}_{\ell-1,\ell}(1) = [p_\ell\ (C_{\ell-1},c_\ell) \times \sigma_{\ell-1}\ (1)]^{-1}, \quad \ell=2,\ldots,L. \tag{6.13}$$

Replacing $\pi_L(1|1)$ in (6.11) by the value yielded by (6.12) and then $\psi^{-1}_{L-1,L}(1)$ by the value yielded by (6.13), gives:

$$\bar{T}_1 = p_L\ (C_{L-1},c_L)\ \mu^{-1}_L + \sigma^{-1}_{L-1}\ (1).$$

Using the relation $\sigma_\ell(1) = \pi_\ell(1|1)\ \sigma_{\ell-1}(1)$, successively for $\ell=L-1,L-2,\ldots 2$, gives:

$$\bar{T}_1 = \sum_{\ell=L}^{2} p_\ell\ (C_{\ell-1},c_\ell)\ \mu^{-1}_\ell + \sigma^{-1}_1(1).$$

But $\sigma^{-1}_1(1) = [\mu_0\ \pi_1(1|1)]^{-1}$

where, by virtue of (4.5) and (4.6), $\pi^{-1}_1(1|1)$ may be replaced by

$$\pi^{-1}_1(1|1) = \frac{[\mu_0\ p_1(c_0,c_1)]^{-1} + \mu^{-1}_1}{[\mu_0\ p_1(c_0,c_1)]^{-1}}\ ;$$

which gives evidence that $\bar{T}_1 = \bar{T}$. Then $\eta_1 = \eta$ results directly from Lemma 6.1, which completes the proof.

Furthermore, let us remark that the *utilization factor* of the storage hierarchy component which performs transfers from $M_\ell$ to $M_0$, $\ell > 1$, is equal to $(1-s_\ell(0))$, probability $s_\ell(0)$ being defined by (4.8). Likewise (4.9) defines the mean response time $\bar{W}_\ell$ of this component.

Finally, if the uppermost level $M_L$ is the model of some infinite reservoir from which all computations originate at a rate given by (2.01) and to which they all return when being completed, then W, the mean response time of the aggregate of $M_0, M_1, \ldots, M_{L-1}$, defined by (4.10), is the mean time required to service all references of a reference string.

### 6.1.6 Dynamic Space Sharing

By virtue of lemmas 6.1 and 6.2, the performance improvement achieved by multiprogramming N computations so as to take advantage of the simultaneity between transfers among memory levels and servicing references in $M_0$ may be measured by the ratios:

$$\frac{\bar{T}_1}{\bar{T}_N} \quad \text{or} \quad \frac{\eta_1}{\eta_N} \tag{6.14}$$

In general, there will exist one optimal value $N_{opt}$ which maximizes these performance ratios. This optimum results from two counteracting effects: on the one hand the probability $(1-a_1(0))$ of having at least one computation not waiting for a page transfer from some level

$M_\ell$, $\ell > 1$, to $M_0$ is inclined to increase with N; on the other hand the average memory space $\frac{c_\ell}{N}$, $\ell=0,1,\ldots$ available to each computation at the lower levels shrinks as N increases with the consequence that the probabilities $p_\ell(\frac{C_{\ell-1}}{N}, \frac{c_\ell}{N})$ of accessing the upper levels increase rapidly. The probabilistic model described above provides a means to evaluate this optimum, given (i) a family of programs defined by their reference strings from which access probabilities may be inferred by e.g., a stack processing technique like that proposed in [Mg70], and (ii) a nearly-completely-decomposable memory hierarchy of an arbitrary depth, each memory level being characterized by its capacity and a distribution of the amount of time required to access, retrieve and transfer a page from this level to the executive memory.

So far we have assumed that a fixed and equal portion $\frac{c_\ell}{N}$, $\ell=0,\ldots L$ of memory space was permanently allocated at every level to every computation in the hierarchy. This is, however, not the case in most hierarchical storage systems. Usually, in order to keep moderate the access probabilities to the upper levels, an upper limit, say $J_{max}$, is imposed upon the number of computations J allowed simultaneously to share space in the faster memory levels, say $M_0,\ldots,M_{\ell(max)}$. Moreover, any of these computations is expropriated from the space it has accumulated in these levels as soon as it makes a reference to or above a certain level, say $M_k$, $k > \ell(max)$. Thus, if $n_{k-1}$ designates the number of computations not waiting for a page transfer from levels $M_k,\ldots,M_L$, we have

$$J = \min(n_{k-1}, J_{max});$$

and the average space allotted to any of these computations at any time

instant $\tau(r_y^\rho)$, $(y=1,2,\ldots)$, $(\rho=1,\ldots,J)$, at any level $M_\ell$, $\ell \le \ell(\max)$, is the integer part of $(c_\ell \times J^{-1})$.

This space allocation policy preserves the near-complete-decomposability property of the hierarchy since its net effect is to enhance the difference between access probabilities to the lower levels $M_0,\ldots,M_{\ell(\max)}$, and the upper levels $M_{\ell(\max)+1},\ldots,M_L$. $J_{\max}$ distinct access probability distributions, conditioned on the value of $n_{k-1}$, must be considered:

$$D(\alpha_i): \begin{cases} p_\ell\left(\sum_{i=0}^{\ell-1} \frac{c_i}{\alpha_i}, \frac{c_\ell}{\alpha_\ell}\right), & \ell=1,\ldots,L \\ p_0 = 1 - \sum_{\ell=1}^{L} p_\ell\left(\sum_{i=0}^{\ell-1} \frac{c_i}{\alpha_i}, \frac{c_\ell}{\alpha_\ell}\right), \end{cases} \tag{6.15}$$

where $\alpha_i = \begin{cases} \min(n_{k-1}, J_{\max}), & i=0,\ldots,\ell(\max), \\ N, & i=\ell(\max)+1,\ldots,L. \end{cases}$

There are three stages in the analysis of such a hierarchy:

(i) The bottom levels $M_0,\ldots,M_{\ell(\max)}$ are modelled by $J_{\max}$ aggregates $\mathfrak{M}_{\ell(\max)}(J)$, $J=1,\ldots,J_{\max}$, each one with the appropriate access probability distribution $D(J)$. The analysis of these aggregates yields the interactive rates $\psi_{\ell(\max),i}(J)$ of the bottom levels upon each level $i = \ell_{\max}+1,\ldots,L$.

(ii) Next, one considers N aggregates $\mathfrak{M}_{k-1}(n_{k-1})$, $n_{k-1}=1,\ldots,N$. The interaction rate of the bottom levels upon any one intermediary level $M_{i'}$, $i'=\ell_{\max}+1,\ldots,k-1$, in each of these aggregates is taken equal to

$$\begin{cases} \psi_{\ell(\max),i'}(n_{k-1}) & \text{if } n_{k-1} < J_{\max} \\ \psi_{\ell(\max),i'}(J_{\max}) & \text{if } n_{k-1} \geq J_{\max}. \end{cases}$$

(iii) The analysis of each aggregate $\mathfrak{M}_{k-1}(n_{k-1})$ yields the interactive rates $\psi_{k-1,i''}(n_{k-1})$ upon the upper levels $i''$, $i''=k,\ldots,L$; using these interaction rates in (4.7.2) and (4.5), the upper levels may be modelled by a system $\mathfrak{M}_L(N)$.

The same argument which leads to an optimal value for N, is applicable to $J_{\max}$. The model outlined above may be used to estimate the value $J_{\max}^{opt}$ of an optimal *maximum degree of multiprogramming* in the lowest memory levels, which for a given N value maximizes the ratios (6.14). Such an optimal value $J_{\max}^{opt}$ will be calculated in the last section for a hypothetical computer system enjoying the space allocation policy discussed above.

### 6.1.7 Remark

A variant of the model investigated here may be found in [Cou70/1], [Cou70/2]. There, a computation is stochastically defined by a set of arbitrary distribution functions

$$B_\ell(x) = \text{Prob}\{\zeta_\ell \leq x\}, \quad \ell=1,\ldots,L$$

where the random variable $\zeta_\ell$ is the number j of references made by a computation $\rho$ within the time interval $(\tau(r^\rho_{y+j}) - \tau(r^\rho_y)]$ if $\tau(r^\rho_y)$ and $\tau(r^\rho_{y+j})$ are two successive instants of an access to memory level $M_\ell$. If n is an arbitrary number of computations, the expectation $\bar{\zeta}_\ell$ and the

access probability $p_\ell(C_{\ell-1},c_\ell)$ are related by

$$\bar{\zeta}_\ell = \int_0^\infty x\,dB_\ell(x) = \lim_{n\to\infty}\left[\frac{\sum_{\rho=1}^{n} f_\ell^\rho(C_{\ell-1},c_\ell)}{\sum_{\rho=1}^{n} d(\rho)}\right]^{-1} = [p_\ell(C_{\ell-1},c_\ell)]^{-1}.$$

When the system is nearly completely decomposable, the conditional probability distribution $\pi_\ell(n_{\ell-1}|n_\ell)$, $\ell=1,\ldots,L$, $n_\ell=1,\ldots,N$, is obtained as the distribution of the congestion in a $M|G|1|N$ queuing process whose distinctive feature is that the service rate $\sigma_{\ell-1}(n_{\ell-1})$, $n_{\ell-1}=1,\ldots,n_\ell$ (defined by (6.8), (6.9)) at which references are serviced is dependent upon the current congestion $n_{\ell-1}$. This type of $M|G|1|N$ queuing process is studied in [Cou71].

## 6.2 Hierarchical Structure of Multiprogramming Computer Operating Systems

The technique of aggregation proved useful to (i) break a system up into a small number of subsystems, (ii) evaluate the interactions within the subsystems as though interactions among subsystems did not exist, and (iii) evaluate the interactions among subsystems without regard to the interactions within sybsystems. We may expect that this evaluation technique has some similarity with the design technique which consists of assembling a complex system from subsystems designed independently; we may expect that the criteria that indicate what variables to aggregate might also help in specifying what the building blocks should be.

In particular, it seems worth discussing the similarity existing between the hierarchical model of aggregate resources defined in Section IV and the well known hierarchical organization advocated by E. W. Dijkstra for the software of multiprogramming computer systems [Di69/1].

### 6.2.1 Levels of Abstraction

An essential function of a multiprogramming computer operating system is to control and allocate hardware resources. The problem of creating a system which achieves this function could be formulated as follows: Given the hardware of a computer, construct a set of programs whose function is to provide a collection of convenient and efficient abstractions from some physical characteristics (quantity, speed, access mechanism,...) of these resources; two constraints which further complicate the problem are that:

(i) to achieve these abstractions, the programs themselves use or consume hardware resources;

(ii) if these abstractions are to serve any useful purpose, their designer, because of his own "inability to do much" [Di69/2], should be their first beneficiary; rather than using and being defined in terms of raw hardware resources, the programs should insofar as possible be defined in terms of and use these abstractions.

E. W. Dijkstra showed [Di68], [Di69/1] that, from a designer point of view, it is advantageous to structure such a system as a hierarchy of *levels of abstraction*. He considers an ordered sequence of machines $A_0, A_1, \ldots, A_\ell, \ldots$ where the level $A_0$ is the given hardware and where the software of level $\ell$, $\ell=0,1,\ldots$, defined in terms of and executed by machine $A_\ell$, transforms machine $A_\ell$ into machine $A_{\ell+1}$. The software of each level $\ell$ creates an abstraction from some physical properties of the hardware. Examples of such abstractions implemented in the THE system are:

- at level 0, the central processor is allocated among concurrent processes so that the actual number of these processor(s) (one in this system) is no longer relevant: each process ready to use a processor may be considered above this level as having access to its own virtual processor;

- at level 1, the differences in mode and speed of access between a drum and a core memory are abstracted from in order to create an homogeneous store. Above this level the physical location of information in this store is no longer relevant; information is identified by a segment name.

Each level gives access to a certain type of *abstracted resource*, e.g., a virtual Processor, a segmented address space,... To do so, it has recourse only to the abstractions created at the lower levels. This approach facilitates the step by step construction, testing and evaluation of the system. These advantages are the direct consequence of the restricted nature of the interactions which are permitted among distinct levels. This very same restriction, however, leads to the following dilemma: if we use the simplification introduced by abstracting from device $R_\ell$ in producing the abstraction from device $R_m$, then we cannot use the simplification introduced by abstracting from $R_m$ in writing the programs that abstract from $R_\ell$; in other words, each level helps only those above it. The choice of the ordering of the abstractions may therefore be quite difficult.

### 6.2.2 Aggregation and Ordering of Abstractions

It is possible to identify at least two types of conditions which must be satisfied to decide that the abstraction from a device $R_i$ should be created at a lower level than the level of abstraction of a device $R_j$ of another type:

1) The abstraction from the details of device $R_i$ should be more *convenient* to program the abstraction from device $R_j$ than conversely; if the abstraction from $R_i$ is not more convenient, it need not come below the level of abstraction from $R_j$.

2) The use of an abstraction from $R_i$ to program the abstraction from $R_j$ should not prevent the latter resource from being controlled and allocated *efficiently*

We may (and in fact we should) expect that in most cases the degree to which an abstracted resource is a 'convenient' tool or concept to program will be reflected in the execution of the program by the frequency at which this abstracted resource is accessed. Then, the result of applying the first condition to each type of abstraction is that the abstracted resources implemented at the lower levels of abstraction are those which will be the more frequently accessed.

The second condition must be interpreted as follows. Drums, disks, tapes, readers,... are mutually asynchronous I/O devices, each one being capable of transferring information at a certain speed. In order to control and allocate efficiently one of these devices, a program must be able to execute at a speed comparable with the speed of the device; in other words, the speeds at which the resources (abstracted or not) used by such a program can be accessed, e.g. those of an abstract machine $A_\ell$, should not be slower than the speed of the hardware being abstracted from at level $\ell$; or, at least, the resources which make exception to that rule should be used infrequently.

The application of conditions 1) and 2) to each type of abstraction leads to an ordering such that the more frequently accessed abstracted resources should be implemented at the lower levels and should also have faster access. The interactions between the levels of an 'efficient' and 'convenient' hierarchy of abstractions should, in other words, obey the conditions for near-complete decomposability.

Since the speed of access to an abstracted resource is bound by the speed of the hardware which is abstracted from, conditions for near-complete-decomposability explain why the fastest and more frequently used hardware

is taken care of at the lowest levels of a hierarchy of abstractions. These conditions specify in terms of usage and speed the type of hardware which should advantageously be abstracted from at a certain level of abstraction and may thereby give the designer some guidance in the choice of the ordering of these levels.

Another consequence is that each level of abstraction may be analyzed as a level of aggregation. Suppose for example that the model of the network of queues defined in Chapter II may be taken as a simplified model of a set of interacting abstracted resources; the abstract machines $A_1,\ldots,A_\ell,\ldots$ could then be evaluated respectively by the aggregates $\mathfrak{M}_1(n_1)$, $\mathfrak{M}_2(n_2),\ldots,\mathfrak{M}_\ell(n_\ell),\ldots$, each aggregate $\mathfrak{M}_\ell(n_\ell)$ being a model of the equilibrium attained in the abstract machine $A_{\ell+1}$ by the interactions between $A_\ell$ and the abstracted resource implemented at level $\ell$. By definition, the dynamic behavior of the aggregate $\mathfrak{M}_\ell(n_\ell)$ towards its equilibrium may be evaluated with good approximation merely in terms of aggregative variables representative of the short-term equilibria attained at the lower levels of aggregation and without regard to the interactions with the upper levels. Since $A_1,A_2,\ldots$ represent as many distinct stages in the design, production and testing of an operating system, such an evaluation technique which allows these stages to be evaluated independently should prove more valuable than techniques which may only be used after the design is completed.

To close this section, let us say that the above discussion reveals a property of these stages $A_1,A_2,\ldots$ of the design process: the rate of interaction between the components of a machine $A_\ell$, i.e., between $A_{\ell-1}$ and the abstraction implemented at level $\ell$-1, is higher than the rate of interaction between the corresponding components of $A_{\ell+1}$. Several arguments in favor of a decomposition of the design process into successive stages at which the subsystems coped with have a similar property, may be found in C. Alexander's essay, "Notes on the Synthesis of Form" [Al64].

# VII. SHORT- AND LONG-RUN EQUILIBRIA IN A TIME-SHARING PAGING MULTIPROGRAMMING SYSTEM

The purpose of this last section is to illustrate the use of an aggregative model in analyzing the performances of a given computing system. It follows on from this concrete case study that aggregation is not only adequate to obtain numerical results when a large number of parameters are involved, but also helps to gain insight and conceptual clarity on the parts played by these parameters.

A set of hardware and software dependent parameters is defined as the model of an hypothetical time-sharing paging system. Conditions for this model to be nearly completely decomposable are defined in terms of the parameters and in terms of a stochastic representation of the computing load. The formulation of these conditions makes their prior verification possible and may give an assessment of the precision which may be achieved by using an aggregative model.

Page traffic between primary and secondary memory is studied as the internal traffic of a resource aggregate of type $\mathfrak{M}_1(n_1)$. This analysis focuses on the respective influence on processor usage of the user program paging activity, the degree of multiprogramming and the length of a so-called execution interval, viz. the amount of processor time a user program is allowed to consume before losing its pages accumulated in primary memory.

The entire system is then regarded at a higher level of analysis as a finite set of N active user terminals supplying tasks for this aggregate.

The congestion and the response time of the hypothetical system are studied as the characteristics of a system $\mathfrak{M}_2(N)$. The concept of *stable* and *unstable* congestions is introduced as the consequence of the peculiar dependency of the aggregate service rate on the system congestion. Likewise, a *saturation* point extending the definition given in [K168] to systems with congestion-dependent service rate is defined. Finally, the evaluation of the system response time reveals that these concepts of stability and saturation relate the phenomenon known as thrashing to parameters defining the computing load upon the system in addition to those defining the page traffic between primary and secondary memory.

## 7.1 The Hypothetical System

A schematic representation of this hypothetical system is given in figure 7.1. Three types of function are essentially ensured:

7.1.1. A finite number N of *active* user terminals originate random requests for program execution.

7.1.2. User programs are executed on a multiprogrammed basis in a primary memory $M_0$ consisting of set of $C_0$ page frames.

7.1.3. Pages which cannot be contained in $M_0$ are swapped in a rotational secondary memory $M_1$ ($T_{rot}$ = duration of a rotation).

These functions are supposed to comply with the following strategies:

7.1.4. A terminal cannot originate a request for program execution before the previous request issued from the same terminal has been served

and completed. In other words, there may exist a maximum of one program per terminal in the system at a time.

7.1.5. User programs are loaded from $M_1$ where all pages are supposed to be initially located into $M_0$ on a *page on demand strategy*. An upper limit K is imposed upon the number of page frames a user program may occupy in $M_0$. Pages of a same user program superimpose each other whenever the number of distinct pages required in $M_0$ by this program exceeds this upper limit.

7.1.6. *Multiprogramming*: At any moment of its lifetime in $M_0$, a user program is in one of three states:

| | |
|---|---|
| *ready* | i.e., demanding but not receiving the control of the processor |
| *running* | receiving the control of the processor |
| *suspended* | waiting for a page transfer between $M_1$ and $M_0$ to be completed |

Multiprogramming means that user programs are concurrently executed in $M_0$ in order to maintain the processor busy as long as not every user program is suspended. We assume that a maximum number $J_{max}$, $0 < J_{max} \leq N$, of user programs may at most be concurrently executed in $M_0$. Further requests for program executions are queued until one of the $J_{max}$ programs ceases to be multiprogrammed (see 7.1.7). $J_{max}$ will be referred to as the *maximum degree of multiprogramming*.

7.1.7. *Time-slicing*: A program ceases to be multiprogrammed either when it is completed or when the total time it has spent in running state since its last loading in $M_0$ reaches a maximum value Q. In this latter eventuality and if programs are waiting to be multiprogrammed, the program will lose all its pages accumulated in $M_0$, and will join the queue of programs waiting to be multiprogrammed until it is allocated an additional quantum Q. If no programs are waiting to ba multiprogrammed, the program keeps its pages in $M_0$.

7.18. The tracks in $M_1$ may be written and read in parallel by fixed heads. Each track is divided into $c_1$ sectors, each sector containing exactly one page. Each request for a page transfer is put into a queue associated with the sector containing the demanded page [Wei66].

## 7.2 The User Programs

The user programs are stochastically defined by the following random variables:

(i) the user reaction time, i.e., the time interval elapsing between the completion of a user program execution requested by a terminal and the origination of the next request by the same terminal

(ii) the total amount of processor time consumed by a user program

(iii) the amount of processor time $\xi_n$, n=1,2,..., consumed by a user program between any two successive references $r_n, r_{n+1}$ to <u>distinct</u> pages not already referred to, $r_{n+1} \neq r_n \neq r_{n-1} \neq \ldots \neq r_1$, (user program paging activity).

The distributions of these random variables are defined in the next section.

## 7.3 Simplifying Assumptions

The hypothetical system takes liberties with reality on a certain number of points:

7.3.1. Pages are assumed to be uniformly distributed over the $c_1$ sectors of $M_1$; the probability of a demanded page being located in a specific sector is thereby equal to $(\frac{1}{c_1})$.

7.3.2. We suppose that the traffic of demanded pages from $M_1$ to $M_0$ is not hindered by the opposite traffic of pages from $M_0$ to $M_1$. This assumption is valid if a page frame is permanently maintained vacant in $M_0$ as well as in each sector of $M_1$ [Sm67]. Hence we need only to take into account the traffic of pages from $M_1$ to $M_0$.

7.3.3. The maximum number of distinct pages a program may accumulate in $M_0$ at some time t is a function of the number J, $1 \leq J \leq J_{max}$ of programs being multiprogrammed at that time t. This function is supposed to be simply

$$K(J) = \underline{\text{entier}}\ (\frac{C_0}{J}).$$

7.3.4. We define an *execution interval* as the amount of processor time consumed (i.e., the amount of time spent in *running* state) by a user program between two consecutive loading of this user program from $M_1$ into $M_0$. An execution interval is at most equal to Q.

The total amount of processor time consumed by a user program is considered as an independently, identically, negative exponentially distributed random variable with mean $(\varphi)^{-1}$. As a result of this assumption, an execution interval is an i.i.d.r.v. with distribution function:

$$Q(t) = \text{Prob}(x \le t) = \begin{cases} 0 & (t \le 0) \\ 1-e^{-\varphi t} & (0 < t < Q) \\ 1 & (Q \le t) \end{cases}$$

Denoting $\varkappa(S)$ the Laplace-Stieltjis transforms of Q(t), one obtains:

$$\varkappa(s) = \int_0^\infty e^{-st}\, d\, Q(t) \qquad (\text{Re}(s) > 0)$$

$$= \int_0^Q e^{-st}\, te^{-\varphi t} dt + e^{-sQ-\varphi Q}$$

$$= \frac{\varphi + se^{-sQ-\varphi Q}}{s+\varphi}$$

Let $(n)^{-1}$ denote the mean length of an execution interval:

$$(\mu)^{-1} = \varkappa'(0) = (1-e^{-Q\varphi})\varphi^{-1}.$$

Q(t) will be approximated by an exponential distribution of parameter $\mu$.

7.3.5. User reaction times are considered as i.i.d.r. exponentially distributed with parameter $\lambda$; a discussion of the validity of this assumption may be found in [Cof66].

7.3.6. Random variables $\xi_n$, n=1,2,..., are considered as i.i.d.r. exponentially distributed with respective parameters $\theta_n$, n=1,2,... .

## 7.4 Actualizing the Hypothetical System

The Hypothetical System is now defined in terms of the parameters $C_0$, $c_1$, $T_{rot}$, Q, $J_{max}$. Although the aggregative model of section 4 is in principle adequate for investigating the influence of any of these parameters on the performance measures defined by (4.7), (4.8), (4.9) and (4.10), we shall center our analysis around the part played by $J_{max}$ and Q. And in order to be able to put our results in concrete form, we shall attribute to each of the other parameters a fixed, and hopefully representative value, namely:

$$C_0 = 48 \text{ page frames},$$
$$c_1 = 4 \text{ sectors},$$
$$T_{rot} = 20.10^{-3} \text{ seconds}.$$

Likewise, as far as the load on the system is concerned, we shall restrict ourselves to the study of the system sensitivity to fluctuations of the number N of active user terminals. The parameters of the distribution functions which define the users' and their programs' stochastic behavior are assigned values inferred from statistical observations. At the time this example was elaborated, available statistics were those observed by SDC, [Fin66] [To65], on the Time-Sharing system Q-32, [Sc64] [Sc67]:

- The total average processor time $(\varphi)^{-1}$ consumed per user program is taken equal to 1.39 sec.

- The average user reaction time $(\lambda)^{-1}$ is taken equal to 32 seconds.

- The paging activity of user programs in the course of their execution is given in figure 7.3. It is based on a page size of 1024 words. The number $\gamma_n$ of instructions executed between any

two successive references $r_n, r_{n+1}$ to distinct pages not already referenced ($r_{n+1} \neq r_n \neq r_{n-1} \neq \ldots \neq r_1$) is plotted against n. The exponential coefficients $\theta_n$ are given the values

$$\theta_n = (y_n \times 5.10^{-6})^{-1}, \quad n=1,2,\ldots,19,$$

where $5.10^{-6}$ sec. is the average total execution time of an instruction in the T-S Q-32 system. No more than 20 distinct pages are supposed to be referred to during a program execution so that $\theta_{20}=0$.

*Remark.* The branch of hyperbola in figure 7.3 is representative of the behavior of programs whose page references would be uniformly distributed over 20 pages. The probability of such programs making a reference to a page not yet referenced decreases linearly with n, the number of distinct pages already referenced. Obviously this probability decreases much more rapidly than linearly for the program executions actually observed. This gives partial evidence that these programs enjoy the *property of locality*, viz. they favor a subset of their pages at each instant of their execution. Such a program behavior has been discussed in [Be66] [Va67] [Be68] [De68/1] [De70]. The values which have been assigned to $\theta_n$, n=1,2,... are therefore representative of this behavior.

## 7.5 The Page Demand Rate

The page demand rate is the mean number of page transfers from $M_1$ to $M_0$ requested per time unit by a program in running state. In the evaluation of this rate two cases must be distinguished depending on whether

more than $J_{max}$ requests for program execution are pending or not.

7.5.1. In the first case the $J_{max}$ programs which are multiprogrammed lose all their pages in $M_0$ when the execution interval has run out. The minimum number of page transfers requested over an execution interval is therefore equal to the number of distinct pages referred to during this interval. This number is augmented whenever the number of distinct pages referred to exceeds $K(J_{max})$ so that pages have to superimpose each other.

Following the approach taken in [Sm67], the average number $\partial(J_{max}, \mu)$ of page transfers per execution interval may be obtained as the average number of events of a non-homogeneous Poisson process occurring within an exponentially distributed time interval:

$$\partial(J_{max},\mu) = 1 + \frac{\theta_1}{\theta_1+\mu} + \ldots + \prod_{n=1}^{K(J_{max})-1} \frac{\theta_n}{\theta_n+\mu} + \prod_{n=1}^{K(J_{max})-1} \frac{\theta_n}{\theta_n+\mu} \left\{ 1 - \frac{\theta_{K(J_{max})}}{\theta_{k(J_{max})}+\mu} \right\}^{-1},$$

or

$$\partial(J_{max},\mu) = 1 + \sum_{z=1}^{K(J_{max})-1} \prod_{n=1}^{z} \frac{\theta_n}{\theta_n+\mu} + \prod_{n=1}^{K(J_{max})-1} \frac{\theta_n}{\theta_n+\mu} \frac{\theta_{K(J_{max})}}{\mu} \qquad (7.1)$$

It is assumed in (7.1) that the page transfer request rate remains constant and equal to $\theta_{K(J_{max})}$ after page superimposition first occurs, viz. after the occurrence of the $K(J_{max})^{th}$ transfer request.

The page demand rate is thus equal in this first case to $\mu \times \partial(J_{max},\mu)$.

7.5.2. In the second case the J programs, $1 \le J \le J_{max}$, which are multiprogrammed do not lose at the end of each execution interval the pages they have accumulated in $M_0$. On the assumption that in this case programs

are never throughout their execution deprived of their $K(J_{max})$ page frames in $M_0$, the page demand rate may be approximated by $(\varphi \times \partial(J_{max}, \varphi)$. This assumption will be justified by the conditions for near-decomposability discussed in §7.8.

7.5.3. We will assume that the probability of a page missing in $M_0$ within some interval $(t,t+h]$ when a program is *running*, depends on h only and not on t, and is equal to

$$h \times \mu \times \partial(J_{max},\mu) + 0(h)$$

$$\text{or} \quad h \times \varphi \times \partial(J,\varphi) + 0(h), \quad 1 \le J \le J_{max},$$

according to the number of requests for program executions which are pending.

7.5.4. Some basic properties of the page demand rate may be found in expression (7.1), taking into account that coefficients $\theta_n$, $n=1,2,\ldots$, are representative of a general property of program paging activity. Considering only the first case above and denoting by $\Delta_{K(J_{max})}$ the page demand rate difference which would result from an allotment of an additional page frame $(K(J_{max}) = K(J_{max}) + 1)$ to each multiprogrammed user program, we have

$$\Delta_{K(J_{max})} = \prod_{n=1}^{K(J_{max})} \frac{\theta_n}{\theta_n+\mu} (\theta_{k(J_{max})+1} - \theta_{k(J_{max})}).$$

The shape of the paging activity displayed in figure 7.3 is, on the average, such that

$$y_1 \le y_2 \le \ldots y_{K(J_{max})}, \quad 1 \le K(J_{max}) \le 20.$$

Thus

$$\theta_1 \geq \theta_2 \geq \ldots \geq \theta_{K(J_{max})} \tag{7.3}$$

and the stronger the locality property is, the greater the inequalities are. $\Delta_{K(J_{max})}$ is thereby negative for all values of $K(J_{max})$ with the well known consequence that the *page demand rate is a decreasing function of the primary memory space guaranteed to each multiprogrammed program*. The product

$$\prod_{n=1}^{K(J_{max})} \frac{\theta_n}{\theta_n+\mu}, \tag{7.4}$$

decreases as $K(J_{max})$ increases. This decrease is *more than exponential* owing to inequalitites (7.3). $|\Delta_{K(J_{max})}|$ decreases even faster than this product when $K(J_{max})$ increases, strong evidence being given by figure 7.3 (and other studies, see e.g. [Va67]), that the decreasing function $\theta_n$ is strictly concave since for its inverse $y_n$:

$$\forall n, y_n > y$$

where y is a positive branch of hyperbola; hence

$$|\theta_2-\theta_1| \geq |\theta_3-\theta_2| \geq \ldots |\theta_{K(J_{max})+1}-\theta_{K(J_{max})}|.$$

Thus *the page demand rate increases all the more steeply when the main memory space guaranteed to each program shrinks as this space allotment is small.*

The product (7.4) and thus the decrement $\Delta_{K(J_{max})}$ increase also as $\mu \to 0$, *this influence of* $\mu$ *growing exponentially with* $K(J_{max})$.

On the other hand, it results from (7.1) that $\mu \times \partial(J_{max},\mu) \rightarrow \theta_{K(J_{max})}$ as $\mu \rightarrow 0$ and $\mu \times \partial(J_{max},\mu) \rightarrow \mu$ as $\mu \rightarrow \infty$, where, owing to inequalities (7.3), $\theta_{K(J_{max})}$ is a minimum for any fixed value of $K(J_{max})$.

So, for any value $K(J_{max})$ of the main memory allotment the page demand rate approaches a minimum $\theta_{K(J_{max})}$ as the average execution interval is increased until it becomes equal to the average processor time consumed per user programs. When the average execution interval decreases ($\mu \rightarrow \infty$), the page demand rate finishes by growing like $\mu$.

7.5.5. These tendencies of the page demand rate in function of $K(J_{max})$ and $(\mu)^{-1}$ are illustrated by figures 7.4 and 7.5. These figures reveal that from the point of view of the page demand rate, there would not be much to be gained from guaranteeing each program a parachor (*) of more than about seven page frames in $M_0$ and an execution interval longer than 0.2 seconds; conversely they show that a minimum execution interval of 0.05 seconds and a minimum parachor of four page frames are mandatory. It will be shown in § 7.9.2 that *these parachor values which are satisfactory for programs considered as individuals are far from optimizing performance criteria which take page transfer rates into account* . What is optimal for programs considered as individuals will not appear optimal when they are considered collectively.

---

* Parachor is a term in use to designate the amount of primary memory needed by a program to achieve some "satisfactory" level of performance [Be68] .

## 7.6 The Rate of Page Transfer Completions

Let J, $1 \leq J \leq J_{max}$, be the number of programs being multiprogrammed at some time t and $i_1'$, $0 \leq i_1' \leq J$, be the number of programs being in *suspended* state at this time t. It is shown in the appendix of [Sm67] that because of the simplifying assumptions 7.3.1 and 7.3.2, the rate at which page transfers from $M_1$ to $M_0$ are completed may be taken equal to

$$\mu_1(i_1') = \frac{2i_1'}{C_1+2i_1'-1} \times \frac{C_1}{T_{rot}} \tag{7.5}$$

This rate is obviously null for $i_1' = 0$, and approaches asymptotically $\frac{C_1}{T_{rot}}$ as $i_1'$ increases (see figure 7.2).

## 7.7 Conditions for Aggregation

It results from the simplifying assumption made in §7.3 and 7.5 that the hypothetical system may be regarded as a closed multi-queues system in which a fixed number N of programs cycle between three stages of service. $i_\ell$, $\ell$=0,1,2, being the number of programs at stage $\ell$, we may suppose that

| | |
|---|---|
| $i_0$ | is the number of programs in *ready* and/or *running* state; |
| $i_1 = i_1'+i_1''$ | where $i_1'$ is the number of programs in *suspended* state and $i_1''$ the number of programs waiting to be multiprogrammed; |
| $i_2$ | is the number of terminals without pending request for program execution. |

At any time t, t > 0, when J programs are being multiprogrammed, these variables must obey the following relations:

$$J = i_0+i_1', \quad J=1,\ldots,J_{max},$$
$$i_2 = N-(i_0+i_1'+i_1''),$$
$$(i_1''>0) \supset (i_0+i_1' = J_{max}),$$
$$(i_1''>0) \equiv ((N-i_2) > J_{max}).$$

Denoting $\mu_i p_{ij}$, $i,j=0,1,2$, the transfer rate from stage i to stage j, $j \neq i$, we have

$$\mu_0 p_{01} = \begin{cases} \mu \times \partial(J_{max},\mu) & \underline{\text{if}}\ i_1'' > 0 \quad (7.6) \\ \varphi \times \partial(J,\varphi) & \underline{\text{if}}\ i_1'' = 0 \quad (7.7) \end{cases}$$

$$\mu_0 p_{02} = \varphi \quad (7.8)$$

$$\mu_1 p_{10} = \mu_1(i_1') \quad \text{(see equation (7.2))} \quad (7.9)$$

$$\mu_2 p_{21} = \lambda \times i_2 \quad (7.10)$$

All other transfer rates being null, the condition (5.1) for near-complete-decomposability amounts in this case to:

For $i_2 = 0,\ldots,N-J_{max}$,

$$\lambda \times i_2 \ll \min_{1 \le i_1' \le J_{max}} [\mu_1(i_1'),(\mu \times \partial(J_{max},\mu))-\varphi]$$

and for $i_2 = N-J_{max}+1,\ldots,N-1$

$$\lambda \times i_2 \ll \min_{1 \le i_1', J \le J_{max}} [\mu_1(i_1'),\varphi \times (\partial(J,\varphi)-1)].$$

Since we have $\varphi \le \mu$, and $\varphi \times \partial(J,\varphi) \le \mu \times \partial(J_{max},\mu)$, $(J \le J_{max})$ as well as for $a < b$:

$$\varphi \times \partial(a,\varphi) \ge \varphi \times \partial(b,\varphi) \text{ and } \mu_1(a) \ge \mu_1(b),$$

a necessary and sufficient condition for (7.11) to be satisfied is

$$\lambda(N-1) \ll \min[\mu_1(1),\ \varphi\times(\partial(1,\varphi)-1)],$$

or, replacing $\mu_1(1)$ by its value yielded by (7.2):

$$\lambda(N-1) \ll \min\ [(\frac{2}{T_{rot}} \times \frac{c_1}{c_1+1}),\ \varphi\times(\partial(1,\varphi)-1)]. \qquad (7.12)$$

Condition (7.12) stipulates that the lowest possible rate of page transfers between primary and secondary memory be higher than the highest possible rate of interaction between the set of terminals and these memories. This is in fact a common state of affairs and we may reasonably expect to see this condition satisfied by most paging computer systems.

Moreover, Condition 7.12 is strictly expressed in terms of the hardware parameters $N, c_1$, $T_{rot}$, the user reaction time $(\lambda)^{-1}$, and the characteristics of a *non-multiprogrammed* user program execution, namely:

$(\varphi)^{-1}$, the average processor time consumed per execution;

$\partial(1,\varphi)$, the average number of page faults per execution if the entire primary memory space $C_0$ is available; relation (7.1) completely determines $\partial(1,\varphi)$ in function of $\varphi$, $C_0$, and the user program activity $\{\xi_n\}$, $n=1,\ldots$ .

A prior knowledge of these hardware and load parameters therefore, makes the verification of near-decomposability possible and may give a prior estimation of the precision achievable by using an aggregative model. In this particular case, given the values assigned in § 7.4 to $\varphi$, $\lambda$, $T_{rot}$ and $c_1$, and since $\partial(1,\varphi) \sim 19$, (7.12) is satisfied as long as $N \ll 1+\lambda^{-1}\times\varphi\times(\partial(1,\varphi)-1) \simeq 300$. Under this condition

the hypothetical system is nearly-completely-decomposable into the set of user terminals on the one hand, and a primary-secondary memory aggregate on the other hand. The short-run equilibrium attained by the aggregate may be studied independently of the interactions with the user terminals; the long run equilibrium attained by these interactions may be studied in terms of the equilibrium characteristics of the aggregate.

## 7.8 Analysis of the Aggregates $\mathfrak{M}_1(J)$, $\mathfrak{M}^*(J_{max})$

Throughout this short-run equilibrium analysis we may therefore disregard the activity of the user terminals. We may simply assume that a fixed number $(N-i_2)$, $0 \le i_2 \le N-1$, of requests for program execution are pending. This implies that a fixed number J of programs, $J = \min[(N-i_2), J_{max}]$ are being permanently multiprogrammed. It was shown in §7.5.1 and 7.5.2 that depending on whether or not $(N-i_2) > J_{max}$, the page demand rate was different. For this reason we consider two types of aggregates: A *saturated* aggregate, denoted $\mathfrak{M}^*(J_{max})$, in which the page demand rate is equal to $\mu \times \partial(J_{max}, \mu)$, and a *non-saturated* aggregate $\mathfrak{M}(J)$, $J=1,\ldots,J_{max}$, in which the page demand rate is equal to $\varphi \times \partial(J, \varphi)$.

Considering this latter case first, a probability distribution $\pi_2(n_0|J)$, $n_0=0,\ldots,J$, $J=1,\ldots,J_{max}$, of $n_0$ programs being in ready or running state on condition that J programs are being multiprogrammed may be obtained as the distribution of the congestion in a $\mathfrak{M}_1(J)$ queueing system, whose input rate is the page transfer rate and whose service rate is the page demand rate $\varphi \times \partial(J, \varphi)$. Since the page transfer rate is dependent on the congestion, probabilities $\pi_2(n_0|J)$ are defined by relations (5.3). Introducing into (5.3) the rates yielded by (7.9) and (7.7) yields for $J=1,\ldots,J_{max}$:

$$\pi_1(n_0|J) = \frac{\prod_{k=J}^{J-n_0+1} \mu_1(k)}{[\varphi \times \partial(J,\partial]^{n_0}} \pi_1(0|J), \quad n_0=1,\ldots,J$$

$$\text{with} \quad \pi_1(0|J) = \left\{ 1 + \sum_{n_0=1}^{J} \frac{\prod_{k=J}^{J-n_0+1} \mu_1(k)}{[\varphi \times \partial(J,\varphi)]^{n_0}} \right\}^{-1} .$$

Next, replacing $[\varphi \times \partial(J,\varphi)]$ by $[\mu \times \partial(J_{max},\mu)]$, yields a similar distribution for $\mathfrak{M}^*(J_{max})$ which will be designated $\{\pi_1^*(n_0|J_{max})\}_{n_0=0}^{J_{max}}$.

In particular,

$$\sigma_1(J) = 1-\pi_1(o|J), \quad J=1,\ldots,J_{max},$$

or in the saturation case

$$\sigma_1^*(J_{max}) = 1-\pi_1^*(0|J_{max}),$$

is the probability of the processor not being idle when J (or $J_{max}$) programs are being multiprogrammed. This probability will be henceforth referred to as the *processor efficiency*.

In figure 7.6, $\sigma_1^*(J_{max})$ is tabulated for $1 \le J_{max} \le 20$, i.e., $2 \le K(J_{max}) \le 48$, and for four values of the average execution interval $[\mu]^{-1}$.

### 7.8.1 Optimal Maximum Degree of Multiprogramming.

Figure 7.6 reveals that distinct optimal maximum degrees of multiprogramming, say $J_{max}^{opt}$, correspond to distinct values of the execution interval. $J_{max}^{opt}$ maximizes the processor efficiency $\sigma_1^*(J_{max})$, i.e., maximizes the sum

$$\sum_{n_0=1}^{J_{max}} \frac{\prod_{k=J_{max}}^{J-n_0+1} \mu_1(k)}{[\mu \times \partial(J_{max},\mu)]^{n_0}} .$$

An essential property of near-decomposability is that the relative values of the short-run equilibria attained in each system $\mathfrak{M}_1(J_{max})$, $J_{max}=1,2,\ldots$, are maintained in the long-run. Therefore, $J_{max}^{opt}$ is also the maximum degree of multiprogramming which maximizes the *long-run* fraction of time the processor is busy when the system is saturated.

The existence and the value of this optimum depend on the relative importance of two antagonistic effects: as the degree of multiprogramming increases, processor efficiency

(i) tends to increase the probability of at least one program being in ready state,

(ii) tends to decrease owing to the increase of the page fault rate which results from the reduction of primary memory space allotted to each program.

A method to evaluate such an optimum has already been proposed in [Wa69] but it applies to the simpler case of identically distributed channel service times and a geometrically distributed number of page faults per program execution; thus the locality property of programs is not taken into account.

Figure 7.6 reveals also that *the optimal maximum degree of multiprogramming increases as the average execution interval* $[\mu]^{-1}$ *decreases*. Indeed, the second counteracting effect does not become operative until programs have exhausted their primary memory allotment $K(J_{max})$. The shorter the execution interval, the smaller this exhaustible space allotment becomes.

### 7.8.2. Thrashing

It is evident from figure 7.6 that the primary memory allotment which must be guaranteed to each program in order to achieve maximal processor efficiency is considerably larger than the parachor value evaluated in §7.5.5.

The difference is justified by the fact that the parachor is an individual program characteristic whereas $\sigma_1(J)$ and $\sigma_1^*(J_{max})$ are measures of the overlapping achieved between page transfers and processor busy periods and depend on secondary memory transfer rates as well.

P. J. Denning's argumentation on trashing [De68|2] helps us to understand why the parachor is smaller than $K(J_{max}^{opt})$. A measure e of the ability of a program to use the processor may be defined as

$$e = \frac{\mu^{-1}}{\mu^{-1} + \partial(J_{max},\mu) \times (\bar{\mu}_1)^{-1}} ,$$

where

$$\bar{\mu}_1 = \sum_{n_0=0}^{J_{max}-1} \pi_1^*(n_0|J_{max})\ \mu_1(J_{max}-n_0)$$

is the average service rate of page transfer requests; $[\partial(J_{max},\mu) \times (\bar{\mu}_1)^{-1}]$ is thus the average time a program spends waiting for page transfers during an execution interval. Denoting by m the page transfer rate, we obtain

$$e = \frac{1}{1 + m \times (\bar{\mu}_1)^{-1}} .$$

The slope of e in function of m is

$$\frac{de}{dm} = - \frac{(\bar{\mu}_1)^{-1}}{[1 + m \times (\bar{\mu}_1)^{-1}]^2} ,$$

which means that e is more sensitive to m variations as $(\bar{\mu}_1)^{-1}$ is great and m small. Therefore, the larger $(\bar{\mu}_1)^{-1}$ is, the larger $K(J_{max}^{opt})$ and the smaller $J_{max}^{opt}$. Large $(\bar{\mu}_1)^{-1}$ values are responsible for $K(J_{max}^{opt})$ exceeding the parachor.

This extreme sensitivity of processor efficiency for large $(\bar{\mu}_1)^{-1}$ to m variations is responsible for serious degradations of the system response time when the degree of multiprogramming is allowed to exceed $J_{max}^{opt}$. These degradations are analyzed in paragraphs 7.12 and 7.13.

### 7.8.3. Processor Efficiency Versus Execution Intervals

Figure 7.6 shows that *the longer the mean execution interval is, the smaller the primary memory space* $K(J_{max})$ *required to attain a specified processor efficiency is*. Or, that the larger $K(J_{max})$ is, the more advantageous it is from the point of view of processor efficiency to provide for the longest possible execution intervals. This is a consequence of the increasingly preponderant influence $(\mu)^{-1}$ has on the page fault rate decrement $\Delta_{K(J_{max})}$, (cf. §7.5.4) as $K(J_{max})$ increases, owing to the property of locality of references (inequalities 7.3).

Therefore a policy which optimizes processor efficiency by taking advantage of the locality property, should provide the user program with the longest possible execution intervals. Such a policy gives most effective results when the primary memory allotment per program is large, viz. the degree of multiprogramming small. This is a supplementary argument in

favor of processor allocation methods which have already been advocated [Mu70] to minimize changes of tasks (and thereby reduce overheads on processors) when the system is underloaded, viz. when the number of outstanding requests for program execution is less than the number of quanta contained within the prerequisite response time.

One may also observe in figure 7.6 that $\sigma_1^*(J_{max})$ is practically insensitive to $J_{max}$ if $(\mu)^{-1}$ is so small, in this example less than about $\sum_{i=1}^{3} \theta_i^{-1} \cong 430\ \mu s$, so that programs are not given enough time, whatever $K(J_{max})$ is, to accumulate in $M_0$ the minimum number of pages necessary for the property of locality to have effective results (about four pages as shown in figure 7.4).

## 7.9 The Long-Run Equilibrium

The entire system may now be regarded as a set of N active user terminals originating requests for program execution to the aggregates $\mathfrak{M}_1(j)$, $J=1,\ldots,J_{max}$, and $\mathfrak{M}_1^*(J_{max})$. Let $n_1 = (N-i_2)$ be the number of pending requests, i.e., the number of programs being or waiting to be multiprogrammed. We will refer to $n_1$ as being the *system congestion*.

We may consider that the programs are executed by a single aggregate $\mathfrak{M}_1(n_1)$ such that

$$\mathfrak{M}_1(n_1) \equiv \mathfrak{M}_1(J), \qquad \text{for } n_1 = J \le J_{max},$$

$$\mathfrak{M}_1(n_1) \equiv \mathfrak{M}_1^*(J_{max}), \quad \text{for } n_1 > J_{max}.$$

The rate at which programs are completed by $\mathfrak{M}_1(n_1)$ is

$$\psi_{1,2}(n_1) = \begin{cases} \varphi[1-\pi_1(0|n_1)] = \varphi x \sigma_1(n_1), & \text{if } n_1 \le J_{max} \\ \varphi[1-\pi_1^*(0|J_{max})] = \varphi x \sigma_1^*(J_{max}) & \text{if } n_1 > J_{max} \end{cases} . \tag{7.13}$$

$\psi_{1,2}(n_1)$ will be referred to as the *system service rate*. $\psi_{1,2}(n_1)$ is displayed in figure 7.8.

The rate at which requests are originated at some time t when the system congestion is equal to $n_1$ is equal to

$$\mu_2 P_{21} = \lambda(N-n_1), \quad n_1=0,\ldots,N \tag{7.14}$$

Introducing (7.13) and (7.14) into (4.7.1) yields the long-run equilibrium probability distribution $\{\pi_2(n_1|N)\}_{n_1=0}^{N}$ of the system congestion:

$$\pi_2(n_1|N) = \lambda^{n_1} \frac{N(N-1)\ldots(N-n_1+1)}{\prod_{k=1}^{n_1} \psi_{1,2}(k)}, \quad n_1=1,\ldots,N$$

where $$\pi_2(0|N) = \left\{ 1 + \sum_{n_1=1}^{N} \lambda^{n_1} \frac{N(N-1)\ldots(N-n_1+1)}{\prod_{k=1}^{n_1} \psi_{1,2}(k)} \right\}^{-1}$$

## 7.10 The System Congestion Avalanche-Like Effect

Assuming N=20, the distribution $\{\pi_2(n_1|N)\}_{n_1=0}^{N}$ has been calculated for various values of $J_{max}$ (see figure 7.7). Some representative values of the mean congestion

$$E = \sum_{n_1=1}^{N} n_1 \, \pi_2(n_1|N)$$

among those obtained are:

| $J_{max}$ | E |
|---|---|
| 1 | 12 |
| 2 | 7.49 |
| 3 | 5.25 |
| 4 | 5.71 |
| 10 | 15.7 |
| 20 | 19.1 |

These mean congestions, as well as the distributions of figure 7.7 suffice to advocate multiprogramming towards 'batch-processing' ($J_{max}=1$) and to advocate policies which control the maximum degree of multiprogramming so as to keep it around its optimal value ($J_{max}^{opt} \simeq 3$).

In particular, the last mean congestion in the list above, almost equal to N=20, indicates how the system may become <u>overcrowded</u> if all programs present are allowed to compete for primary memory ($J_{max}=20$). An intuitive interpretation of this behavior may be deduced from the shape of $\psi_{1,2}(n_1)$ as a function of $n_1$ for some given value of $J_{max}$ (cfr. figure 7.8): *once the congestion exceeds an optimal value which maximizes the system service rate* $\psi_{1,2}(n_1)$, *any increase of the congestion results in a service rate decrease which, in turn, accentuates the initial congestion increase.* The upper limit $J_{max}$ of the degree of multiprogramming acts as a barrier against this *avalanche-like effect*: the system service rate never decreases below $\varphi \times \sigma_1^*(J_{max})$, no matter how great the congestion becomes.

This avalanche-like behavior of the congestion is analyzed in greater detail in section 7.12.

## 7.11 The System Saturation Points

In figure 7.9, the mean system congestion E has been plotted against the number N of active user consoles ($1 \leq N \leq 20$) for various values of $J_{max}$.

As N increases, the mean congestion approaches and follows asymptotes of slope 1. This is due to the fact that, for increasing values of N, all other system parameters remaining constant:

$$\{\pi_2(n_1|N) \rightarrow 0\}_{n_1=0}^{J_{max}} ;$$

the system therefore tends to behave as a N-consoles system with constant service rate $[\varphi x \sigma^*(J_{max})]$. Kleinrock [17] has shown how, beyond a certain N value, such systems get saturated, each additional user interfering completely with all the other users and adding one more unit to the mean congestion. He has shown that the corresponding mean congestion asymptote of slope 1 intersects the line E=1 for a certain value of N which is precisely the average number of consoles the system may handle without experiencing any mutual delaying interference among the user requests; this number, which he defines as the saturation point of the system, is equal to the average number of user requests which may be satisfied until completion within a time interval equal to the sum of the average user reaction time and the average user request service time.

We may well verify that, in figure 7.9, the asymptotes intersect the line E=1 at point

$$N^*_{J_{max}} = \frac{[\varphi x \sigma_1^*(J_{max})]^{-1} + \lambda^{-1}}{[\varphi x \sigma_1^*(J_{max})]^{-1}} \qquad (7.15)$$

which would be "the saturation point" of the system if its service rate were not congestion-dependent but equal to $[\varphi \times \sigma_1^*(J_{max})]$. On the other hand, the behavior of the mean congestion in the domain of smaller N values is considerably different from that of a system with congestion-independent service rate. In these latter systems the mean congestion cannot increase more than linearly with N. The causes of the sharp non-linearities which may be observed in figure 7.8 are discussed in the next section where a definition of saturation for systems with a congestion-dependent service rate of the same shape as $\psi_{1,2}(n_1)$ is introduced.

## 7.12 The Congestion Stability

A peculiarity of the aggregate $\mathfrak{M}_1(n_1)$ is that both its input rate $\lambda(N-n_1)$ at which requests for program executions are originated and its service rate $\psi_{1,2}(n_1)$ at which these executions are completed are dependent on the system congestion $n_1$, $n_1=0,\ldots,N$. The general shape of these rates is displayed in function of $n_1$ on figure 7.10.

One observes that, depending on the relative values of N, $J_{max}$, $\lambda$, $\varphi$, $\sigma_1(n_1)$, there may be at most three congestion values $n_A$, $n_B$, $n_C$ for which the input equates the service rate, viz. which are solutions of

$$\lambda(N-n_1) = \psi_{1,2}(n_1), \tag{7.16}$$

where $\psi_{1,2}(n_1)$ is given by (7.13).

It is simple to prove that $n_A$ *and* $n_C$ *are stable congestions* around which the system is inclined to come into equilibrium. Let us consider the system at some instant when the congestion is equal to $n_A$. An increment $\Delta n_1$ of the congestion will cause the service rate to exceed the input rate.

This excess, which is greater to the extent that $\Delta n_1$ is large (provided $n_A + \Delta n_1 \leq n_{opt}$), will tend to reduce the congestion to its original value $n_A$. Likewise a decrement $-\Delta n_1$ would cause the input rate to exceed the output rate, compelling the congestion to re-increase. The same reasoning applies to the congestion $n_C$.

Inversely, a similar argumentation indicates that in the vicinity of $n_B$ the congestion variations are reinforced instead of being deadened by the alterations they cause to the output to input rate ratio . $n_B$ *is an unstable congestion*, i.e. a state the system will always be zealous to leave in favor of stable congestions in the vicinity of $n_1$ or $n_C$.

Depending on the relative values of the various sytem parameters, intersection B may or may not exist. In the first case the congestion will become preferentially steady around values near $n_A$ or $n_C$. This is well reflected by the shape of the long-run equilibrium distribution $\{\pi_2(n_1|N)\}_{n_1=0}^{N}$, figure 7.7. Comparing figure 7.7 and 7.8, it may be verified for $J_{max}=3$ that the two congestions of highest probability, $n_1 \simeq 1$ and $n_1 \simeq 6$, are respectively equal to $n_A$ and $n_C$, and that the congestion of least probability, viz. $n_1 \simeq 3$, is identical to $n_B$. Similar extremes, too small to be distinguishable on figure 7.7, affect the distributions corresponding to $J_{max} = 1,10,20$. It may be conjectured that the transient time behavior of the congestion in such systems is increasing *oscillation* between values in the vicinity of $n_A$ and values in the vicinity of $n_C$.

If no B intersection exists, $\{\pi_2(n_1|N)\}_{n_1=0}^{N}$ exhibits a single maximum equal either to $n_A$ or $n_C$, depending on the value of N. Compare for instance figure 7.8 with the distributions displayed on figure 7.11 for

$J_{max}=3$ and N=25,14 respectively.

The behavior of the congestion as a function of N, $0 \leq N \leq \infty$, may be summed up as follows. For N small, the congestion at first remains in the vicinity of $n_A$. As soon as N exceeds a certain value, denoted $N^{\dagger}_{J_{max}}$ and defined graphically in figure 7.10, B and C intersections exist and the mean congestion must reach abruptly a value between $n_A$ and $n_C$, $n_C \gg n_A$. This explains the sharp increase of E on figure 7.9, once N exceeds a value which is equal to $N^{\dagger}_{J_{max}}$. For N larger, the system tends to behave as a system with constant service rate and the congestion increases almost linearly with N as does $n_C$. The smaller $\sigma_1^*(J_{max})$ is, the closer to N is $n_C$, obviously. For $N \to \infty$, it is easy to show that, for a given value of $J_{max}$, $E \to n_C$. In the steady state we may equate the mean input and service rates; the mean input rate is

$$\lambda \sum_{n_1=0}^{N} \pi_2(n_1|N) \times (N-n_1) = \lambda(N-E);$$

Equating with the mean service rate $\sum_{n_1=1}^{N} \pi_2(n_1|N)\psi_{1,2}(n_1)$, we find:

$$E = N - \sum_{n_1=1}^{N} \pi_2(n_1|N)\frac{\psi_{1,2}(n_1)}{\lambda} . \tag{7.17}$$

For $N \to \infty$, the system tends to behave as a system of constant service rate $\psi_{1,2}(n_C)$ so that

$$\lim_{N\to\infty} E = N - [1 - \pi_2(0|N)]\frac{\psi_{1,2}(n_C)}{\lambda} \tag{7.18}$$

As $n_C$ is solution of equation (7.16), and $\pi_2(0|N) \to 0$ as $N \to \infty$, (7.18) becomes

$$\lim_{N\to\infty} E = n_C.$$

$N^{\dagger}_{J_{max}}$ is the *saturation point* of the system: the mean congestion increases less than linearly with N, for $N < N^{\dagger}_{J_{max}}$, and more than linearly for $N^{\dagger}_{J_{max}} < N < \infty$. $N^{\dagger}_{J_{max}}$ may easily be deduced for given values of the parameters which define the hypothetical system by constructing an equilibrium diagram of the type defined in figure 7.10.

## 7.13 The System Response Time

The mean response time of the system is given by relation (4.70):

$$\bar{W} = [\lambda(1 - \pi_2(N|N]^{-1} \times E.$$

It is the mean time spent by a program in the system, waiting for or being multiprogrammed. $\bar{W}$ is plotted in function of N for $J_{max}$=3,10,20, in figure 7.12. Sharp non-linearities may be observed as soon as N exceeds the saturation point $N^{\dagger}_{J_{max}}$. For N larger, the system tends to behave as if the service rate were congestion-independent so that the limit of the slope of $\bar{W}$ with N is

$$\lim_{N\to\infty} \frac{d\bar{W}}{dN} = [\varphi \times \sigma_1^*(J_{max})]^{-1};$$

the smaller $\sigma_1^*(J_{max})$, the steeper this boundary slope is.

These sharp increases of the congestion and of the response time when the number of active terminals exceeds the saturation point are the consequences of the extreme sensitivity of the processor efficiency (see section

7.92 on trashing) to fluctuations of the page demand rate when the page transfer rate is slow and the degree of multiprogramming is allowed to exceed $J_{max}^{opt}$.

The fluctuations of $\bar{W}$ in function of $J_{max}$ for N fixed are displayed in figure 7.13. The discontinuities for $J_{max}$=7,13 correspond to $\sigma_1^*(J_{max})$ minimae (cfr. figure 7.6).

By virtue of the property of near-decomposability we have been able to specify within a single analytical model the condition of existence of thrashing in terms of parameters defining (i) the traffic of pages between primary and secondary memory and (ii) the load on the system, and to measure the consequences of this phenomenon on overall system performances such as the congestion and the response time.

## 7.14 Long-Run and Unconditional Distributions

The distributions $\{s_0(i_0)\}_{i_0=0}^{J_{max}}$, $\{s_1(i_1')\}_{i_1'=0}^{J_{max}}$ and $\{s_1(i_1'')\}_{i_1''=0}^{N-J_{max}}$ may be obtained from the system congestion $\{\pi_2(n_1|N)\}_{n_1=0}^{N}$ by means fo relations (4.8). More precisely, the long-run and unconditional probability of $i_o$ programs in *ready* or *running* state is

$$s_0(i_0) = \sum_{n_1=i_0}^{J_{max}} \pi_2(n_1|N)\pi_1(i_0|n_1)$$

$$+ \pi_1^*(i_0|J_{max}) \sum_{n_1=J_{max}+1}^{N} \pi_2(n_1|N), \quad i_0=0,\ldots,J_{max};$$

The long-run and unconditional probability of $i_1'$ programs in suspended state is:

$$s_1(i_1') = \sum_{n_1=i_1'}^{J_{max}} \pi_2(n_1|N)\pi_1(n_1-i_1'|N)$$

$$+ \pi_1^*(J_{max}-i'_1|J_{max}) \sum_{n_1=J_{max}+1}^{N} \pi_2(n_1|N)$$

$$i_1'=0,\ldots,J_{max};$$

The long-run and unconditional probability of $i_1''$ programs waiting to be multiprogrammed is simply $\pi_2(J_{max}+i_1''|N)$, $i_1''=0,\ldots,N-J_{max}$.

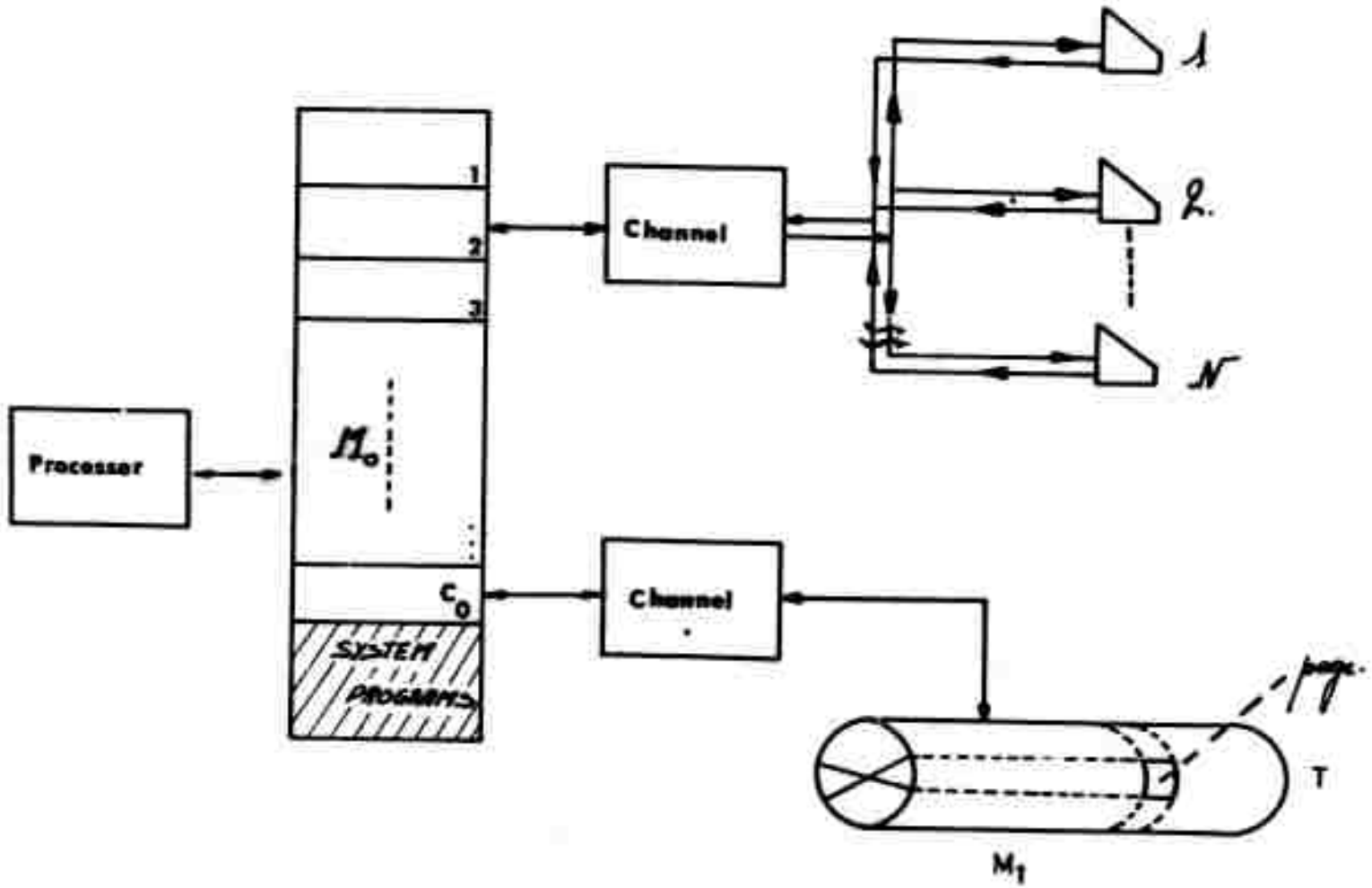

Processor
1
2
3
$M_0$
$C_0$
SYSTEM PROGRAMS
Channel
Channel
1
2
N
T
$M_1$

FIG. 7.1

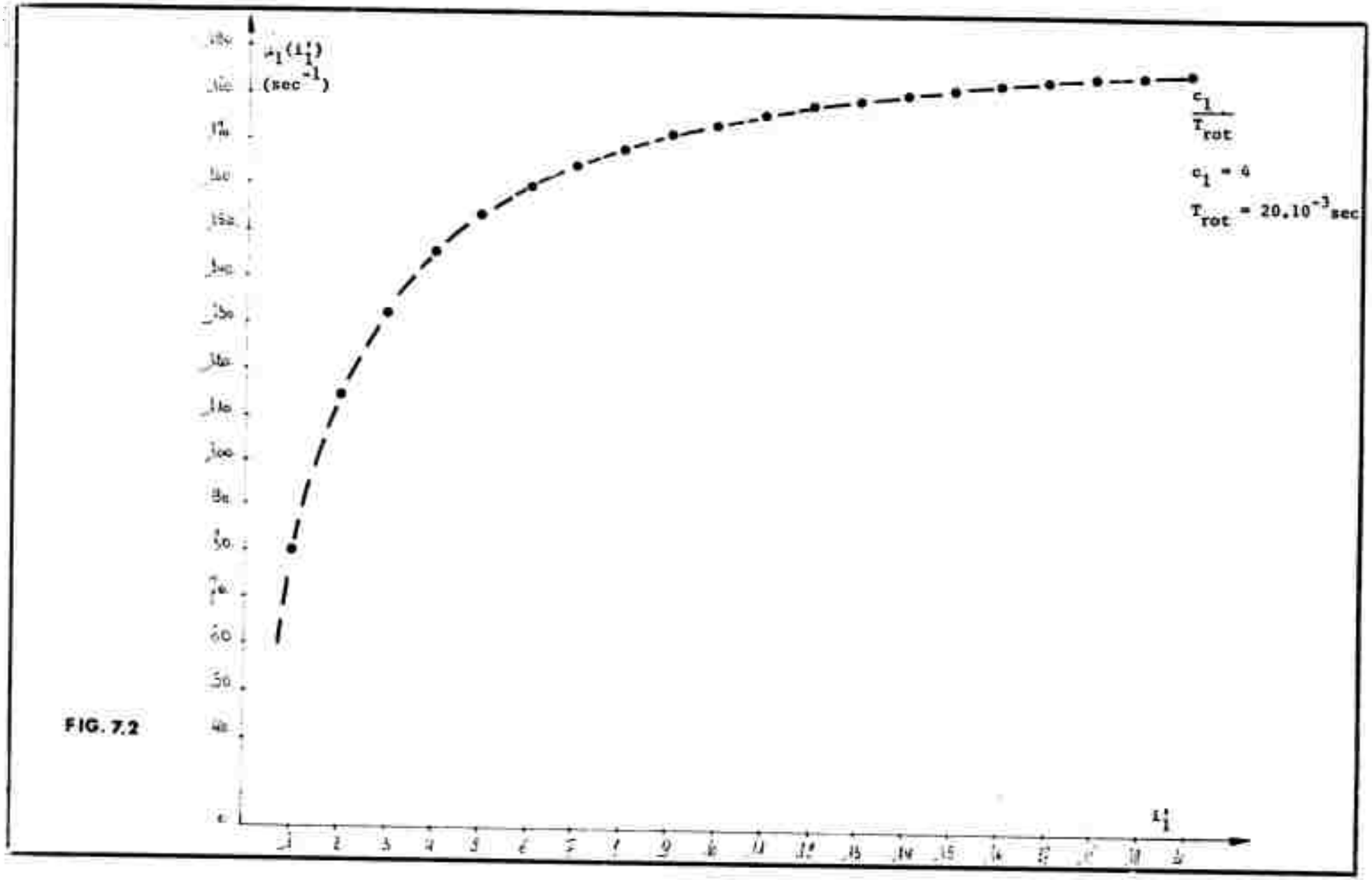



FIG. 7.2

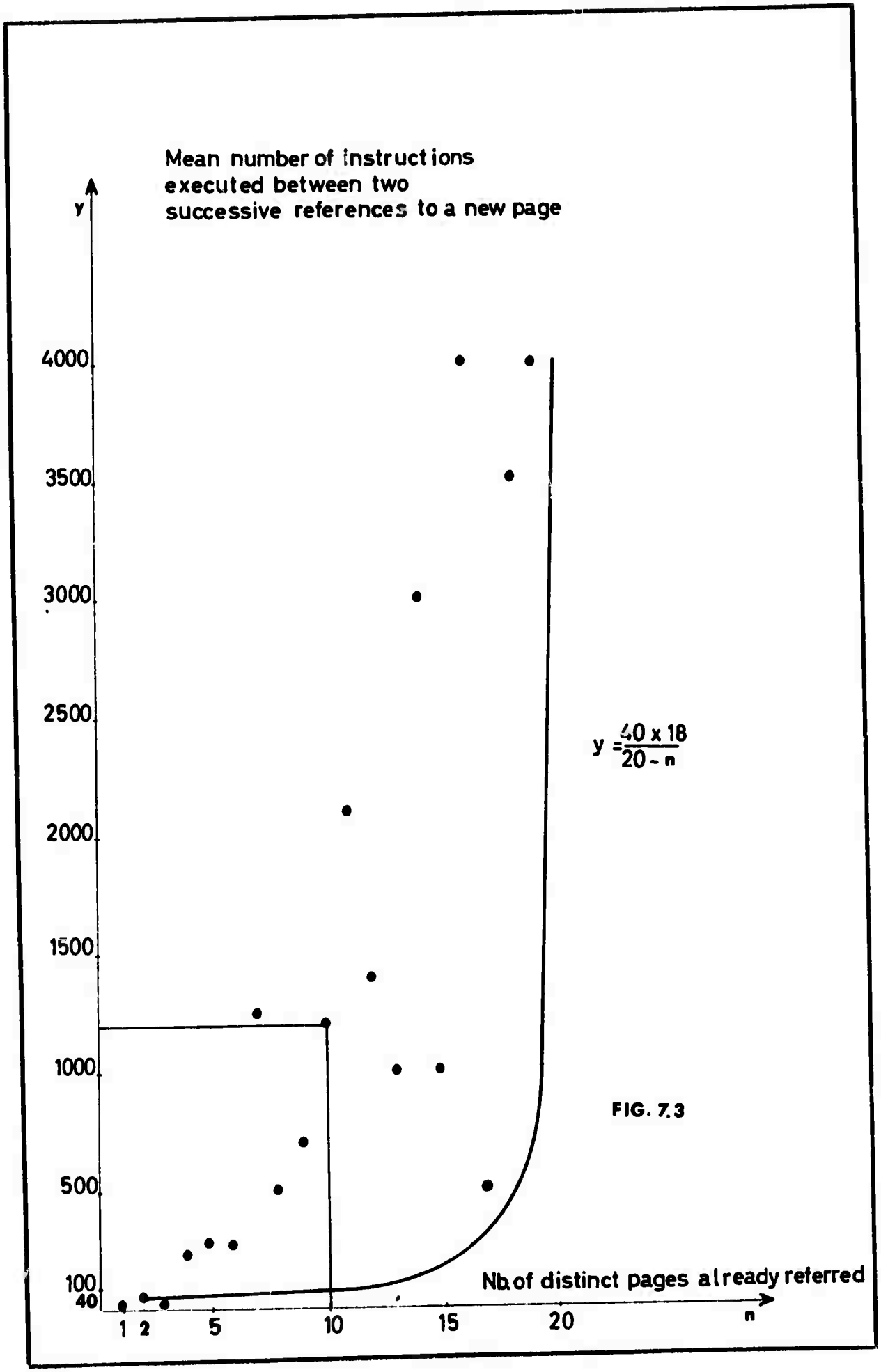



FIG. 7.3

972100

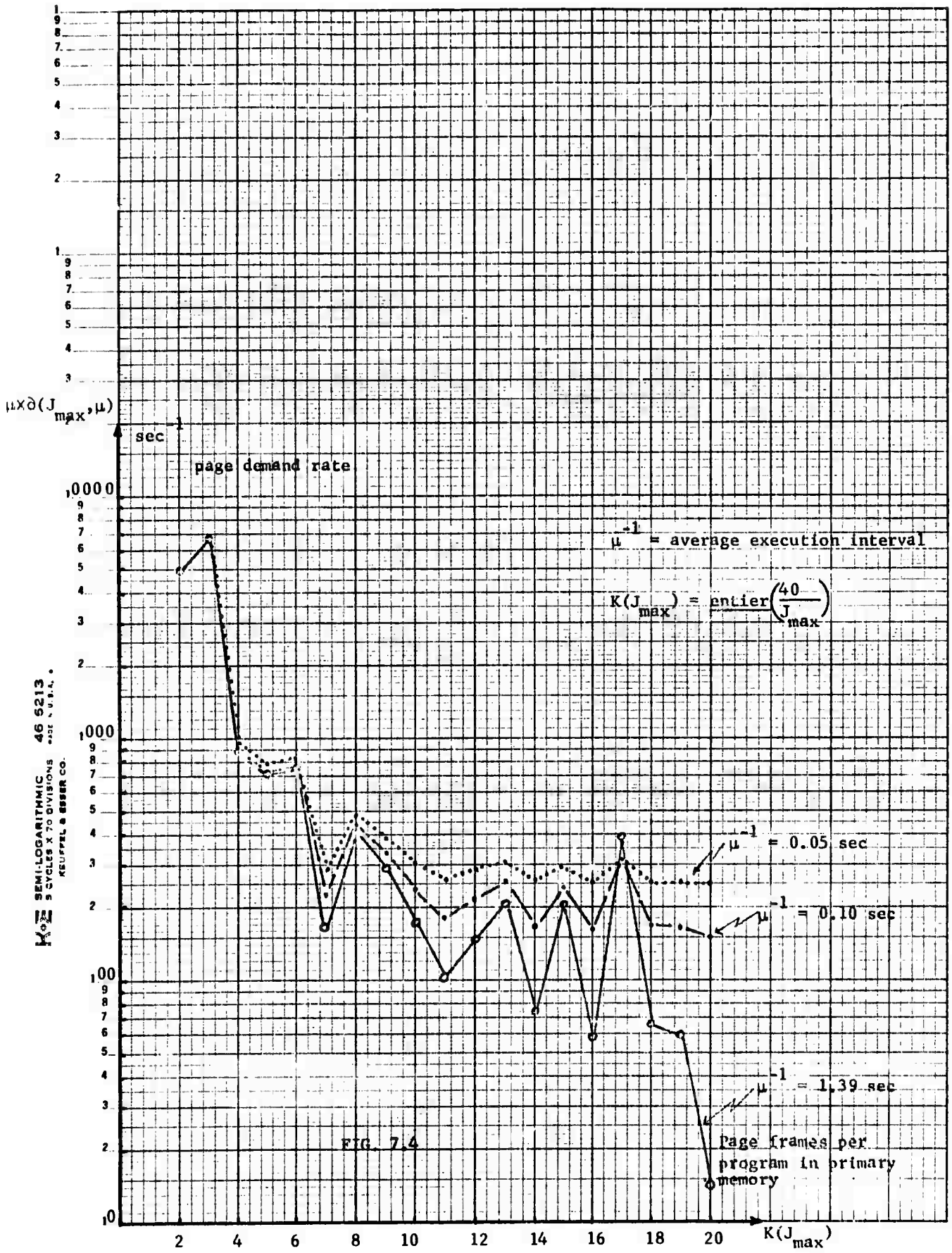

μxδ(J_max, μ)
sec^-1
page demand rate
μ^-1 = average execution interval
K(J_max) = entier(40/J_max)
μ^-1 = 0.05 sec
μ^-1 = 0.10 sec
μ^-1 = 1.39 sec
Page frames per program in primary memory
K(J_max)
10000
1000
100
10
2
4
6
8
10
12
14
16
18
20
SEMI-LOGARITHMIC 46 5213
5 CYCLES X 70 DIVISIONS
KEUFFEL & ESSER CO.

FIG. 7.4

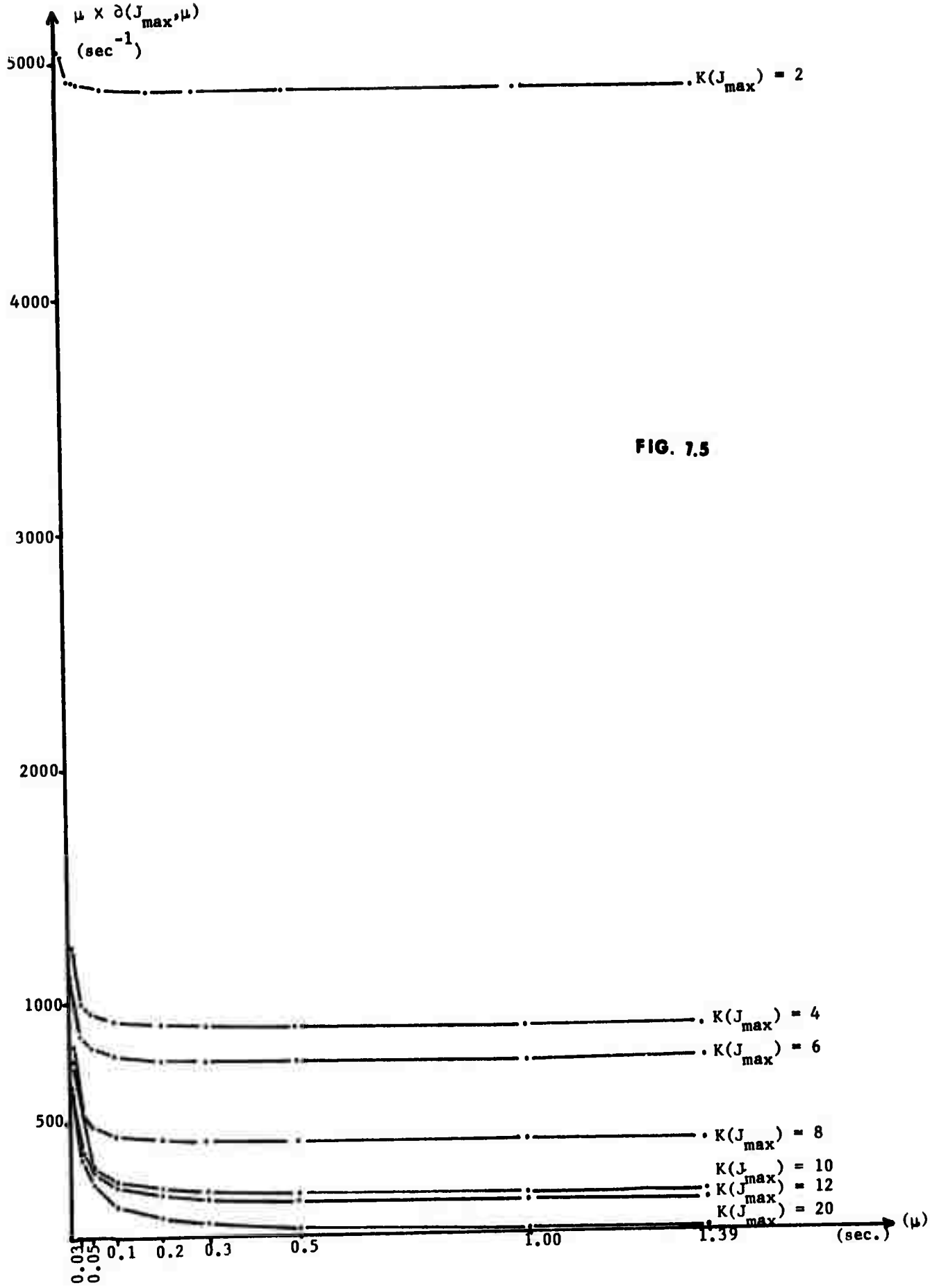

μ x ∂(J_max,μ)
(sec^-1)
5000
4000
3000
2000
1000
500
K(J_max) = 2
K(J_max) = 4
K(J_max) = 6
K(J_max) = 8
K(J_max) = 10
K(J_max) = 12
K(J_max) = 20
0.03
0.05
0.1
0.2
0.3
0.5
1.00
1.39
(sec.)
(μ)

FIG. 7.5

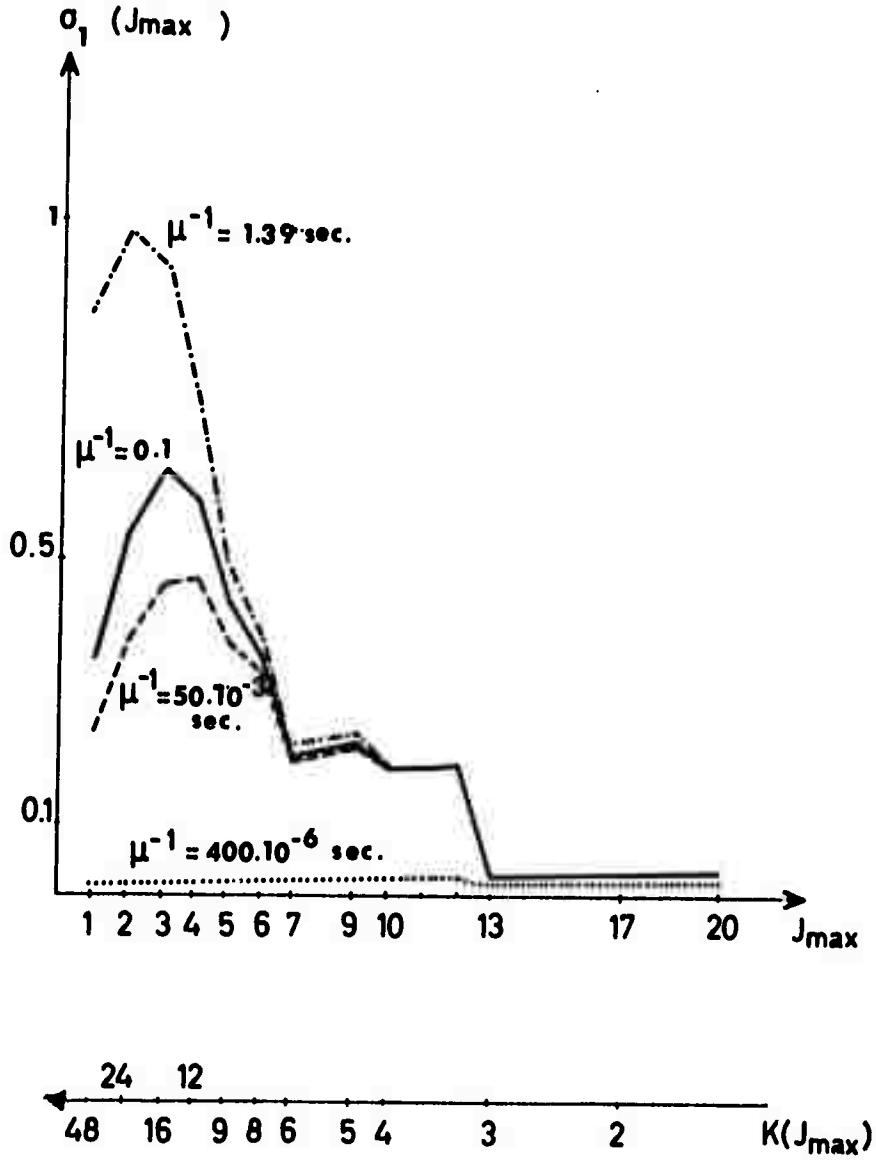

σ1 (Jmax )
1
0.5
0.1
μ-1 = 1.39 sec.
μ-1 = 0.1
μ-1 = 50.10-3 sec.
μ-1 = 400.10-6 sec.
1 2 3 4 5 6 7 9 10 13 17 20
Jmax
24 12
48 16 9 8 6 5 4 3 2
K(Jmax)

FIG. 7.6

| $J_{max}$ | Mean |
|---|---|
| 20 | 19.1 |
| 10 | 15.7 |
| 3 | 5.25 |
| 1 | 12.0 |

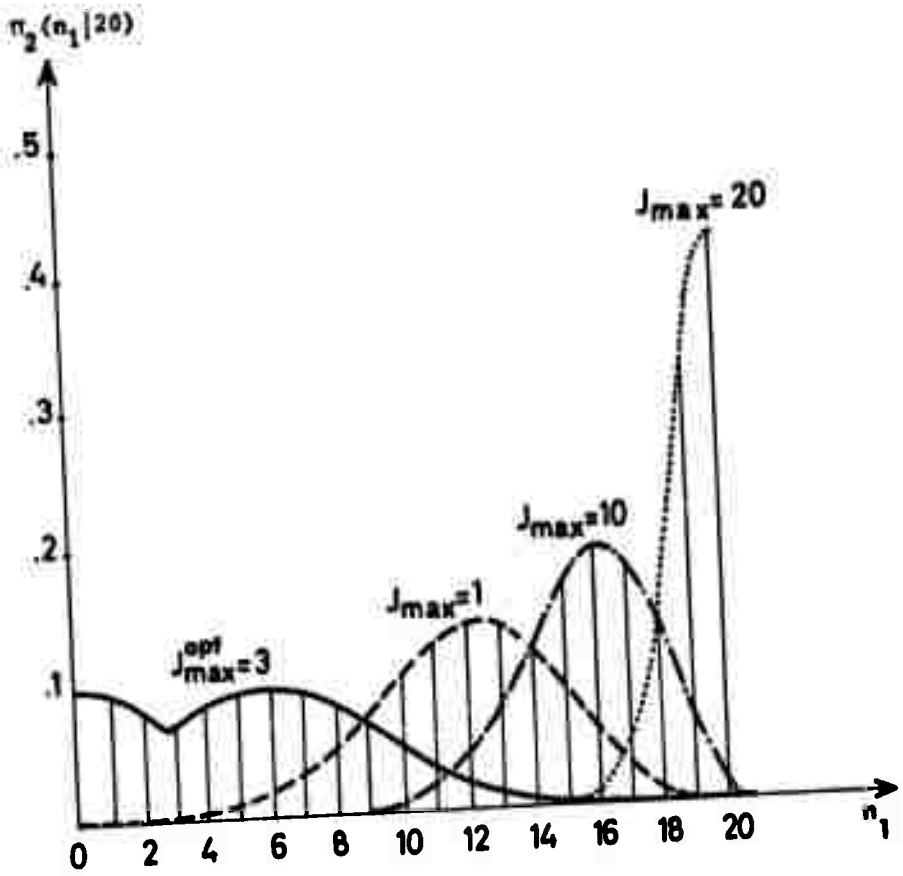



FIG. 7.7

972100

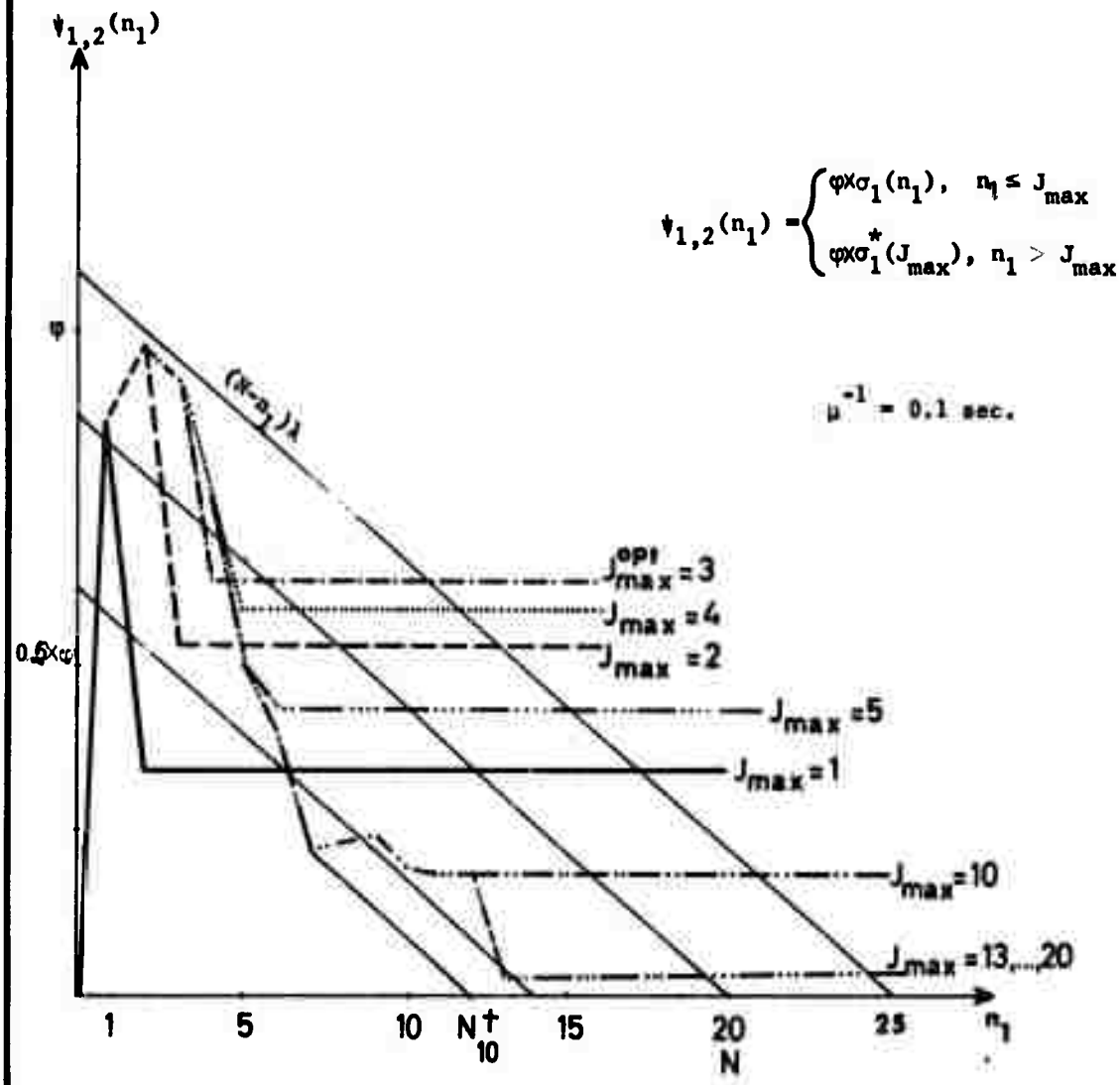
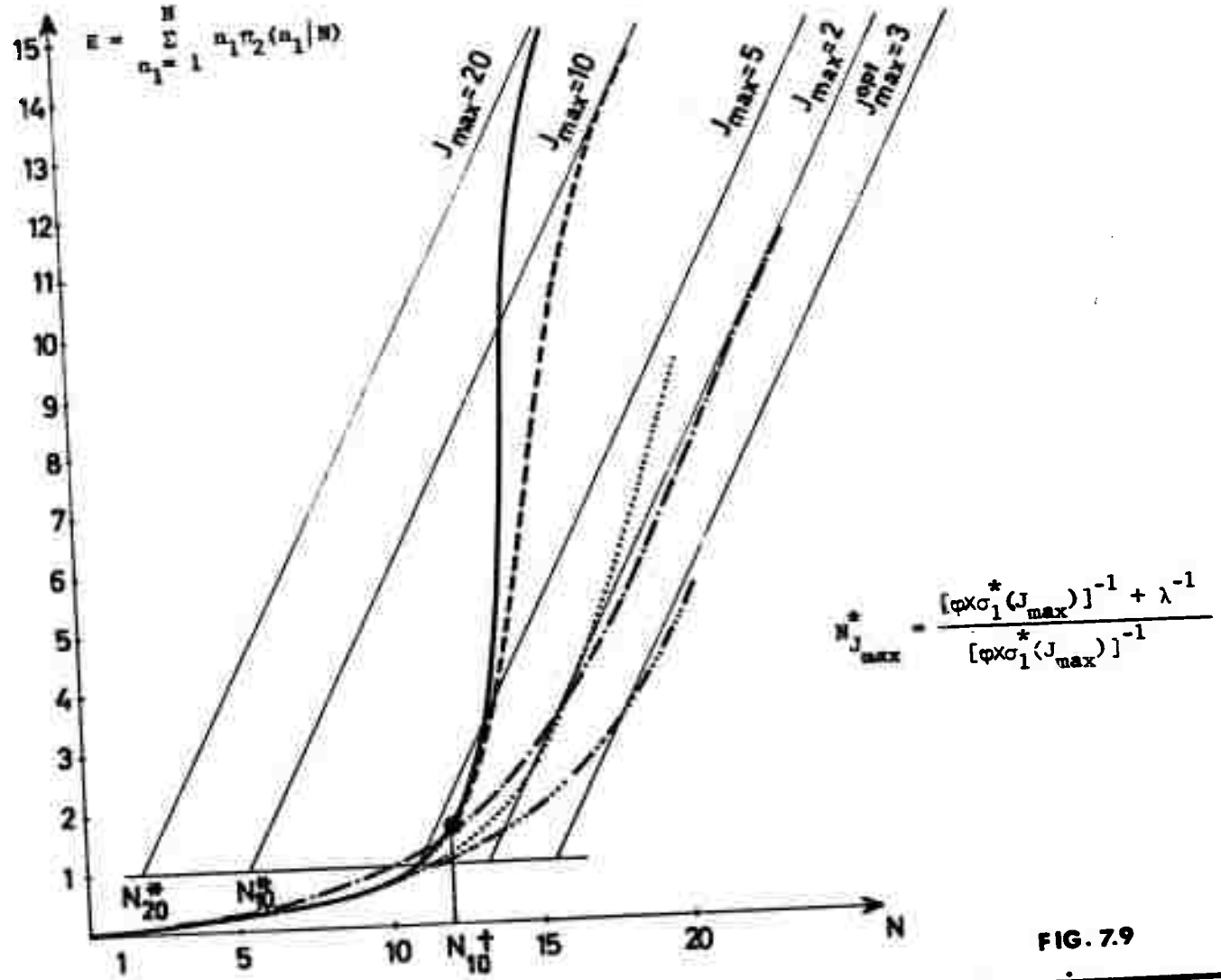



FIG. 7.8

972100




$$N^*_{J_{max}} = \frac{[\varphi x \sigma_1^*(J_{max})]^{-1} + \lambda^{-1}}{[\varphi x \sigma_1^*(J_{max})]^{-1}}$$

FIG. 7.9

9/2100

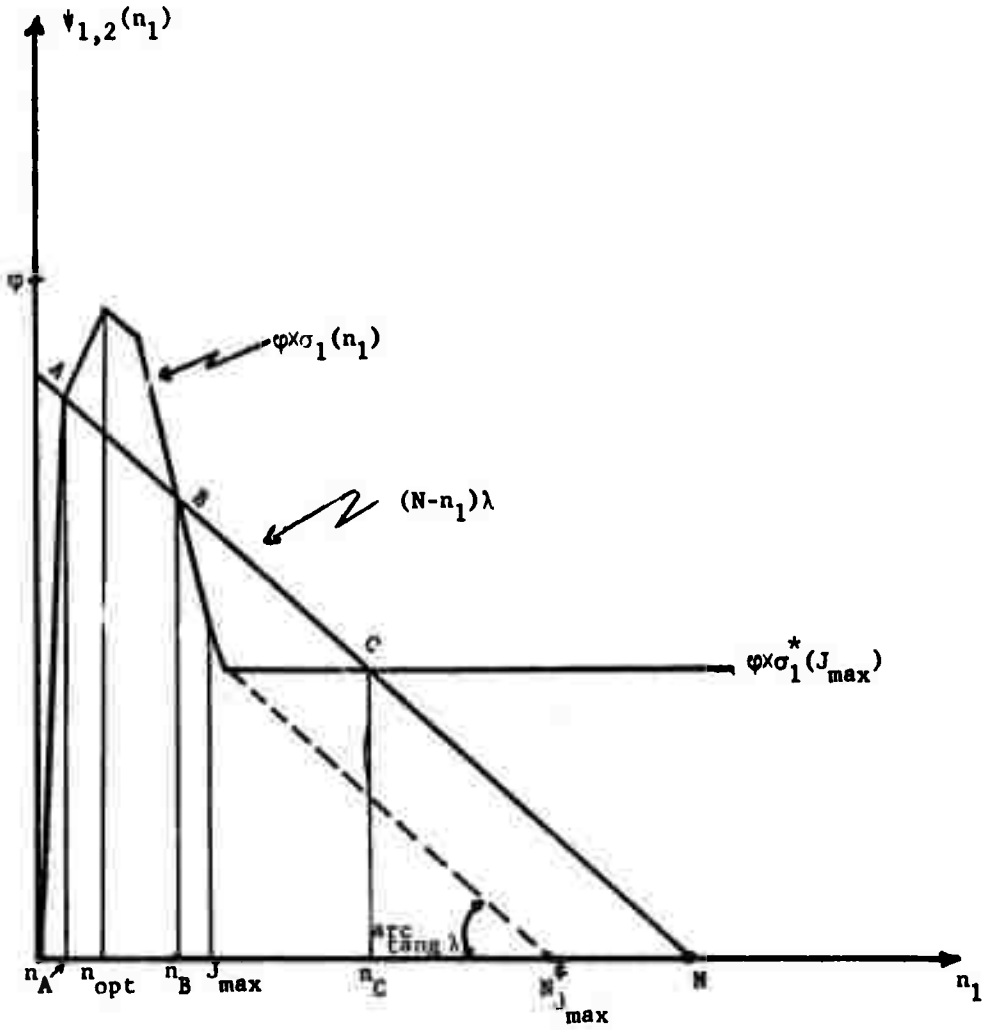



FIG. 7.10

972100

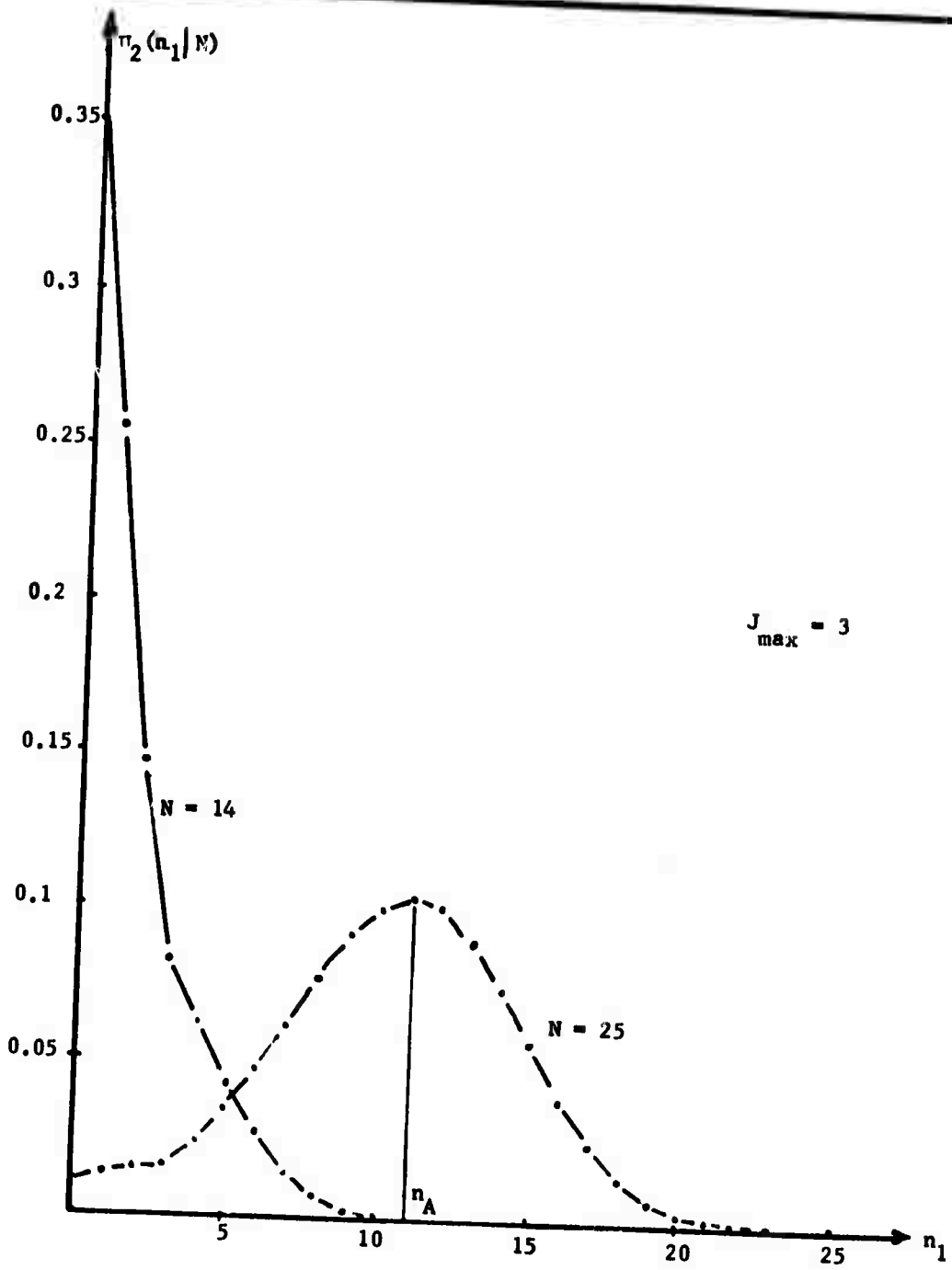



FIG. 7.11

972100

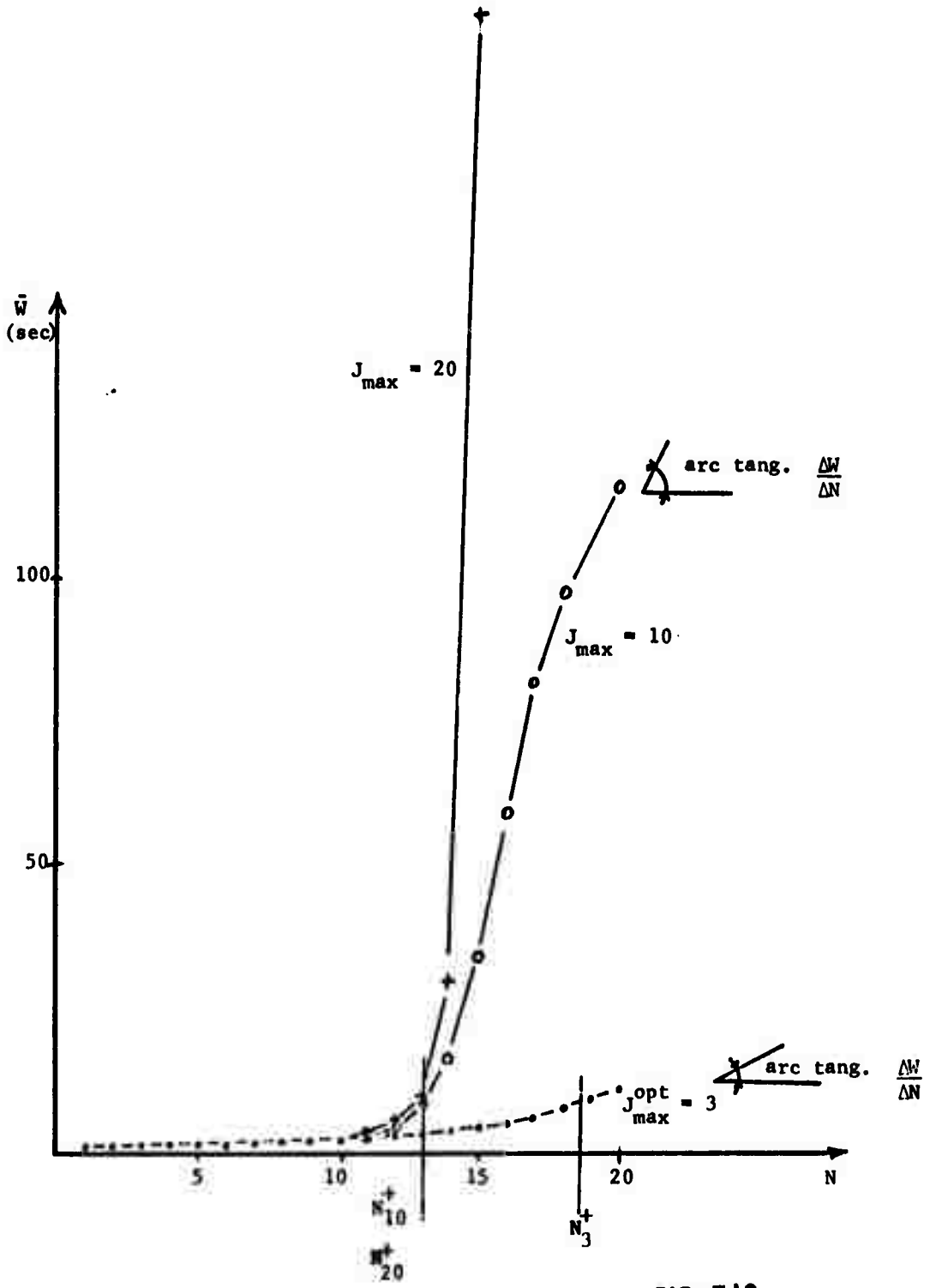

$\bar{W}$ (sec)
100
50
$J_{max}$ = 20
arc tang. $\frac{\Delta W}{\Delta N}$
$J_{max}$ = 10
arc tang. $\frac{\Delta W}{\Delta N}$
$J^{opt}_{max}$ = 3
5
10
15
20
N
$N^+_{10}$
$N^+_{20}$
$N^+_3$

FIG. 7.12

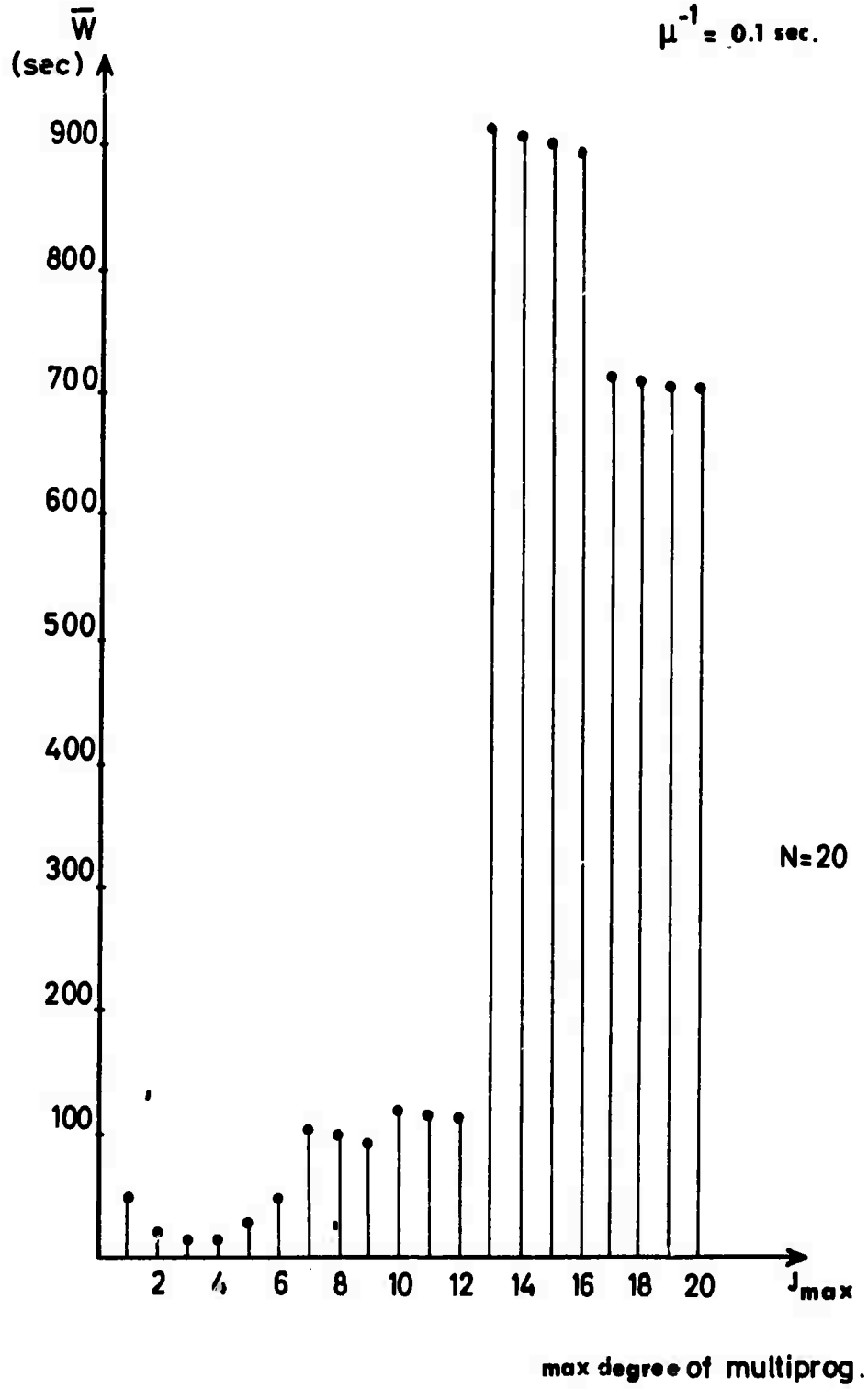

$\overline{W}$
(sec)
$\mu^{-1}$ = 0.1 sec.
900
800
700
600
500
400
300
200
100
N=20
2
4
6
8
10
12
14
16
18
20
$J_{max}$
max degree of multiprog.

FIG. 7.13

## REFERENCES


Al 64 C. ALEXANDER. Notes on the Synthesis of Form, Harvard University Press, Cambridge, Mass., 1964.

An 63 A. ANDO, F. M. FISHER. "Near-Decomposability, Partition and Aggregation, and the Relevance of Stability Discussions." International Economic Review, Vol. 4, No. 1, January 1963. Reprinted in Essays on the Structure of Social Science Models by A. Ando, F. M. Fisher, H. A. Simon.

Ar 71 S. R. ARORA, A. GALLO. "The Optimal Organization of Multiprogrammed Multi-level Memory", Proc. ACM SIGOPS Workshop on Syst. Performance Evaluation, Harvard University, April 1971.

Be 66 L. A. BELADY. "A Study of Replacement Algorithms for a Virtual Storage Computer,"IBM Syst. J. 5,2, 1966.

Be 68 L. A. BELADY, C. J. KUEHNER. Dynamic Space Sharing in Computer Systems, IBM Research Division Report RC-2064 (#10525), April 1968.

Bo 07 M. BÔCHER. Introduction to Higher Algebra, MacMillan Co., New York, 1907.

Bu 71 J. BUZEN. "Analysis of System Bottlenecks Using a Queuing Network Model," Proc. ACM SIGOPS Workshop on Syst. Performance Evaluation, Harvard University, April 1971.

Cou 70/1 P. J. COURTOIS, J. GEORGES. "An Evaluation of the Stationary Behavior of Computations in Multiprogramming Computer Systems," ACM Int. Comp. Symposium, Bonn, Germany, 1970.

Cou 70/2 P. J. COURTOIS. "Multiprogramming in Storage Hierarchies," Séminaires de Programmation 1969-1970, Centre de Calcul, Univ. de Grenoble, France.

Cou 71 P. J. COURTOIS, J. GEORGES. "On a Single-Server Finite Queuing Model with State-Dependent Arrival and Service Processes," Operations Research, Vol. 19, No. 2, March-April 1971.

Cof 66 E. G. COFFMAN, JR., R. C. WOOD. "Interarrival Statistics for T-S Systems," CACM, Vol. 9, No. 7, July 1966.

De 68/1 P. J. DENNING. Resource Allocation in Multiprocess Computer Systems, Ph.D. Thesis, MAC-TR-50, May 1968.

De 68/2 P. J. DENNING. "Thrashing: Its Causes and Prevention," AFIPS FJCC 1968.

De 70 P. J. DENNING. "Virtual Memory," Computing Surveys, Vol. 2, No. 3, 1970.

Di 68 E. W. DIJKSTRA. "The Structure of the 'THE' Multiprogramming System," CACM, Vol. 11, No. 5, May 1968.

Di 69/1 E. W. DIJKSTRA. "Complexity Controlled by Hierarchical Ordering of Function and Variability," Software Engineering, Ed. Naur and Randell, NATO, Brussels, January 1969.

Di 69/2 E. W. DIJKSTRA. Notes on Structured Programming, Technische Hogeschool, Eindhoven, August 1969.

Fe 68 W. FELLER. An Introduction to Probability Theory and Its Applications, Vol. I, J. Wiley and Sons, 3rd Ed., 1968.

Fi 62/1 F. M. FISHER, A. ANDO. "Two Theorems on *Ceteris Paribus* in the Analysis of Dynamic Systems," American Political Science Review, 61, March 1962.

Fi 62/2 F. M. FISHER. "An Alternate Proof and Extension of Solow's Theorem on Non-Negative Square Matrices," Econometrica, Vol. 30, No. 2, April 1962.

Fin 66 G. H. FINE et al. "Dynamic Program Behavior Under Paging," Proc. 21st Nat. Conf. ACM, 1966.

Fr 52 FRAZER et al. Elementary Matrices, Cambridge University Press, 1952.

Go 67 W. J. GORDON, G. F. NEWELL. "Closed Queuing Systems with Exponential Servers," Operations Research, Vol. 75-2, March-April 1967.

Ja 63 J. R. JACKSON. "Jobshop-Like Queuing Systems," Management Sc. 10, 131-142, 1963.

Kh 55 A. I. KHINTCHINE. Mathematical Methods in the Theory of Queuing, Griffin, London, 1955, 1960 (translation).

Kl 68 L. KLEINROCK. "Certain Analytic Results for Time Shared Processors," Proc. IFIP Congress 68, 1968.

Li 61 J. D. C. LITTLE. "A Proof for the Queuing Formula $L = \lambda W$," Operations Research Vol. 9, 383-387, 1961.

Ma 64 M. MARCUS, H. MINC. A Survey of Matrix Theory and Matrix Inequalities, Allyn and Bacon, Boston, 1964.

Mg 70 R. L. MATTSON, J. GESCEI, D. R. SLUTZ, I. L. TRAIGER. "Evaluation Technique for Storage Hierarchies," IBM Systems Journal, Vol. 9, No. 7, 1970.

Mu 70 A. P. MULLERY, G. C. DRISCOLL. "A Processor Allocation Method for Time-Sharing," CACM, Vol. 13, No. 1, January 1970.

Pa 67 D. L. PARNAS, J. A. DARRINGER. "SODAS and a Methodology for System Design," Proc. AFIPS Fall Joint Comp. Conf., 1967.

Pa 69 D. L. PARNAS. "More on Simulation Languages and Design Methodology for Computer Systems," Proc. AFIPS Spring Joint Comp. Conf., 1969.

Ra 69 B. RANDELL. "Towards a Methodology of Computing System Design," Software Engineering, Ed. Naur and Randell, NATO, Brussels, January 1969.

Sc 64 SCHWARTZ et al. "A General Purpose Time-Sharing System," AFIPS SJCC, 1964.

Sc 67 SCHWARTZ et al. "The SDC Time-Sharing System Revisited," Proc. 22nd Nat. Conf. ACM, 1967.

Si 61 H. A. SIMON, A. ANDO. "Aggregation of Variables in Dynamic Systems," Econometrica, Vol. 20, No. 2, April 1961. Reprinted in Essays on the Structure of Social Sciences Model, A. Ando, F. M. Fisher, H. A. Simon, MIT Press, 1963.

Si 62 H. A. SIMON. "The Architecture of Complexity," Proc. of the American Philosoph. Soc., Vol. 106, No. 6, December 1962.

Si 69 H. A. SIMON. The Sciences of the Artificial, MIT Press, Cambridge, 1969.

Sm 66 J. L. SMITH. "An Analysis of Time-Sharing Computer Systems Using Markov Models," AFIPS Proc. Spring Joint Comp. Conf., 1966.

Sm 67 J. L. SMITH. "Multiprogramming under a Page on Demand Strategy," CACM, Vol. 10, No. 10, October 1967.

Ta 60 L. TAKACS. Stochastic Processes, Problems and Solutions, Methuen and Co., London, 1960.

To 65 R. A. TOTSCHEK. An Empirical Investigation into the Behavior of the SDC Time-Sharing System, SDC report SP-2131/000/00, August 1965.

Va 67 L. VARIAN, E. G. COFFMAN. "An Empirical Study of the Behavior of Programs in a Paging Environment," ACM Symp. on O.S. Principles, Gatlinsburg, October 1967.

Ve 69 M. H. VAN EMDEN. "Hierarchical Decomposition of Complexity," Machine Intelligence 5, Edinburgh University Press, 1969.

Wa 66 V. L. WALLACE, R. S. ROSENBERG. "Markovian Models and Numerical Analysis of Computer System Behavior," AFIPS Proc. Spring Joint Comp. Conf., 1966.

Wa 69 V. L. WALLACE, D. L. MASON. "Degree of Multiprogramming in Page-on-Demand Systems," CACM, Vol. 12, No. 6, June 1969.

Wed 34 J. H. M. WEDDERBURN. Lectures on Matrices, American Mathematical Society, 1934.

Web 98 H. WEBER. Lehrbuch der Algebra, Zweite Auflage, Erster Band, 1898.

Zu 68 F. W. ZURCHER, B. RANDELL, "Iterative Multilevel Modelling, A Methodology for Computer System Design," IFIP Congress 68, Edinburgh, 1968.