THE DESIGN OF INFORMATION STRUCTURES:

BASIC ALLOCATION STRATEGIES FOR ORGANIZATIONS

by

Debra Ann Stabile

B.S., State University of New York at Stony Brook
(1976)

M.S., State University of New York at Stony Brook
(1978)

SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF

MASTER OF SCIENCE
IN
OPERATIONS RESEARCH

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1981

Signature of Author Signature redacted
Sloan School of of Management
June 12, 1981

Certified by Signature redacted
Wilbur B. Davenport, Jr.
Thesis Supervisor

Signature redacted
Alexander H. Levis
Thesis Supervisor

Accepted by
Jeremy F. Shapiro
Chairman, Interdepartmental Committee

THE DESIGN OF INFORMATION STRUCTURES:

BASIC ALLOCATION STRATEGIES FOR ORGANIZATIONS

by

DEBRA ANN STABILE

Submitted to the Sloan School of Management
on June 12,1981 in partial fulfillment of the
requirements for the Degree of Master of Science in
Operations Research

ABSTRACT

A methodology for designing the information structures for decision makers who comprise the boundary between an organization and its environment is presented. The environment is modeled as a source that generates symbols or messages that the organization members must process without being overloaded. Two basic information reduction strategies are considered: 1) creation of self contained tasks, and 2) creation of slack resources. The former leads to the partitioning of the input signal and the parallel processing of the partition; the latter to alternate processing where each decision maker receives signals according to some stochastic or deterministic rule but is given more time to process them, i.e., a delay is introduced. These two strategies are then integrated to produce a variety of information structures for special cases. The concepts of specialization and redundancy are considered and their impact on the information structures explored.

Thesis Supervisor: Wilbur B. Davenport, Jr.

Title: Professor of Communications Science and Engineering

Thesis Supervisor: Alexander H. Levis

Title: Senior Research Scientist

ACKNOWLEDGEMENT

The task of writing a thesis requires inputs from many sources. It also requires processing information and producing an accurate document in the time available. To avoid overload, the task can be partitioned and subtasks assigned to various individuals.

A thesis requires a topic. The searching for and developing of ideas can be shared with a thesis advisor. I have been extremely fortunate to have benefitted from the wisdom of two men, Professor Wilbur B. Davenport, Jr. and Dr. Alexander H. Levis. Professor Davenport's guidance and understanding helped ease the transitions I faced during my stay at M.I.T. Alex Levis has demonstrated a faith, sense of humor and a patience that can be equaled by few. His insight, his advice and suggestions, his standard of excellence have all contributed significantly to the writing of this thesis. His friendship has become a valued outcome of this work.

The writing of any thesis includes periods of frustrations and periods of absolute exhilaration. An outstanding family of friends and relations had the task of providing encouragement, perspective and distractions. Michael Abraham has performed light years beyond the call of friendship for me throughout a good portion of my stay at M.I.T.. Without his the final production of this thesis could never have been achieved as rapidly and accurately; i.e., efficiently. However I will refrain from commenting about the word processor. Maqbool Dada,along with Michael has been my Sloan Answer man as well as a very special friend. If any roadblocks came up I could turn to him for an answer (be it right or wrong!). Bruce Smith, Kevin Boettcher, and Peter Doerschuk had the patience and humor to listen to my rantings of impending doom that thankfully never came. I would also like to thank Professor Arnold I. Barnett and Professor Alvin W. Drake for their humor, support and friendship.

My family has provided a continuous input of love and encouragement and the main source of strength I needed to perform this task. It is to them that I dedicate this thesis.

The task of actually producing the final document was greatly facilitated by my typist, and friend, Diane McCoy. The beautiful figures are the work of Art Giordani and LIDS and are gratefully appreciated.

This research was carried out at the M.I.T. Laboratory for Information and Decision Systems with support provided in part by the Office of Naval Research under contract ONR/N0O014-77-C-0532 (M.I.T. OSP No. 85552) and in part by the Air Force Office of Scientific Research under Contract AFOSR-80-0229 (M.I.T. OSP No. 89620).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

ORGANIZATIONS AND INFORMATION STRUCTURES

"Why is there never enough time to do a job right
but always enough time to do it over again?"

Anon.

1.1 INTRODUCTION

"Organizations are set up to execute a task which is too complex or too time-consuming to be performed satisfactorily by an individual." [1] No single structure is ideal for all organizations [2]; actual organizations range from being centralized to totally decentralized, from strict hierarchies to matrices.

The design of organizations can be decomposed into two interrelated problems: the organizational form problem in which the information and decision structures are specified, and the organizational control problem in which the operating rules or procedures and the monitoring and enforcement strategies are determined. The manner in which information is received and processed by organization members is a key descriptor of the structure of an organization. Indeed, "communication - the exchange of information and the transmission of meaning - is the very essence of an organization. To move from an unorganized state to an organized state requires the introduction of constraints and restrictions to reduce diffuse and random communications to channels appropriate for the accomplishment of organizational objectives (tasks)." [3]

The organization, perceived as an open system [4], interacts with its environment; it receives signals or messages in various forms that contain information relevant to the organization's tasks. These messages must be identified, analyzed and then transmitted to their appropriate destinations within the organization.

From this perspective the organization acts as an information user. It can also act as an information generator or source; in this mode, the organization's output to its environment, whether in the form of actions or signals, is the input to other organizations. To assure the smooth functioning of the organization with respect to the execution of tasks, it is essential that tasks be executed accurately and quickly, i.e., effectively.

How an organization is structured to receive signals from its environment has direct consequences on the internal structure of the organization and on its performance. The specific structure depends on the nature and characteristics of the signals that can be received, on the task to be performed, and on the capabilities and limitations of the individual members comprising the organization. By considering only the boundary between the organization and the environment, a major simplification occurs; the boundary itself can be thought of as a single echelon of organization members. While these members may occupy different positions in the internal organizational structure, the relevant characteristic is that they receive direct inputs from the environment. In that sense, no member is subordinate to another; the members constitute a single echelon. However, individuals, or groups of individuals, can have very different capabilities and limitations that reflect, indirectly, their position in the organization. For example, they can process only certain classes of signals (specialization) or they can deal with limited levels of uncertainty. Since it is important to remember that the single echelon may include commanders as well as operators of monitoring systems, executives as well as clerks, the term decision-maker has been used to describe all members.

The determination of a methodology for designing the information structure for the single echelon just described is the basic goal of this thesis. The approach taken to meet this goal consists of the development of an analytic framework for design, the analysis of basic structures, and the use of them in designing more complex organizational forms.

1.2 DESIGN OF ORGANIZATIONS

In the previous section, the statement of objective of this thesis was presented, i.e., to design single-echelon (S-E) structures which perform complex information processing tasks efficiently. The decision makers (DMs) in the single echelon and the particular sequencing and allocation of information received by the DMs define the organizational form. In order to perform a complex task efficiently, it is assumed that more than one DM is required. The design of the S-E structures in which no DM is subordinate to any other DM will depend upon the constraints imposed by the organization. Two basic premises of the design method are that (a) there is no one best way to organize in general and (b) any way of organizing is not equally effective with respect to performing a specific task. [2] Characteristics of organizations (S-Es) will be defined which suggest that certain structures will, in fact, be more effective than others for performing a specific complex task. For the types of organizations considered, the performance of a complex task is equivalent to the processing of information, where information is defined to be the data received by the DMs in the S-E. Galbraith has argued that variations in the amounts of information (data) that are processed are primarily responsible for the variations in organizational forms. [2] Such variations are largely a result of the uncertainty associated with a given task. Uncertainty has been defined to be "the difference between the amount of data required to perform the task and the amount of data already possessed by the organization." [2]

A strong positive correlation exists between the level of uncertainty and the amount of data necessary to be processed among DMs so that the task is performed efficiently. The diversity of the outputs produced by the S-E, the number of different input resources utilized and the level of performance required all contribute strongly towards determining the amount of data needed. [2]

It will be assumed that the number of DMs necessary in the S-E is greater than one. The underlying premise is that no DM alone is able to process the required amount of data while simultaneously achieving the

required performance level. When a DM is overloaded, i.e., has been assigned more data than he is able to process in the prescribed time interval while still maintaining a given performance level, the DM can react in one of several ways. [5]

The DM may decide to reduce the amount of data he has to process by either randomly (rejection) or selectively (filtering) omitting data. The amount of data he may be required to process may be reduced also by having it preprocessed. He may decide to reduce the number of categories of discrimination, i.e., approximate the inputs, or he may reduce the required level of accuracy for processing the data and, in so doing, reduce the number of different outputs. If these alternatives seem unsatisfactory, he may decide to receive all the data allowing queues to build up, delaying the processing during periods of peak loads and attempting to catch up during time intervals when input symbols are assigned to other DMs. Otherwise the DM may simply choose not to perform the task. J. Miller found that at moderate rates of information input overload, all these methods described were used about equally. When the input rate far exceeded a DM's processing capacity, however, random and selective omission were the most significant methods of dealing with the situation. [5]

An alternative to having the data preprocessed, which was suggested before J. Miller's paper, was the employment of multiple parallel channels. [4] This thesis will focus on the use of parallel channels (DMs) which guarantee all the data received is processed immediately. Queuing also guarantees all the information is processed, but not immediately. The other approaches to reducing overload are unappealing to the organization designer in two ways: (1) data are omitted and (2) it costs to send data if the data are not used, i.e., wasteful expenditures are being made.

The concept of parallel DMs is analogous to the idea of distributed information processing with each DM performing a subtask. Thus the idea of considering a S-E's structure only is a relevant issue for any organization. Many studies in the literature have revealed that as the

uncertainty of the tasks increase, the "flatter," i.e., more distributed, an organization should become with respect to its DMs. [4]

Galbraith has suggested two information reduction strategies for organizations to address this issue: (1) Creation of Self-Contained Tasks, and (2) Creation of Slack Resources. In the first strategy, the original task is divided into a set of subtasks. This reduces the diversity of outputs each DM in the organization can produce as well as the number of different inputs since the DM need only receive the inputs pertaining to the given subtask. In this thesis, the slack resource will be time and the second strategy will involve a reduction in the required level of performance with respect to time. For example, if it was originally required to have the task completed in time δ, this time may be extended to some multiple of δ.

Two types of processing modes will be considered, parallel processing, which is associated with the first strategy, and alternate processing, which is associated with the second strategy. These two modes are the fundmental strategies employed to guarantee all data are processed without overloading any DM in the S-E. The two fundamental modes can be integrated in various ways so as to develop more complex organizational forms for the S-E.

1.3 PARALLEL PROCESSING MODE

First parallel processing is introduced in order to reduce the amount of information any particular DM receives. This assumes the task can be divided into subtasks with each subtask requiring some subset (not necessarily mutually exclusive) of the information. The subtasks are selected and assigned to DMs in such a way that each DM is capable of processing his data in time δ, or less, i.e., before his next input is received. This is referred to as parallel processing because the subtasks are carried out in parallel within the same time interval. This processing mode guarantees the expected output time of any processed input is equal to the expected input time, δ. The structure of the single echelon with parallel processing is shown in Figure 1.1. The detailed

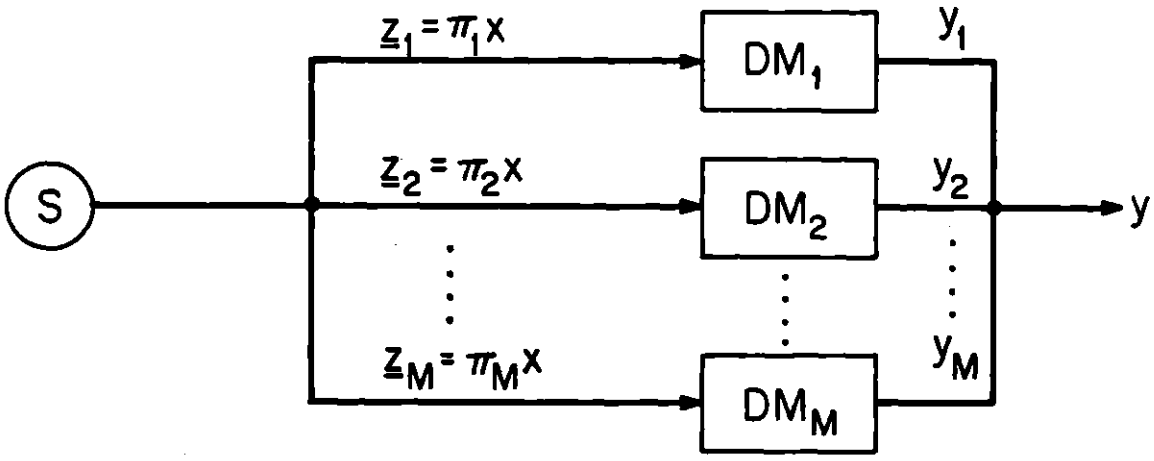


Figure 1.1. Parallel Processing

analysis and design of parallel processing is presented in Chapters 3, 4 and 5.

1.4 ALTERNATE PROCESSING MODE

Since information is being generated at a rate δ^{-1}, and a DM requires more than time δ to process the information (queueing of information has been removed from further consideration) additional DMs are introduced into the S-E. Each DM receives a different input signal. The number of additional DMs must be sufficient to receive and process information so that no DM receives another input signal until the previous one he received has been processed. This is referred to as alternate processing because the assignment of the inputs alternates among the DMs in the S-E. The structure of a single echelon with one form of alternate processing is shown in Figure 1.2. The precise rule for allocating inputs to the various DMs determines the minimum number of DMs necessary to process the inputs without overloading any DM. It will be shown in Chapter 6 that minimizing the number of DMs leads to the maximization of the expected output rate. This, in turn, leads to the minimum reduction in the level of performance of the organization.

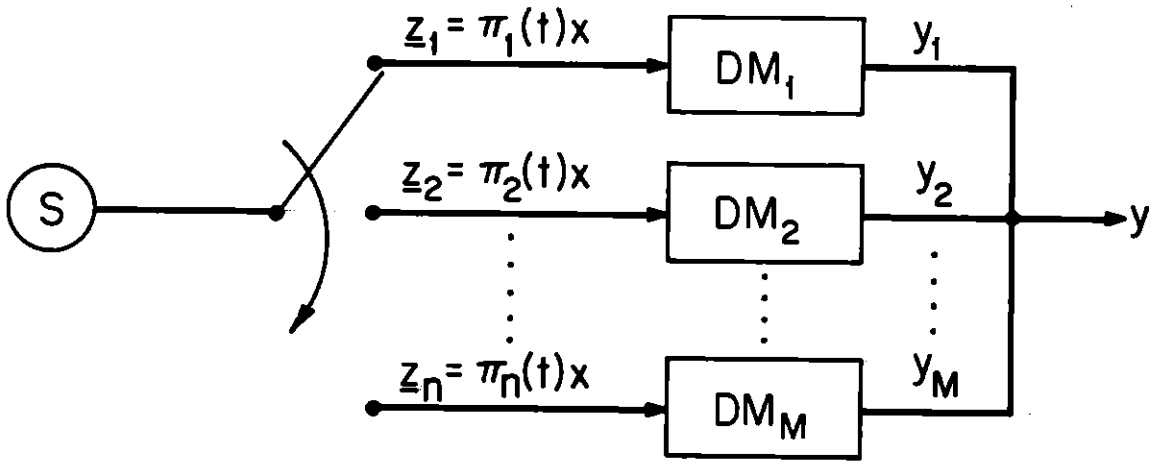


Figure 1.2. Alternate Processing (Periodic)

1.5 INFORMATION STRUCTURES

This thesis is concerned with the information structures and their associated characteristics for both the parallel processing mode and the alternate processing mode. The design of an information structure is the specification of what information is assigned to which DM in the S-E without causing overload. Both modes assume at least two DMs are necessary to perform the complex task and all DMs in the S-E work in parallel, i.e. no DM is subordinate to any other DM.

The information structures that are discussed can be characterized by two classes of attributes. The first class of attributes describes the complex task to be performed, e.g., the number of different types of data that must be processed, the uncertainty associated with each type of data, and the interdependence between types of data. The second class of attributes describe the slack resource, time. Input symbols are generated at some rate δ^{-1} and must be processed by the DMs in the S-E without overloading any DM, i.e., giving him more information to process than he has time available to process it in.

Parallel processing is associated with the partitioning of the task

into subtasks, in the absence of any slack in time. Alternate processing is associated with the allocation of the slack resource (time). The task is not partitioned, but each DM is allotted more time to carry out complete tasks. The information structure associated with alternate processing defines for each DM the rate at which he is assigned a complex task. In designing realistic organizations, both strategies are used through the integration of these two processing modes. For example, the complex tasks are partitioned in subtasks and then each subtask is processed via alternate processing.

In parallel processing, since there is no slack resource, the expected output rate of a processd input symbol must be equal to δ^{-1}, the expected input rate. The properties of the different types of data to be processed by the DMs in the S-E are the primary factors in determining the information structure.

In alternate processing, the expected output rate remains equal to the expected input rate; however, a delay that is strictly greater than δ has been introduced. Since each DM must process an entire input symbol, the information structure associated with alternate processing is not particularly sensitive to the properties of the information contained in the input symbol.

The major factors affecting the information structure are the attributes that characterize each DM. The primary factor is the efficiency with which a DM can process an input symbol, i.e. how much time each DM requires to process his inputs without overload. The latter, in turn, determines the delay by the single echelon as a whole in performing the assigned tasks.

1.6 OUTLINE OF THESIS

In Chapter 2 a more formal definition of the inputs, the decision makers (DMs) and information structures is presented. A detailed discussion of the parallel processing mode and its associated assumptions is presented in Chapter 3. A mathematical program (generalized network)

is formulated, the solution of which leads to the construction of the corresponding information structure. This formulation considers explicitly all possible subtasks that can be assigned to each DM in the single-echelon. This explicit enumeration of all possible subtask assignments causes the corresponding problem's dimension to be very large.

In Chapter 4, various simplifying assumptions are introduced into the problem which lead to significant reductions in the problem's size. These assumptions concern the properties of the DMs and the data necessary to perform the subtasks. Examples are included to illustrate more clearly the impact.

The dimensionality issue is addressed in Chapter 5 by reformulating the mathematical program. Although the problem is still formulated as a generalized network, the subtasks are considered implicitly and are assumed to be completely independent. A substantial reduction in the size of the problem is achieved as a result of this assumption and the corresponding formulation.

In Chapter 6 a detailed discussion of the alternate processing mode and its associated assumptions is presented. Procedures are discussed which lead to the optimal construction of the information structure. Special cases are considered under various simplifying assumptions.

Finally, a methodology for designing single echelon information structures is proposed in Chapter 7. The methodology allows for the integration of the parallel and alternate processing modes. Information structures associated with more realistic organizational designs are constructed as a result of this integration.

CHAPTER 2

TASKS, DECISION MAKERS, AND ORGANIZATIONS

In this chapter, two elements of the design problem will be explored: the modeling of the task to be performed and the properties of the decision makers (DMs) who comprise the single echelon (S-E).

2.1 SOURCES

The single echelon receives data from one or more sources external to it. Every δ_n units of time on the average, each source n generates symbols, signals, or messages x_{ni} from its associated alphabet X_n, i.e.,

$$x_{ni} \in X_n \qquad i = 1,2,\ldots,\gamma_n \tag{2.1}$$

where γ_n is the dimension of X_n. Therefore, $1/\delta_n$ is the mean frequency of symbol generation from source n. Let p_{ni} be the probability that x_{ni} is the generated symbol:

$$\begin{aligned} p_{ni} &= p(x_n = x_{ni}) \qquad && i = 1,2,\ldots,\gamma_n \\ \sum_{i=1}^{\gamma_n} p_{ni} &= 1 && n = 1,2,\ldots,N \end{aligned} \tag{2.2}$$

The task to be performed is defined as the processing of the input symbols x_n by the single echelon to produce output symbols. It is assumed that a specific complex task that must be performed can be modeled by N' such sources of data. Rather than considering these sources separately, one supersource, $\underline{x}'$, composed of these N' sources, is created. The input symbol $\underline{x}'$ may be represented by an N'-dimensional vector with each of the sources represented by a component of this vector, i.e.,

$$\underline{x}' \equiv (x_1,\ldots,x_n,\ldots,x_{N'}). \tag{2.3}$$

The set of components comprising the input vector, $\underline{x}'$, is denoted by $\mathbf{L}$. Each symbol generated from this supersource (input vector in all future discussions) is selected from an alphabet set, X. Each element $\underline{x}_j$ of X is of dimension N', i.e.,

$$\underline{x}_j \equiv (x_{1j},\ldots,x_{nj},\ldots,x_{N'j}) \tag{2.4}$$

It follows that the dimension of X is γ, the product of the dimensions of each of the components' alphabets, i.e.,

$$\gamma = \prod_{n=1}^{N'} \gamma_n. \tag{2.5}$$

To determine the probability that symbol $\underline{x}_j$ is generated,

$$p_j = p(\underline{x} = \underline{x}_j) \qquad j = 1,2,\ldots,\gamma,$$

the independence between components must be considered. If all components are mutually independent (for definition see Hogg and Craig [6]) then p_j is the product of the probabilities that each component of $\underline{x}_j$ takes on its respective value from its associated alphabet:

$$p_j = \prod_{n=1}^{N'} p_{nj}. \tag{2.6}$$

When all components of the input vector are mutually independent this is referred to as being of <u>finest grain</u>. In many situations, this assumption is unrealistic. It is more common to have some components probabilistically dependent.

If two or more components are probabilistically dependent on each other but as a group are mutually independent from all other components of the input vector, then these dependent components can be treated as one supercomponent, x(n). Let $\mathbf{L}(n)$ be the set of interdependent components comprising x(n) and let s(n) be its dimension.

Each element of the alphabet X(n), associated with this component x(n) is of dimension s(n) and contains one element from each X_n, $n \in \mathcal{L}(n)$. The probability associated with each element of X(n) can be derived in one of two ways. If the probability of each element of the input vector is given initially, then the marginal probabilities associated with each $x_j(n)$ can be computed. Alternatively, if the individual probabilities of each element, $x_j(n)$, $n \in \mathcal{L}(n)$, are given, coupled with the restrictions imposed by the dependence of the individual elements, the joint probability of each $x_j(n)$ can be computed.

Supercomponents are constructed for each group of interdependent components. Let $\mathcal{L}'$ be this set of supercomponents. A new input vector, $\underline{x}$, is defined, composed of the mutually independent components and these supercomponents. This new $\underline{x}$ is of finest grain. The dimension of this input vector is N where

$$N = N' - \sum_{n \in \mathcal{L}'} (s(n) - 1). \tag{2.7}$$

For all future discussions this input vector $\underline{x}$ of dimension N is used.

The mean interarrival time of input vectors is δ. It is assumed that the mean interarrival time for each component n, δ_n, is equal to δ, i.e.,

$$\delta_n = \delta \qquad n = 1,2,\ldots,N. \tag{2.8}$$

This assumption imposes synchronization between the generation of the individual source elements so that they may, in fact, be treated as one input symbol. It is also assumed that the generation of a particular input vector (input symbol), $\underline{x}_j$, is independent of the symbols generated prior to or after it.

2.2 DECISION MAKERS

Each of the decision makers (DMs) in the single-echelon (S-E) processes some subset of the components comprising the input vector. Each DM is distinguished by his

a) processing time function, $\bar{\tau}^m$, which yields the mean time for processing a particular set of components,

b) specialization; which components he is able or qualified to process, and

c) cost.

It seems intuitive that the more efficient (smaller $\bar{\tau}^m$) a DM is, the greater his cost will be.

The S-E consists of the (minimum) number of DMs required to process the input vector's components without anyone being overloaded. A DM is overloaded when the time required for him to process the components he is assigned exceeds δ, the mean interarrival time of input symbols requiring processing. As discussed in Chapter 1, if a DM is overloaded, he either makes mistakes in trying to complete his portion of the task or he rejects some of the data. Either situation results in a degradation of performance. Processing modes which are designed to help DMs avoid being overloaded were introduced in Chapter 1.

2.3 INFORMATION STRUCTURES

Organizational form problems, the specification of communication links between members of the organizaton, are being considered throughout this thesis. In particular, the links which specify the allocation of data from the input vector $\underline{x}$ to each member of the single-echelon (S-E) are examined. Information structures are sought which will enable each decision maker (DM) to process the input symbols he receives without being overloaded.

Two processing modes, parallel and alternate, which lead to different information structures, are examined. Each processing mode results in M* DMs placed in parallel. The associated information structure specifies what data are allocated to which DM.

The implementation of the parallel processing mode leads to an information structure constructed from the partitioning of the input vector by components. The implementation of the alternate processing mode leads to an information structure specifying the frequency with which each

DM receives an input symbol.

The partitioning of the input vector, $\underline{x}$, into groups of components, is considered first. Each group (partition) of components is assigned to different DM. The k-th partition of components is denoted by $\underline{z}_k$. It is an s_k dimensional vector where s_k equals the number of components in the k-th partition. The vector $\underline{z}_k$ is derived from the corresponding partitioning matrix, π_k, which has dimension s_k x N and rank s_k, i.e.,

$$\underline{z}_k = \pi_k \underline{x}. \tag{2.9}$$

The elements of the partitioning matrix, π_k, are either zero or one. Each column of this matrix has at most one non-zero element. Each row has exactly one non-zero element. Since the order of the components in the partition vector, $\underline{z}_k$, is of no consequence, any other matrix π_k' obtained by interchanging the rows of π_k is represented by π_k.

The partitioning matrix of the m-th DM, π_k^m, specifies the components assigned to him.

$$\underline{z}_k^m = \pi_k^m \underline{x} \tag{2.10}$$

An information structure, Π, is composed of the set of partitioning matrices associated with the M^* DMs in the S-E, i.e.,

$$\Pi = (\pi_{k_1}^1, \ldots, \pi_{k_m}^m, \ldots, \pi_{k_{M^*}}^{M^*}). \tag{2.11}$$

As stated previously, the second type of information structure specifies the frequency q_m, with which each DM receives an input symbol. Two strategies are considered, stochastic and deterministic. For stochastic strategies, the frequency is, in fact, the probability that a DM will receive the next input symbol generated. The information structure associated with the m-th DM is:

$$\pi^m(t) = \begin{cases} I & \text{with probability } q_m \\ 0 & \text{with probability } (1 - q_m) \end{cases} \qquad m = 1,2,\ldots,M^* \tag{2.12}$$

where q_m satisfies the conditions

$$\sum_{m=1}^{M^*} q_m = 1; \quad q_m \geq 0 \tag{2.13}$$

and δ/q_m is the average available time the m-th DM has to process this symbol.

For deterministic strategies, the frequency q_m expresses the exact number of symbols assigned to the m-th DM out of the total number of input symbols generated in a sequence of length δ'. In addition, the information structure specifies the exact order in which these $q_m\delta'$ symbols are assigned to the m-th DM; i.e.,

$$\Pi^m(t) = \begin{cases} I & \text{if } t = t^m_{jf} \quad m = 1,2,\ldots,M^*; \; j = 1,2,\ldots,q_m\delta'; \\ & \qquad\qquad f = 1,2,\ldots,\delta' \\ 0 & \text{otherwise} \end{cases} \tag{2.14}$$

where t^m_{jf} indicates the j-th time the m-th DM is assigned an input symbol and it is the f-th assignment in a sequence of length δ' and

$$(\sum_j t^m_{jf})/\delta' = q_m \qquad m = 1,2,\ldots,M^* \tag{2.15}$$

A special case of the deterministic strategy is the periodic strategy, in which the frequency of symbol arrival to each DM is the same.

$$q_m = q = 1/M^*$$

The information structure constructed for this strategy is:

$$\Pi^m(t) = \begin{cases} I & \text{when } t = \theta M^* + m \qquad m = 1,2,\ldots,M^*; \\ & \qquad\qquad\qquad\quad \theta = 0,1,2,\ldots \\ 0 & \text{otherwise} \end{cases}$$

where θ is the index of the cycle.

Each of the information structures presented in this section is

discussed in detail in the following chapters.

2.4 PROCESSING TIME FUNCTIONS

Several factors affect the time required for a decision maker (DM) to process information. [5] The uncertainty of the input symbol generated and the number of possible input-output responses are two of these factors. Keying in on these two factors, Miller introduced a processing time function which has an information interpretation. [5] Hyman and Hick provided experimental evidence that information, rather than number of inputs or outputs, was a more appropriate measure [5]; Shannon's measure of information was used. [5] Let

s	be the number of components the m-th DM processes
$t^m(s)$	be a parameter which is a function of the number of components assigned to the m-th DM
c^m	be a constant
$p(\underline{z}_{kj})$	be the probability associated with the j-th element of the k-th partition vector's alphabet.

Then

$$\tau^m_{kj} = t^m(s) - c^m \log p(\underline{z}_{kj})$$

If this processing time function is averaged over all possible elements of the input symbol to be processed, [5] i.e.,

$$\bar{\tau}^m_k = \sum_j p(\underline{z}_{kj}) \tau^m_{kj} \tag{2.16}$$

the following result is obtained:

$$\bar{\tau}^m_k = \sum_j p(\underline{z}_{kj}) \left(t^m(s) - c^m \log p(\underline{z}_{kj})\right)$$

$$= \sum_j p(\underline{z}_{kj})\, t^m(s) - \sum_j p(\underline{z}_{kj})\, c^m \log p(\underline{z}_{kj}) \tag{2.17}$$

$$= t^m(s) - c^m \sum_j p(\underline{z}_{kj}) \log p(\underline{z}_{kj}).$$

In information theory, the last term of eq. (6.17),

$$- \sum_j p(\underline{z}_{kj}) \log p(\underline{z}_{kj}) = H_k \tag{2.18}$$

is defined to be the entropy associated with the group of components, k. It follows that

$$\bar{\tau}^m_k = t^m(s) + c^m H_k. \tag{2.19}$$

The quantity $t^m(s)$ may be a function of the number of components assigned to the m-th DM. It describes the fixed cost, in terms of time, required to process any component regardless of the amount of information contained in it. As the number of components increases $t^m(s)$ increases; a multiplicative function of the number of components assigned to DM m, is often assumed, i.e.,

$$t^m(s) = t^m_o s. \tag{2.20}$$

where t^m_o is a constant.

The parameter c^m is assumed to be a constant for each DM considered.

To compute the average processing time for DM_m to process the group of components, k, the entropy H_k must first be computed. Many distinct groupings of the components of $\underline{x}$ can be constructed and an entropy must be computed for each of these groupings. For each supercomponent, x(n), the corresponding entropy is:

$$H_n = - \sum_{i=1}^{\gamma_n} p[x(n) = x_i(n)] \log p[x(n) = x_i(n)].$$

Since it has been assumed that all components in the input vector are mutually independent, the entropy of any group of components is equal to

the sum of the entropies of each of the components in the group, i.e.,

$$H_k = \sum_{n \in \mathbf{l}_k} H_n . \tag{2.21}$$

This results in substantial savings in computation. Rather than needing to compute the probabilities of each element of each alphabet for each distinct grouping of components in order to compute the associated entropy, it is only necessary to compute the entropy for each component of the input vector.

In the next four chapters the two types of decision structures, parallel and alternate, will be discussed. In Chapter 7 the integration of these processing modes will be explored, and a design methodology will be presented.

CHAPTER 3

PARALLEL PROCESSING: EXPLICIT ENUMERATION

3.1 INTRODUCTION

Information structures based on parallel processing are feasible when the input vector can be partitioned. As discussed in Chapter 2, an input vector is composed of components which are mutually independent. If the input vector is of dimension two or more, then partitions composed of different combinations of components can be constructed.

In a parallel processing structure, partitions of the input symbol are selected and assigned to the decision makers (DMs) in the organization. The group of DMs who, together, process the entire input symbol form the single-echelon (S-E). Each DM is constrained to process a partition of components from those that do not overload him, i.e., from those that result in a mean processing time of δ or less.

Every component of each input vector must be processed by at least one DM. If the objective is to minimize the number of DMs necessary to process the input vector, then there is no advantage to having a component processed more than once, since this could require additional DMs.

Redundancy, defined here as processing a component(s) more than once, may be required by an organization, however. Distributed Database Systems (DDSs) are characterized by having identical data processed and stored at different locations (i.e., with different DMs). In this case, the objective may be to minimize the number of DMs subject to the requirement of processing prespecified components more than once.

DMs may be specialized. This implies that they are able to process only certain specified components. Thus, only partitions composed of these specified components may be selected and assigned to these DMs.

Mathematical models can be formulated for selecting and assigning input symbol partitions satisfying constraints on the processing time or specialization of DMs. Mathematical programming is an appropriate modeling approach for this class of problems. This approach seeks "the optimum allocation of limited resources among competing activities under a set of constraints imposed by the nature of the problem being studied." [7] In this context, the components of the input vector correspond to the limited resources, the DMs correspond to the competing activities and the constraint sets include processing time capabilities and specializaion limitations of each DM. The general formulation of a mathematical program includes an objective function

$$J = f_0(\underline{y})$$

and a matrix of constraints

$$\begin{cases} \underline{f}(\underline{y}) \leq \underline{b} \\ \underline{y} \geq 0 \end{cases}$$

where $\underline{y}$ is the vector of variables representing the allocation of the scarce resources.

3.2 MATHEMATICAL PROGRAMMING FORMULATION

In order that the complex task be performed, every component of the input vector $\underline{x}$ must be processed. To achieve this, the components of $\underline{x}$ are partitioned into groups with a group assigned to each DM. Let all the possible partitions (distinct groupings of the components of x) be ordered and let **K** be the set of positive integers corresponding to these ordered partitions with $k \in \mathbf{K}$. The dimension of **K** is K, the total number of distinct partitions that can be constructed and should be considered explicitly by each DM. Each component x_n of the input vector $\underline{x}$ is either included in each partition or it is not and thus $K = 2^N$, where N equals the dimension of the input vector. Had some of the components not been mutually independent, then several partitions of components would not be allowed since these dependent components would have to be included

together in a partition. A partition would not be valid if it included a strict subset of these dependent components.

The dimension of the k-th partition, S_k, is the number of components included in the partition. The alphabet and corresponding probability distribution associated with each partition are constructed easily since the components comprising each partition are mutually independent. Let the dimension of the alphabet of the k-th partition be γ^k. Then,

$$\gamma^k = \prod_{n \in \mathbf{L}_k} \gamma_n \tag{3.1}$$

where γ_n is the dimension of the alphabet associated with each component x_n and $\mathbf{L}_k$ is the set of positive integers corresponding to the components comprising the k-th partition.

The k-th partition, $k \in \mathbf{K}$, can be represented by a vector $\underline{z}_k$ composed of the components in that partition. A particular partition vector, $\underline{z}_k$, is defined as

$$\underline{z}_k = \pi_k \underline{x} \tag{3.2}$$

where π_k is the partitioning matrix that defines the k-th partition. The dimension of π_k is $(S_k \times N)$ and its rank is S_k.

Let $\mathbf{K}^m$ represent the set of partitions of components the m-th DM can process without overload, i.e.,

$$\bar{\tau}^m_k = t^m(S_k) + c^m H_k \leq \delta , \qquad m = 1,2,\ldots,M \tag{3.3}$$

the dimension of $\mathbf{K}^m$ is K_m. If the m-th DM is specialized, then the set $\mathbf{K}^m$ is reduced further to include only those partitions composed of components DM_m can process.

If the k-th partition is selected and assigned to the m-th DM, then the corresponding partitioning matrix is π^m_k and the corresponding partition vector is $\underline{z}^m_k$. The information structure of the overall

partitioning problem is

$$\Pi = (\pi^1_{k_1}, \ldots, \pi^m_{k_m}, \ldots, \pi^M_{k_M}) \tag{3.4}$$

The partition of components assigned to each DM is selected from his set of allowable partitions, $\mathbf{K}^m$. This guarantees an information structure in which all components are processed without overloading the decision makers. A mathematical program (MP) is formulated, which seeks the optimal selection and assignment of partitions of components to DMs so that all components are processed in time δ or less on the average using the minimum number of DMs. The information structure will be derived from the optimal solution to this problem.

The constraint matrix of the formulation is derived first. The initial formulation is concerned with guaranteeing that every component is processed exactly once. All components of data contained in the source are assumed necessary to perform the task.

If the k-th partition is selected and assigned to some DM, then every component contained in that partition can be assumed processed. Furthermore, each partition is restricted implicitly to being selected and assigned to at most one DM. Variable D_{kn} is defined to exist if the k-th partition contains the n-th component. Thus, S_k variables D_{kn} are associated with every partition k, where $k = 1,2,\ldots,K$. Variables D_{kn} are also defined to be binary so that if the k-th partition is selected and assigned to some DM then D_{kn} equals one for each $n \in \mathbf{L}_k$ and zero otherwise.

To guarantee that each component is processed once and only once, exactly one of the partitions which contains this component must be selected and assigned to a DM. Let $\mathbf{K}_n$ be the set of partitions which contain the component n. The corresponding set of constraints follows directly:

$$\sum_{k \in \mathbf{K}_n} D_{kn} = 1 \qquad n = 1,2,\ldots,N \tag{3.5a}$$

$$D_{kn} = 0,1 \qquad k \in \mathbf{K}_n;\ n = 1,2,\ldots,N \qquad (3.5b)$$

Exactly one of the variables, D_{kn} will be equal to one for each n (i.e., in each equation) and all other variables will be equal to zero. No variable D_{kn} exists presently for the partition k which corresponds to no components being assigned to a DM, i.e., the empty set. An (N+1)-st component, referred to as the SINK, is created which corresponds to the empty set. Thus, if K denotes the empty set partition, the corresponding partition-component variable is $D_{K,N+1}$. This allows a DM not to process any components of the input vector. Since the goal is to find M^*, the minimum number of DMs necessary to process the input vector without overload, the initial number of DMs available, should be greater than or equal to M^*. This is where the artistry of mathematical modeling comes into play. The number of available DMs in the formulation should be large enough to guarantee that a feasible solution can be found, but small enough so as not to require an unnecessarily large number of constraints to be included and considered. If a DM should have no components assigned to him, this DM is, by definition, no longer part of the single-echelon.

To define which DM is assigned what partition, variables Y_{mk} are introduced where, as previously defined, m corresponds to the m-th DM and k represents the k-th partition. Each DM is assigned one partition. Thus Y_{mk} is a binary variable which equals one if DM_m processes partition k, zero otherwise. This set of constraints can be expressed as

$$\sum_{k=1}^{K} Y_{mk} = 1 \qquad m = 1,2,\ldots,M \qquad (3.6a)$$

$$Y_{mk} = 0,1 \qquad m = 1,2,\ldots,M;\ k = 1,2,\ldots,K \qquad (3.6b)$$

Each DM must also be constrained to select from among those partitions whose average processing times do not overload him, i.e., $\bar{\tau}^m_k \leq \delta$. The values, $\bar{\tau}^m_k$, have been computed independently of the MP formulation. To guarantee that a DM is not assigned to process components which will overload him the set of constraints

$$\sum_{k=1}^{K} \bar{\tau}_k^m Y_{mk} \leq \delta \qquad m = 1,2,\ldots,M \tag{3.7}$$

must also be included. For each DM exactly one variable Y_{mk} will have a positive value, all others being equal to zero. For this variable, the processing time constraint (3.7) will be satisfied.

Note, however, that earlier in this section the set $\mathbf{K}^m$ was defined to be the set of partitions of components that the m-th DM could process without overload. Therefore, constraint set (3.7) need not be considered explicitly. Rather, constraint set (3.6a) could be modified for the m-th DM so that only those partitions k contained in $\mathbf{K}^m$ are considered, i.e.,

$$\sum_{k \in \mathbf{K}^m} Y_{mk} = 1 \qquad m = 1,2,\ldots,M \tag{3.8}$$

The sets of constraints already considered, (3.5), (3.6b), and (3.8) can be decoupled completely. However, a relation between the partition assigned to each DM and the partition which processed a particular set of input symbol components must be established.

If the components of partition k' can be processed by the m'-th decision maker, $DM_{m'}$, without overload and partition k' is assigned to $DM_{m'}$, then $Y_{m'k'} = 1$ and $Y_{m'k} = 0$ for all $k \in \mathbf{K}^m$ excluding k'. If $DM_{m'}$ is assigned partition k', then the components of partition k', $\mathbf{L}_{k'}$, will be processed. It follows directly that

$$\begin{aligned} D_{k'n} &= 1 \quad \text{for} \quad n \in \mathbf{L}_{k'} \\ D_{kn} &= 0 \quad \text{for} \quad n \in \mathbf{L}_{k'};\ k \in \mathbf{K}_n \ \text{where } k \neq k' \end{aligned} \tag{3.9}$$

The corresponding coupling constraint must guarantee that 1) if any DM is assigned to process partition k', then each component of k' is processed and 2) if no DM is assigned partition k' then the processing of its components must be achieved via the assignment of other partitions that contain these components. As previously stated, it is assumed that a partition is assigned to at most one DM. Thus $\sum_{m=1}^{M} Y_{mk}$ equals either one,

if some DM is assigned the partition, or zero if no DM is assigned this partition. If $\sum_{m=1}^{M} Y_{mk} = 1$ then it follows directly that $D_{kn} = 1$ for each $n \in \mathcal{L}_k$. The set of coupling constraints becomes

$$\sum_{m=1}^{M} Y_{mk} - D_{kn} = 0 \qquad k = 1,2,\ldots,K;\ n \in \mathcal{L}_k \qquad (3.10)$$

Constraint sets (3.5a) and (3.10) allow the binary constraints on each D_{kn} (3.5b) to be relaxed. It is enough to require $D_{kn} \geq 0$ for $k = 1,2,\ldots,K$ and $n \in \mathcal{L}_k$. Since Y_{mk} is binary, $\sum Y_{mk}$ will always be integer. This restricts D_{kn} to be integer. The condition that $\sum D_{kn}$ be equal to unity implicitly restricts every D_{kn} to be binary. Consequently, the problem becomes a mixed integer linear program (MILP). The general formulation for the MILP constraints is [8]

$$\begin{cases} A_1\underline{x} + A_2\underline{v} \leq \underline{b} \\ \underline{x} \geq 0, \text{ integer} \\ \underline{v} \geq 0 \end{cases}$$

The MILP formulation for this problem is presented in Table 3.1. The sparseness of the coefficient matrix is immediately apparent. Algorithms which solve this class of problem and take advantage of the sparseness of A_1 and A_2 will be discussed in Section 3.4.

The information partitioning matrices, π_k^m, which comprise the information structure, Π, can be constructed easily from the solution to this optimization problem. The optimal solution will include exactly M^* variables Y_{mk} which are equal to unity. All other variables Y_{mk} will be equal to zero. Each variable, Y_{mk}, which equals unity identifies a DM and the partition assigned to him. Thus the partition vector assigned to the m-th DM, $\underline{z}_k^m$, and its corresponding partitioning matrix, π_k^m, are defined immediately for each m.

Mathematical programs also require objective functions. An initial objective is to process without overload the entire input vector using the minimum number of DMs. This corresponds to assigning the empty set

TABLE 3.1 MIXED INTEGER LINEAR PROGRAMMING FORMULATION

(a) General Formulation

$$\sum_{k \in K^m} Y_{mk} = 1 \qquad m = 1,2,\ldots,M \tag{3.11a}$$

$$\sum_{m=1}^{M} Y_{mk} - D_{kn} = 0 \qquad k = 1,2,\ldots,K;\ n \in L_k \tag{3.11b}$$

$$\sum_{k \in K_n} D_{kn} = 1 \qquad n = 1,2,\ldots,N \tag{3.11c}$$

$$Y_{mk} = 0,1 \qquad m = 1,2,\ldots,M;\ k \in K^m \tag{3.11d}$$

$$D_{kn} \geq 0 \qquad k = 1,2,\ldots,K;\ n \in L_k \tag{3.11e}$$

(b) MILP Formulation

$$A_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 & & & & & & & & & & & \\ & & & & 1 & 1 & \cdots & 1 & & & & & & & \\ & & & & & & & & \ddots & & & & & & \\ & & & & & & & & & \ddots & & & & & \\ & & & & & & & & & & \ddots & & & & \\ & & & & & & & & & & & 1 & 1 & \cdots & 1 \\ 1 & & & & 1 & & & & & & & 1 & & & \\ & 1 & & & & & 1 & & & & & & & & \\ & & \ddots & & & & & \ddots & & & \ddots & & & & \\ & & & \ddots & & & & & \ddots & & & & & & \\ & & & & 1 & & & & 1 & & & & & & 1 \\ & & & & & \cdots & & & \mathbf{0} & & \cdots & & & & \end{bmatrix} \qquad A_2 = \begin{bmatrix} & & & & & & & & & & & & & & \\ & & & & & \cdots & & \mathbf{0} & & \cdots & & & & & \\ & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & \\ -1 & & & & & & & & & & & & & & \\ & -1 & & & & & & & & & & & & & \\ & & \ddots & & & & & & & & & & & & \\ & & & \ddots & & & & & & & & & & & \\ & & & & -1 & & & & & & & & & & \\ 1 & 1 & & & & 1 & & & & & & & & & \\ & 1 & & & & & & & & & & & & 1 \\ 1 & & & & 1 & 1 & & & & & & & & & \\ & 1 & & & & & & & & & & & & & \\ 0 & \cdot & \cdot & & 0 & 1 & & & & & & & & & \end{bmatrix}$$

$$\underline{x} = [Y_{11}\ Y_{12} \cdots Y_{1K_1} \quad \cdots \quad Y_{m1}\ Y_{m2} \cdots Y_{mK_m} \quad \cdots \quad Y_{M1}\ Y_{M2} \cdots Y_{MK_M}]$$

$$\underline{v} = [D_{11}\ D_{22} \cdots D_{kn} \cdots D_{K,N+1}]$$

$$\underline{b} = [\underbrace{1\ 1 \ldots 1}_{M}\ \underbrace{0\ 0 \ldots 0}_{1 + N2^{N-1}}\ \underbrace{1\ 1 \ldots 1}_{N}]$$

$\underline{x} \geq 0$, binary; $\underline{v} \geq 0$

partition to as many of the available DMs as possible. The S-E includes only those DMs who process some component(s) of the input vector. To achieve this a cost C, where C is arbitrary but strictly positive, can be assigned to each variable Y_{mk}, and a zero cost to the remaining variables. It follows that the cost is reduced every time a DM is not assigned a partition of components. This makes intuitive sense in that labor costs are reduced, the smaller the number of DMs required. The objective function is thus

$$v = \min \sum_{m=1}^{M} C_m Y_{mk} \qquad (3.12)$$

Objective functions to achieve other goals can be presented. However, the initial goal is simply to find feasible solutions.

3.3 NETWORK FORMULATION

Mathematical programs often can be classified by the matrix structure corresponding to the set of constraints of the problem. Special structures of these formulations are often sought because efficient algorithms have been developed which can solve the problem quickly or provide insight into it. The sparseness of the constraint matrices presented in the previous section suggests the existence of a special structure. Figure 3.1 shows a graphic representation of the constraint sets (3.11a - 3.11e) for a small example (N = 3, M = 2), which suggests that an underlying network may exist. The ability to represent a mathematical model graphically and to solve the problem with a high degree of efficiency are two compelling reasons to model the problem as a network. [9]

A network is composed of nodes and arcs. Flow moves along the arcs and through the nodes. One or more source nodes exist from which flow originates. The conservation of flow law states that the flow entering a node equals the flow exiting that node. One or more terminal nodes exist at which all flow eventually collects. The mathematical programming representation of the pure network constraint matrix is

$$\begin{cases} \sum_j x_{ij} - \sum_h x_{hi} = b_i & i = 1,2,\ldots,n \\ \ell_{ij} \leq x_{ij} \leq u_{ij} \end{cases}$$

As shown in Section 3.2, general mathematical programming formulations are expressd as $A\underline{x} = \underline{b}$. The special structure of this matrix A is such that

a) every element is either 1, -1, or 0

b) every column contains at most two non-zero elements.

The special feature of the constant vector, $\underline{b}$, is that the sum of its elements is zero.

One advantage to pure network formulations is that the matrix is totally unimodular and, therefore, if all elements of $\underline{b}$ and the upper and lower bounds on $\underline{x}$ are integer, integer solutions to the MP are guaranteed without restricting the variables to be integer. [10] The constraint sets for the formulation of Section 3.2 can be modified so that one may take advantage of the special structure. The variables Y_{mk} and D_{kn} are arcs in the network and the DMs, partitions, and components are nodes.

A source node, S, and a terminal node, T, are added to the network. These nodes help to maintain the consistency of the network formulation, and to establish where the flow originates and where it collects. The binary variable which represents the arc carrying the flow from source S to the m-th DM is denoted by B_{Sm}. Since every DM is available for processing exactly one partition of components, B_{Sm} is equal to unity for all m. Similarly, V_{nT} is the binary variable which represents the arc carrying the flow from the component x_n to the terminal node T. Since every component of the input vector should be processed exactly once, V_{nT} is equal to unity for all n. Constraint set (3.11c) can be modified so that

$$\sum_{k \in K^m} D_{kn} = V_{nT} \qquad n = 1,2,\ldots,N \qquad (3.13)$$

The average processing-time feasibility constraints (3.7) again do

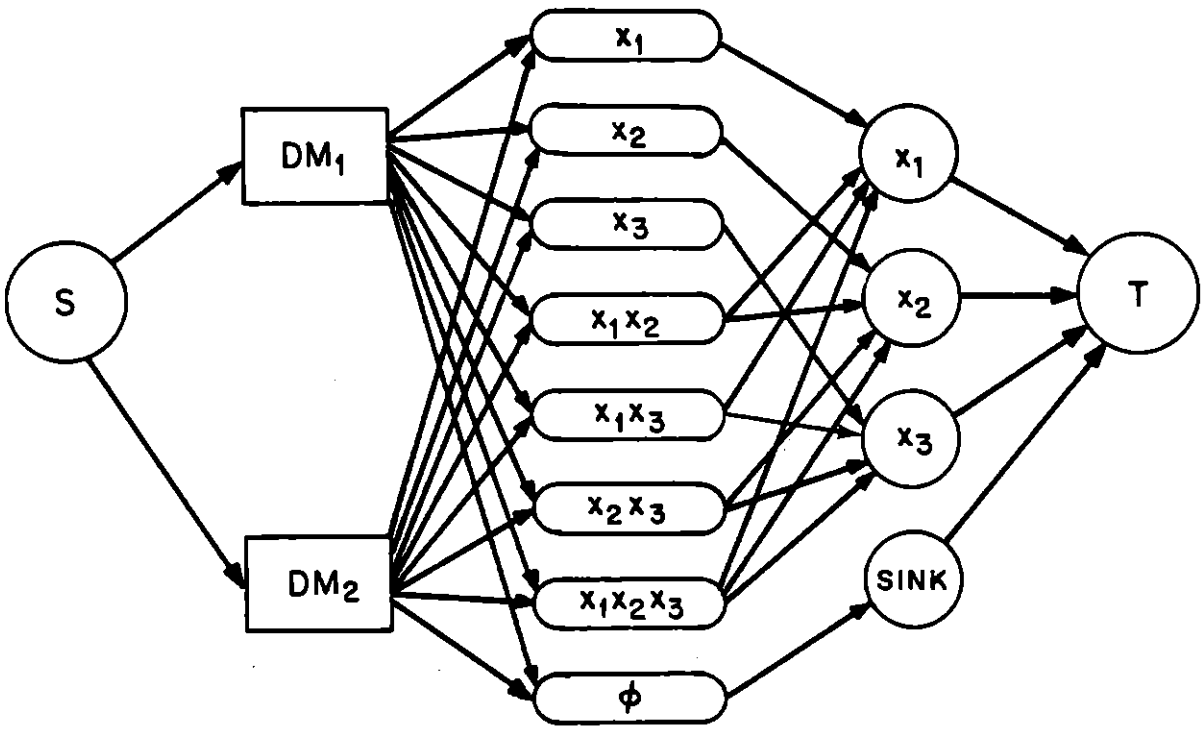


Figure 3.1. Network Structure

not need to appear explicitly in the network formulation. Since the $\bar{\tau}_k^m$'s must be computed before the mathematical program is solved, the $\bar{\tau}_k^m$'s which are feasible are known apriori. In Section 3.2, ^{m} was defined to be the set of partitions of components the m-th DM can process without overload. The feasibility constraints are again imbedded within the partition-selecting set of constraints as follows:

$$B_{Sm} - \sum_{k \in K^m} Y_{mk} = 0 \qquad m = 1,2,\ldots,M \tag{3.14}$$

Since $B_{Sm} = 1$, constraint set (3.14) is almost identical to (3.11a). The two differ only with respect to the set of partitions they each consider.

The coupling constraints, (3.11b), are modified to reflect the network structure of the problem. The general case is illustrated in Figure 3.1. For each partition node k, M arcs defined by Y_{mk} are drawn to it and S_k arcs, D_{kn}, emanate from it. The flow conservation law requires that the total flow entering a node k be equal to the total flow exiting that node. This may be expressed as

$$\sum_{m=1}^{M} Y_{mk} - \sum_{n \in L_k} D_{kn} = 0 \tag{3.15}$$

However, this leads to an inconsistency in the formulation; if partition k is assigned to a DM then $D_{kn} = 1$ for all $n \in L_k$. Therefore,

$$\sum_{n \in L_k} D_{kn} = S_k \tag{3.16}$$

where S_k is the dimension of partition k. However,

$$\sum_{m=1}^{M} Y_{mk} = S_k \tag{3.17}$$

Equality would imply S_k DMs are assigned partition k. In order to resolve this issue and guarantee that at most one DM can be assigned a particular partition, implying $\sum_{m=1}^{M} Y_{mk}$ is 0 or 1, the expression $\sum_{m=1}^{M} Y_{mk}$ is

multiplied by a coefficient S_k. The modified constraint set is:

$$S_k \sum_{m=1}^{M} Y_{mk} - \sum_{n \in L_k} D_{kn} = 0 \qquad k = 1,2,\ldots,K \qquad (3.18)$$

By introducing this multiplier S_k along each of the arcs Y_{mk} the conservation of flow law is not violated. In the network literature, this is referred to as a generalized network (network with gains). [11] This differs from a pure network in that the non-zero entries may take values other than +1 or -1 and the sum of the right hand side (RHS) constants is not necessarily zero. In fact, the RHS constants sum to zero only if M = N. The generalized network formulation is presented in Table 3.2. The violation of the pure network formulation results from the existence of the S_k entries in the matrix with S_k restricted to being a positive integer between 1 and N. Again, the variables Y^*_{mk} which equal one in the optimal solution yield $\underline{z}^m_k$, the assigned partition vector, and π^m_k, the information partitioning matrix. The information structure, Π, is constructed directly.

The objective function (3.12) would be appropriate for minimizing cost in this formulation as well and would accomplish the identical result of minimizing the number of DMs required to process the input vector without overload. Alternatively, maximizing the flow through the network would also achieve this. Each arc, V_{nT}, for all n is capacitated so that each component is processed exactly once. The only uncapacitated arc is $V_{N+1,T}$. Thus, flow would be maximized by having as many DMs as feasible assigned the empty set partition. The other arcs will sum to N regardless of whether one DM processes the entire N-dimensional input vector or N DMs each process one separate component. Hence, the final solution will be optimal in terms of minimizing the number of DMs necessary.

3.4 ALGORITHMS

Although the formulations presented in the previous two sections fall into different classes of mathematical programming problems, the integral properties of the variables included in each formulation require that the

TABLE 3.2 GENERALIZED NETWORK FORMULATION

Let:

B_{Sm} = the amount of flow from source S to the m-th DM

Y_{mk} = the amount of flow from the m-th DM to the k-th partition

D_{kn} = the amount of flow from the k-th partition to the n-th component

V_{nT} = the amount of flow from the n-th component to the terminal node T

Then:

$$-B_{Sm} = -1 \qquad m = 1,2,\ldots,M \qquad (3.19a)$$

$$B_{Sm} - \sum_{k \in \mathbf{K}^m} Y_{mk} = 0 \qquad m = 1,2,\ldots,M \qquad (3.19b)$$

$$S_k \sum_{m=1}^{M} Y_{mk} - \sum_{n \in \mathbf{L}_k} D_{kn} = 0 \qquad k = 1,2,\ldots,K \qquad (3.19c)$$

$$\sum_{k \in \mathbf{K}_n} D_{kn} - V_{nT} = 0 \qquad n = 1,2,\ldots,N+1 \qquad (3.19d)$$

$$V_{nT} = 1; \quad V_{N+1,T} \geq 0 \qquad n = 1,2,\ldots,N \qquad (3.19e)$$

$$Y_{mk} = 0,1 \qquad m = 1,2,\ldots,M; \quad k \in \mathbf{K}^m \qquad (3.19f)$$

$$0 \leq D_{kn} \leq 1 \qquad k = 1,2,\ldots,K; \quad n \in \mathbf{L}_k \qquad (3.19g)$$

algorithms selected to solve these problems incorporate branch-and-bound techniques. [8]

Branch-and-bound is an optimization technique which enumerates potential solutions to the problem, tests if they are feasible and, finally, if they are optimal. An optimal solution is guaranteed to be found. A tree is constructed, each node corresponding to a variable and each branch incident to a given node from below corresponding to possible integer values the variable might assume. A path from the root node to some other node is said to be fathomed if no further exploration from that vertex can improve the current value of the objective function. [8] To accelerate the fathoming process and lessen the number of solutions enumerated, upper and lower bounds on the objective function are calculated. Various procedures have been developed to compute these upper and lower bounds and expedite the search process.

Davis, Kendrick and Weitzman [12] present an efficient branch-and-bound algorithm for solving the MILP formulated in Section 3.2. Glover and Molvey [13] have discussed the computational efficiency resulting from transforming certain 0-1 Integer Programming Problems into Generalized Networks. The GN algorithm used a Last In First Out (LIFO) branch-and-bound strategy. Heuristic solution procedures using this strategy were developed for three cases; dramatic results were achieved. In one case the execution time was reduced from 7 hours to 0.5 hours. In another case the original problem formulation was not computationally infeasible, but use of the GN formulation led to the solution in 10 seconds. In the third case, a problem involving 20,000 nodes and 100,000 arcs was solved in less than 30 minutes.

Review of the literature reveals that other approaches are being researched currently. A relaxation of the MILP based on solving matching problems is one of the most promising approaches that is being explored. [8,14,15]

3.5 DIMENSIONALITY

Regardless of the formulation that is used, the number of variables and constraints that must be considered explicitly is of order 2^N. For any reasonably-sized problem this could prove computationally infeasible. Table 3.3 presents the number of variables and constraints of each type that are involved in the GN formulation.

TABLE 3.3 THE DIMENSIONALITY PROBLEM

VARIABLES	NUMBER OF VARIABLES	
	GENERAL CASE	EXAMPLE (N = 8, M = 3)
B_{Sm}	M	3
Y_{mk}	$M2^N$	768
D_{kn}	$1+N2^{N-1}$	1,025
V_{nT}	1+N	9
Total:	$2+M+N+(2M+N)2^{N-1}$	1,805

CONSTRAINTS	NUMBER OF CONSTRAINTS	
	GENERAL CASE	EXAMPLE (N = 8, M = 3)
(3.19a)	M	3
(3.19b)	M	3
(3.19c)	2^N	256
(3.19d)	N+1	9
(3.19e)	N+1	9
Total:	$2(1+M+N+2^{N-1})$	280

An example is also presented in which the input vector is of dimension eight (N = 8) and the number of available DMs is three (M = 3). For this

relatively small problem 1,805 variables and 280 constraints must be considered. It is clear the dimensionality is most sensitive to the dimension of the input vector.

3.6 SUMMARY

In this chapter, two mathematical programming formulations have been presented for assigning components of the input vector to DMs such that all components are processed without overload. The information structure can be constructed directly. The problem of dimensionality and its sensitivity to the size of the input vector is critical. In the next chapter, simplifying assumptions concerning the DMs and the components are made and their effects on the dimensionality issue are explored.

CHAPTER 4

PARALLEL PROCESSING: DECISION MAKERS AND ORGANIZATIONS

4.1 INTRODUCTION

In the previous chapter, the general formulation of the parallel processing problem was presented. The dimensionality of the general problem can be reduced substantially by making additional assumptions about the properties of the input source and the decision makers. Assumptions that lead to useful special cases include:

a) decision makers have identical processing time functions,

b) components have alphabets with identical probability distributions,

c) decision makers are specialized, i.e., different decision makers can process only certain components of the input vector,

d) specified components are required to be processed together although they are probabilistically independent,

e) specified components must be transmitted to more than one decision maker, i.e., redundancy.

In the following sections, the effects on dimensionality of considering one or more of these assumptions are explored.

4.2 IDENTICAL DECISION MAKERS

4.2.1 Single Group

The first special case considered assumes that in addition to all components being mutually independent, all DMs have identical properties, i.e., they possess identical processing time functions, are not specialized, and have identical costs. Therefore, feasibility of processing a partition of components without overload need only be checked for any one DM rather than for all M DMs. Rather than defining the set of positive integers corresponding to partitions of components that the m-th

DM can process without overload, $\mathbf{K}^m$ for all m, only one $\mathbf{K}$ need be determined. This results in a substantial reduction in the number of computations that must be done - 2^N rather than $M \cdot 2^N$.

The Generalized Network formulation can also be modified so that the number of variables and constraints that must be considered explicitly is reduced substantially. The DMs do not have to be considered separately but instead, a representative DM can be selected. Except for the empty set partition which may be assigned more than once, each other partition can be assigned at most one time to the representative DM. Thus, the number of variables, B_{Sm}, is reduced by (M-1) and the number of variables, Y_{mk}, is reduced by $(M-1)2^N$. No other variables are affected.

The constraint sets of the GN formulation presented in Table 3.2 are modified accordingly; the modified GN formulation is presented in Table 4.1.

TABLE 4.1 GN FORMULATION: IDENTICAL DECISION MAKERS

$$-B_{Sm} = -M \tag{4.1a}$$

$$B_{Sm} - \sum_{k \in \mathbf{K}^m} Y_{mk} = 0 \tag{4.1b}$$

$$S_n Y_{mk} - \sum_{n \in \mathbf{L}_k} D_{kn} = 0 \qquad k \in \mathbf{K}^m \tag{4.1c}$$

$$\sum_{k \in \mathbf{K}'_n} D_{kn} - V_{nT} = 0 \qquad n = 1,2,\ldots,N+1 \tag{4.1d}$$

$$V_{nT} \geq 1; \qquad V_{N+1,T} \geq 0 \qquad n = 1,2,\ldots,N \tag{4.1e}$$

$$Y_{mk} \geq 0, \text{ integer} \qquad k \in \mathbf{K}^m \tag{4.1f}$$

$$D_{kn} = 0,1 \qquad k \in \mathbf{K}^m;\ n \in \mathbf{L}_k \tag{4.1g}$$

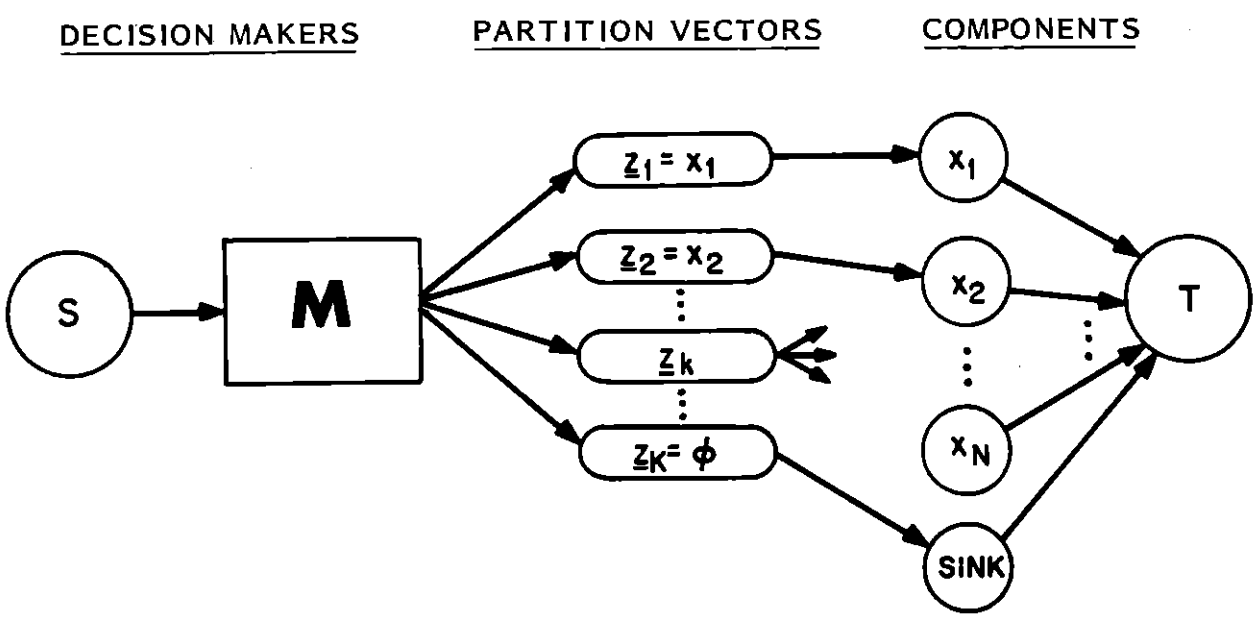
Variable definitions remain the same. The constraint set modifications result in $(M-1)(1+2^N)$ fewer constraints that need to be considered explicitly. Although this also is a substantial reduction, the overall problem is still of order 2^N. This formulation is illustrated in Figure 4.1(a).

The information structure is derived in a similar fashion to that demonstrated in Section 3.4. Each DM is assigned a partitioning matrix, π_k^m corresponding to one of the M partitions assigned to the representative DM, i.e., $Y_{mk} = 1$ for M values of k, $k \in \mathbf{K}$. The assignment of components to the particular DMs is arbitrary since they are identical. If the empty set partition was assigned one or more times, then a corresponding number of DMs are eliminated from the S-E and are not represented in the information structure.

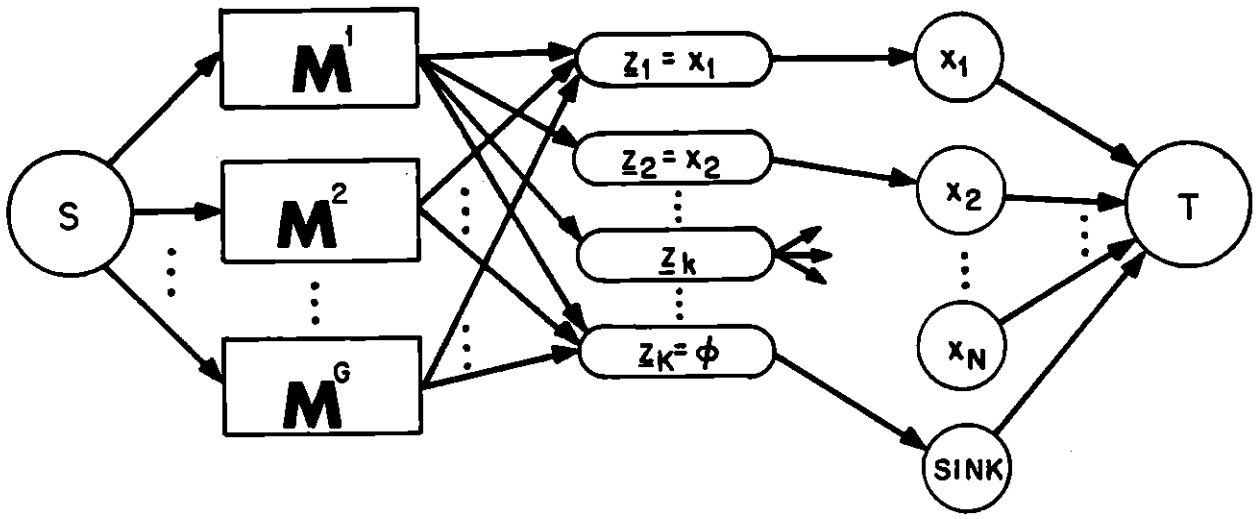
4.2.2 Many Groups

Groups of identical DMs are considered next. Decision makers within a specified group, $\mathbf{M}^g$, g = 1,2,...,G, possess identical processing time functions, are able to process the same types of components and have identical costs. This case differs only slightly from the previous one. The feasibility of processing a partition of components without overload must be checked for G representative DMs, one from each group $\mathbf{M}^g$. Therefore, the set $\mathbf{K}^m$ must be defined for G representative DMs. This also results in a substantial reduction in the number of computations compared to the original problem: $G \cdot 2^N$ computations must be performed.

The Generalized Network formulation of Table 3.2 is again modified to reduce the number of variables and constraints considered explicitly. Following the procedure of the single group case, the g-th representative DM has d^g partitions assigned to him, where d^g is the dimension of $\mathbf{M}^g$. The implicit assumption that any partition, except the empty set partition, can be assigned at most once, holds over all representative DMs. The number of variables B_{Sm} is reduced from M to G, the number of variables Y_{mk} is reduced from $M \cdot 2^N$ to $G \cdot 2^N$, but no other variables are affected.



(a) One Group



(b) Many Groups

Figure 4.1. Identical Decision Makers

The corresponding optimization problem of this Generalized Network is presented in Table 4.2; its objective function is the cost of the DMs comprising the S-E. A cost C_g is assigned to each DM in the g-th group. This cost is assumed positive and fixed, i.e., independent of the size of the partition of components to be processed. The number of partitions processed by the the g-th group of DMs is given by

$$\sum_{k=1}^{K-1} Y_{gk}.$$

The K-th partition (empty set) is excluded since DMs assigned that partition are not included in the S-E. The total cost of using DMs in $\mathbf{M}^g$ for each g is given by

$$C_g \sum_{k=1}^{K-1} Y_{gk}$$

The total cost for all DMs in the S-E is

$$\sum_{g=1}^{G} \sum_{k=1}^{K-1} C_g Y_{gk}.$$

This formulation does not guarantee that the minimum number of DMs will be used however. It may be possible that two inefficient DMs may be cheaper than one very efficient DM in processing the same data in time δ or less. Figure 4.1(b) illustrates this formulation: G representative DMs are enumerated explicitly rather than the M original DMs. The modified constraint sets result in $(M-G)(1+2^N)$ fewer constraints to be considered explicitly. The overall problem, however, remains of order 2^N.

The information structure is determined as in the single group case. Each DM within a group $\mathbf{M}^g$ is assigned a partitioning matrix π_k^m corresponding to one of the d^g partitions assigned to the representative DM of that group, i.e., $Y_{gk} = 1$ for d^g k's, $k \in \mathbf{K}^g$. The generic form of the information structure is

TABLE 4.2 GN FORMULATION: IDENTICAL DM'S

Let:

$\mathbf{M}^g$ = g-th group of identical DMs $\quad g = 1,2,\ldots,G$

d^g = dimension of $\mathbf{M}^g$ $\quad g = 1,2,\ldots,G$

C_g = cost of a Dm in $\mathbf{M}^g$ includeed in the S-E $\quad g = 1,2,\ldots,G$

Then:

$$J = \min \sum_{g=1}^{G} \sum_{k=1}^{K-1} C_g Y_{gk}$$

such that:

$$-B_{Sg} = -d^g \quad g = 1,2,\ldots,G \quad (4.2a)$$

$$B_{Sg} - \sum_{k \in \mathbf{K}^g} Y_{gk} = 0 \quad g = 1,2,\ldots,G \quad (4.2b)$$

$$S_k \sum_{g=1}^{G} Y_{gk} - \sum_{n \in \mathbf{L}_k} D_{kn} = 0 \quad k = 1,2,\ldots,K \quad (4.2c)$$

$$\sum_{k \in \mathbf{K}_n} D_{kn} - V_{nT} = 0 \quad n = 1,2,\ldots,N+1 \quad (4.2d)$$

$$V_{nT} = 1; \quad V_{N+1,T} \geq 0 \quad n = 1,2,\ldots,N \quad (4.2e)$$

$$Y_{gk} = 0,1 \quad g = 1,2,\ldots,G; \; k \in \mathbf{K}^g \quad (4.2f)$$

$$0 \leq D_{kn} \leq 1 \quad k = 1,2,\ldots,K; \; n \in \mathbf{L}_k \quad (4.2g)$$

$$\Pi = (\underbrace{\pi_{z_1}^1, \ldots, \pi_z^{d^1}}_{d^1}, \underbrace{\pi_z^{d^1+1}, \ldots, \pi_z^{d^1+d^2}}_{d^2}, \ldots, \underbrace{\pi_z^{M^*-d^G+1}, \ldots, \pi_z^{M^*}}_{d^G})$$

assuming no representative DM was assigned an empty set partition.

4.3 IDENTICAL PROBABILITY DISTRIBUTION

4.3.1 Single Group

The second special case illustrates that under the assumptions that

a) all components are independent and

b) all components' alphabets are of the same dimension and have identical probability distribution functions,

the problem of selection and assignment of partitions of components to DMs does not require a GN algorithm or any other mathematical programming (MP) algorithm to solve it. A feasible solution which requires the minimum number of DMs can be obtained directly with relatively few computations.

It was shown in Chapter 2 that if the components of a partition vector are mutually independent, then the entropy associated with that partition is equal to the sum of the entropies of each of the components in the partition, i.e.,

$$H(\underset{\sim}{z}_{N'}) \equiv H(x_1, \ldots, x_{N'}) = \sum_{n=1}^{N'} H(x_n). \tag{4.3}$$

From assumption (b), it follows directly that the entropies of each of the components are equal, i.e.,

$$H(x_n) = H_0 \qquad n = 1, \ldots, N' \tag{4.4}$$

Combining equations (4.3) and (4.4), yields

$$H(\underline{z}_{N'}) = \sum_{n=1}^{N'} H(x_n) = \sum_{n=1}^{N'} H_0 = N'H_0 \qquad (4.5)$$

The entropy of any partition of dimension s is equal to sH_0. For example, partitions $\underline{z}_{11} = (x_1, x_2, x_3)$ and $\underline{z}_{12} = (x_1, x_2, x_4)$ both have associated entropies equal to $3H_0$. Rather than computing 2^N entropies, only one entropy need be computed, namely, H_0.

No restrictions have yet been placed on the DMs as to which components they may process. Thus, any component(s) that the m-th DM can process without overload may be assigned to him. Feasibility with respect to mean processing time requires that

$$t^m(s) + c^m H(\underline{z}_s) = st^m + c^m(sH) \leq \delta, \qquad m = 1,2,\ldots,M \qquad (4.6)$$

where it is assumed that

$$t^m(s) = st^m$$

(see eq. (2.20)).

The maximum number of components that the m-th DM can process without overload, s_m^*, is derived from the inequality of (4.6) as follows:

$$st^m + c^m(sH_0) \leq \delta \qquad m = 1,2,\ldots,M$$

$$\Rightarrow \quad s \leq \delta(t^m + c^m H_0)^{-1} \qquad m = 1,2,\ldots,M$$

$$\Rightarrow \quad s_m^* = \lfloor \delta(t^m + c^m H_0)^{-1} \rfloor \qquad m = 1,2,\ldots,M$$

where $\lfloor \cdot \rfloor$ denotes the function that yields the greatest integer less than or equal to its argument.

In order to minimize the number of DMs required to process the input vector, components are assigned to those DMs who can process the greatest number of components in time δ. The s_m^*'s are ranked in order of

magnitude. The quantity s^*_{mf} is defined to be the f-th largest s^*_m, $m = 1,2,\ldots,M$; $f = 1,2,\ldots,M$. If each of two or more DMs can process the same maximum number of components, then the ranking among them is arbitrary.

To determine M^*, the minimum number of DMs necessary to process the input vector, DMs are added until all components are assigned, i.e.,

$$\sum_{f=1}^{M^*} s^*_{mf} \geq N$$

Note that in this formulation there is no distinction in the cost C between DMs of different capabilities.

The DMs corresponding to $f = 1,2,\ldots,M^*$ will be included in the information structure. Each of the first M^*-1 DMs will process partitions of dimension s^*_{mf}. The last DM will process a partition of dimension less than or equal to $s^*_{mM^*}$:

$$N - \sum_{f=1}^{M^*-1} s^*_{mf}$$

Since the components all have alphabets with identical probability distributions and DMs are not restricted as to which components they may process, the number of possible assignments is

$$\frac{N! / \prod_{j=1}^{N} h_j}{(s^*_{m1}!)(s^*_{m2}!) \ldots (s^*_{mM^*}!)}$$

where h_j is the number of groups of dimension j, $j = 1,2,\ldots,N$. Let h_j be unity if no groups of dimension j exist. The generic information structure is:

$$\Pi = (\pi^1_{k_1}, \pi^2_{k_2}, \ldots, \pi^{M^*}_{k_{M^*}})$$

where $\pi^{m}_{k_f}$ is the k_f-th partitioning matrix of dimension s^{*}_{mf} x N and $f = 1,2,\ldots,M^{*}$; $m = 1,2,\ldots,M^{*}$.

An example will best illustrate the substantial reduction in dimensionality. Consider the example presented in Table 3.4 where $M = M^{*} = 3$, $N = 3$. The GN formulation requires a total of 2,085 variables and constraints. Under the simplifying assumptions only $2M + 1 = 7$ computations are required.

4.3.2 Many Groups

Although the assumption that all components have identical probability distributions is not realistic, it does demonstrate the significant reduction in the dimensionality of the problem. A much more realistic assumption is that components can be grouped so that within each group the components possess alphabets with identical probability distributions. This case is considered next.

Let $\mathcal{L}^{g}$, $g = 1,2,\ldots,G$ represent the g-th group of scalar components, components within the group possessing alphabets with identical probability distributions. These groups are disjoint and their union forms the complete set of components contained in the input vector. The assumption of independence among all components is maintained.

Let H_g be the entropy associated with a component in $\mathcal{L}^{g}$. It has been established previously that components with alphabets possessing identical probability distributions have equal entropies. Thus, G distinct entropies must be computed.

Let the dimension of $\mathcal{L}^{g}$ be s^{g}; then

$$\sum_{g=1}^{G} s^{g} = N$$

A representative component, x^{g}, can be selected for each group.

These representative components and the dimension of each group, s^g, are all that is needed to enumerate all possible partitions. Using combinatorial theory it can be shown that the total number of unique partitions K is given by [16]

$$K = \prod_{g=1}^{G} (s^g+1) \tag{4.7}$$

A simple proof of this is that for each group L^g, the number of components that can be included in the partition range from 0 to s^g. Therefore, for each group, there are (s^g+1) ways to include the representative component. It follows that the total number of partitions is K. This is a significant reduction from the 2^N partitions needed to be considered originally.

It is not immediately clear what the generic form is of a set of partitions of a given dimension. It is necessary to develop a methodology to construct all possible partitions for G disjoint groups of like components of dimension s^g, g = 1,2,...,G. Table 4.3 enumerates partitions of components of every possible dimension for an input vector of dimension 8 (N = 8). The rows indicate s_k, the dimension of the partition. The columns headed 0,1,...,8 indicate s^g, the size of a group of like components. Thus an element (i,j) in this matrix indicates the number of groups of like components of size j composing a partition of dimension i. The very last column presents the results for a numerical example in which the number and size of groups of like components are G = 3, s^1 = 3, s^2 = 3, s^3 = 2. These specifications further restrict the partitions that can be constructed. In this example, it would be impossible to construct a 5-dimensional partition which contains 4 like components, since there exist, at most, 3 like components in any group. The last column also indicates the number of partitions of a given dimension composed of the specified number of like components of each type that can be constructed. For example, let x^1, x^2, x^3 be the representative components of L^1, L^2, L^3, respectively. Row 6 of the matrix in Table 4.3 states that 6 three-dimensional partitions containing two like components and one different component can be constructed. These

TABLE 4.3 PARTITION ENUMERATION EXAMPLE:

IDENTICAL COMPONENTS - MANY GROUPS

EXAMPLE: N = 3, G = 3, $S^1 = 3$, $S^2 = 3$, $S^3 = 2$

Dimension of Partition (S_k)	Size of Group (S^g) 0	1	2	3	4	5	6	7	8	Number of Partitions	Total
0	1									1	1
1	0	1								3	3
2	0	2	0							3	
	0	0	1							3	6
3	0	3	0	0						2	
	0	1	1	0						6	
	0	0	0	1						1	9
4	0	4	0	0	0						
	0	2	1	0	0					3	
	0	1	0	1	0					4	
	0	0	2	0	0					3	
	0	0	0	0	1						10
5	0	5	0	0	0	0					
	0	3	1	0	0	0					
	0	2	0	1	0	0					
	0	1	0	0	1	0				2	
	0	1	2	0	0	0				3	
	0	0	1	1	0	0				4	
	0	0	0	0	0	1					9
6	0	6	0	0	0	0	0				
	0	4	1	0	0	0	0				
	0	3	0	1	0	0	0				
	0	2	0	0	1	0	0				
	0	2	2	0	0	0	0				
	0	1	0	0	1	1	0			4	
	0	1	1	1	0	0	0				
	0	0	3	0	0	0	0			1	
	0	0	1	0	1	0	0				
	0	0	0	2	0	0	0			1	
	0	0	0	0	0	0	1				6

TABLE 4.3 PARTITION ENUMERATION EXAMPLE (CONTINUED)

EXAMPLE: N = 8, G = 3, S^1 = 3, S^2 = 3, S^3 = 2

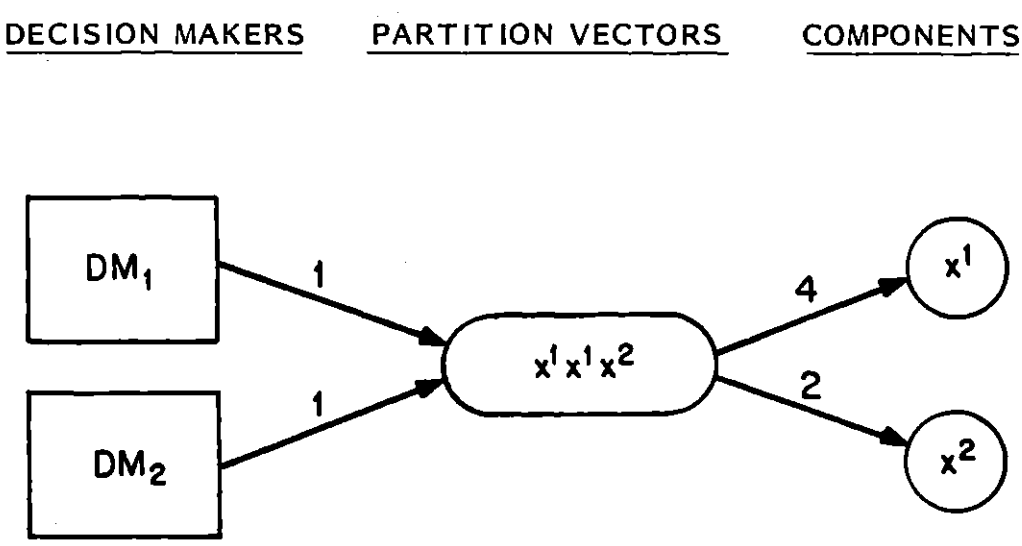
Dimension of Partition (S_k)	Size of Group (S^g) 0	1	2	3	4	5	6	7	8	Number of Partitions	Total
7	0	7	0	0	0	0	0	0			
	0	5	1	0	0	0	0	0			
	0	4	0	1	0	0	0	0			
	0	3	0	0	1	0	0	0			
	0	3	2	0	0	0	0	0			
	0	2	0	0	0	1	0	0			
	0	2	1	1	0	0	0	0			
	0	1	0	0	0	0	1	0			
	0	1	0	2	0	0	0	0		1	
	0	1	3	0	0	0	0	0			
	0	1	1	0	1	0	0	0			
	0	0	2	1	0	0	0	0		2	
	0	0	1	0	0	1	0	0			
	0	0	0	1	1	0	0	0			
	0	0	0	0	0	0	0	1			3
8	0	8	0	0	0	0	0	0	0		
	0	6	1	0	0	0	0	0	0		
	0	5	0	1	0	0	0	0	0		
	0	4	0	0	1	0	0	0	0		
	0	4	2	0	0	0	0	0	0		
	0	3	0	0	0	1	0	0	0		
	0	3	1	1	0	0	0	0	0		
	0	2	0	0	0	0	1	0	0		
	0	2	0	2	0	0	0	0	0		
	0	2	3	0	0	0	0	0	0		
	0	2	1	0	1	0	0	0	0		
	0	1	0	0	0	0	0	1	0		
	0	1	0	1	1	0	0	0	0		
	0	1	1	0	0	1	0	0	0		
	0	0	4	0	0	0	0	0	0		
	0	0	2	0	1	0	0	0	0		
	0	0	1	2	0	0	0	0	0	1	
	0	0	1	0	0	0	1	0	0		
	0	0	0	1	0	1	0	0	0		
	0	0	0	0	2	0	0	0	0		
	0	0	0	0	0	0	0	0	1		1

partitions would be (x^1,x^1,x^2), (x^1,x^1,x^3), (x^2,x^2,x^1), (x^2,x^2,x^3), (x^3,x^3,x^1), (x^3,x^3,x^2). Note that the sum of the elements in this last column equals 48 which is consistent with eq. (4.7):

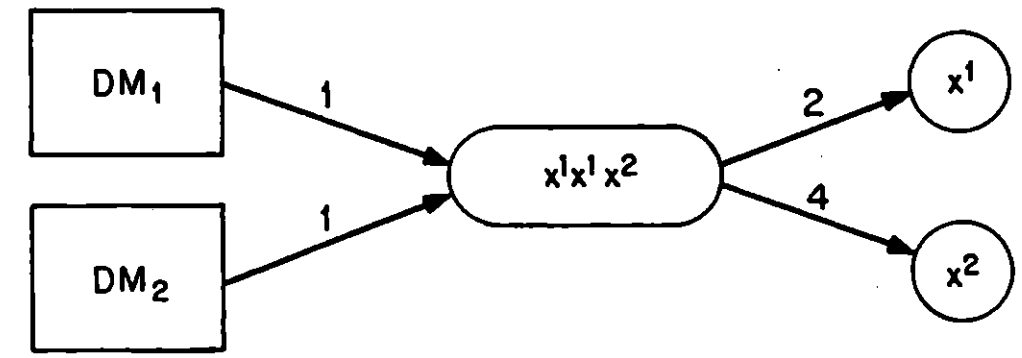
$$K = \prod_{g=1}^{3} (s^g+1) = (3+1)(3+1)(2+1) = 48$$

While the number of partitions to be considered explicitly has been reduced, the problem of selecting and assigning these partitions of components to various DMs can no longer be formulated as a Generalized Network Problem. This is a consequence of the additional constraints that must be imposed to guarantee that all components of the input vector are processed. In particular, a partition k may now be assigned to more than one DM, i.e., $\sum_{m=1}^{M} Y_{mk}$ may be strictly greater than one. The flow on the arcs D_{kn}, where $n \in L_k$, cannot be bounded effectively from above since the exact number of DMs assigned to process this partition is not known. Therefore, the distribution of the flow along the arcs D_{kn} may be incorrect. This could cause the true number of representative components processed by assigning partition k to a DM to differ from the number indicated. The partitions of components necessary to be processed may be inaccurate. Figure 4.2 illustrates, for 2 DMs, what the true distribution of flow along a section of a network should be and what a feasible, although incorrect, flow could be, if additional constraints were not imposed.

A true distribution of flow guarantees that for every unit of flow entering the partition node exactly one unit of flow exits to each component node comprising the partition node. For every unit of flow into (x^1,x^1,x^2) exactly two units of flow exit to x^1 and one unit of flow exits to x^2. However, the current constraints can only guarantee that three units of flow exit for every unit of flow that enters. This could result in an incorrect distribution of flow, i.e., one unit may exit to x^1 and two units to x^2. Figure 4.3 illustrates, for two DMs, the network structure when two representative components are selected and the corresponding partitions are enumerated.



(a) True Distribution of Flow



(b) Feasible Distribution of Flow

Figure 4.2. Flow Distribution Using GN Constraints

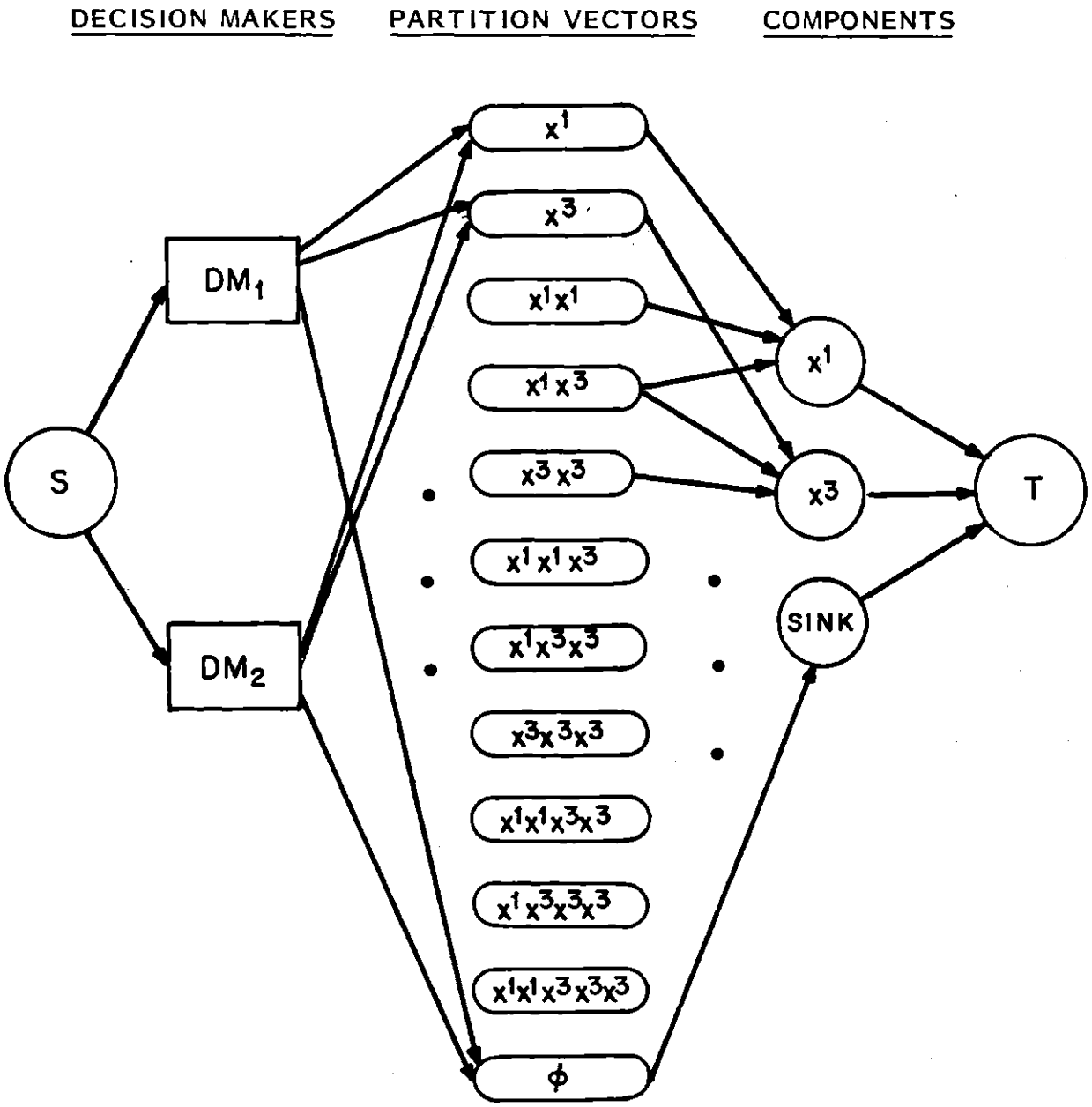


Figure 4.3. Groups of Identical Components

The loss of the GN formulation is disappointing only because of the efficient algorithms which exist to solve GNs. Several alternative formulations exist, however. The grouping of components possessing alphabets with identical probability distributions can be eliminated and 2^N distinct partitions can be enumerated. This problem can be formulated as a GN. The dimensionality issue persists but overload need only be checked $M \prod_{g=1}^{G} (s^g+1)$ times rather than $M \cdot 2^N$ times.

Another alternative is to formulate this as an integer program. This formulation resembles the structure of the GN formulation of Table 3.2, but now the partitions include only representative components from each group L^g. This set of partitions is represented by K'; the component set L' of dimension G consists of these representative components only. To guarantee correctness of the flows D_{kg} along each arc, from partition node $k \in K'$ to representative component node $g \in L'$, constraints which enforce the appropriate distribution of flow among these arcs must be included.

Consider all arcs $D_{k'g}$ directed from partition k'; i.e., $g \in L_{k'}$. For every unit of flow which enters partition node k', $s_{k'}$ units exit. The distribution of these $s_{k'}$ units of flow among the arcs $D_{k'g}$ is in a certain proportion. Let $s_{k'}$ equal the number of representative components g in partition k'. Clearly,

$$\sum_{g \in L_{k'}} s^g_{k'} = s_{k'}$$

Select the maximum $s^g_{k'}$, say $s^{g'}_{k'}$. If a tie exists, select among these. Let

$$\sigma^g_{k'} = s^g_{k'}/s^{g'}_{k'} \tag{4.8}$$

It then follows that

$$\sigma^g_{k'} D_{k'g'} = D_{k'g} \qquad k' \in K';\ g \in L_{k'} \tag{4.9}$$

Adding these equations for each partition node will guarantee the appropriate distribution of flow. However, the number of additional

constraints increases by a factor of 2^G. To avoid this increase, these additional constraints can be imbedded into the formulation.

Constraint set (3.10c) of the GN formulation of Table 3.1 is modified so that each term, $D_{k'g'}$, $g \in \mathcal{L}_{k'}$, is replaced with $\sigma^g_{k'} D_{k'g'}$, $g \in \mathcal{L}_{k'}$, $k' \in \mathbf{K}'$. The modified constraint set is:

$$\sum_{m=1}^{M} Y_{mk'} - D_{k'g'} \sum_{g \in \mathcal{L}_{k'}} \sigma^g_{k'} = 0 \qquad k' = 1,2,\ldots,K \qquad (4.10)$$

Constraint set (3.10 d) is modified in an identical manner.

$$\sum_{k' \in \mathbf{K}'} \sum_{g} \sigma^g_{k'} D_{kg'} - V_{g'I} = 0 \qquad g = 1,2,\ldots,G \qquad (4.11)$$

The violation of the network formulation occurs because within these two sets of constraints at least one variable will appear more than two times. Recall that a property of network formulations is that each variable appears in at most two constraint equations.

Table 4.4 presents the integer programming formulation of this problem in a format which closely resembles the formulation in Table 3.1. Of course, constraints (4.12a) and (4.12f) can be eliminated and B_{Sm} set equal to 1, $m = 1,2,\ldots,M$, in constraint set (4.12b) and $V_{g'I}$ set equal to $s^{g'}$ in constraint set (4.12d).

A branch-and-bound algorithm is sufficient for solving this problem. However, since the matrix A (or matrices A_1, A_2) is very sparse, more efficient algorithms may be devised for solving this particular problem. It is worth noting that network algorithms are generally the most efficient. If the number of groups G is comparable to the total number of components, it may still be advantageous to formulate the problem directly as a GN and not attempt to reduce the number of partitions considered explicitly.

If the GN formulation is used to solve the problem, the information

TABLE 4.4 GN FORMULATION: IDENTICAL COMPONENTS - MANY GROUPS

Let:

z_k^* = k-th partition vector composed of representative components

L^g = g-th group of components, components within a group possessing alphabets with identical probability distributions

L_k = set of representative components in partition z_k^*

S^g = dimension of L^g, $g = 1,2,\ldots,G$

s_k^g = number of times a representative component from group L^g appears in partition k, $g = 1,2,\ldots,G$; $k = 1,2,\ldots,K$

$\mathbf{K}_g$ = set of partitions which contain the g-th representative component

$\sigma_{k'}^g = s_{k'}^g / s_{k'}^{g'}$

Then:

$$-B_{Sm} = -1 \quad m = 1,2,\ldots,M \quad (4.12a)$$

$$B_{Sm} - \sum_{k' \in \mathbf{K}^m} Y_{mk'} = 0 \quad m = 1,2,\ldots,M \quad (4.12b)$$

$$\sum_{m=1}^{M} Y_{mk'} - D_{k'g'} \sum_{g \in L_{k'}} \sigma_{k'}^g = 0 \quad k' = 1,2,\ldots,K \quad (4.12c)$$

$$\sum_{k' \in \mathbf{K}_{g'}} \sigma_{k'}^g D_{k'g'} - V_{g'T} = 0 \quad g' = 1,2,\ldots,G \quad (4.12d)$$

$$V_{g'T} = S^{g'} \quad V_{0,T} \geq 0 \quad g' = 1,2,\ldots,G \quad (4.12e)$$

$$Y_{mk'} = 0,1 \quad m = 1,2,\ldots,M; \; k' \in \mathbf{K}^m \quad (4.12f)$$

$$D_{k'g'} \geq 0, \text{ integer} \quad k' \in \mathbf{K}', \; g' \in L_{k'} \quad (4.12g)$$

structure is obtained directly as shown in Section 3.4. If the integer programming formulation is used, a modified version of the procedure presented in Section 3.2 is used. Each variable Y_{mk} which equals unity indicates that the m-th DM is assigned partition k composed of representative components. Each representative component, x^g, must be replaced by a component from the set ι^g until all assigned partitions are composed of the input vector components, not the representative components. The assignment of the particular components within a group ι^g to the corresponding partitions is arbitrary.

For example, if $\iota^1 = (x_1, x_2, x_3, x_4)$, $\iota^2 = (x_5, x_6)$, and $\underline{z}_k = (x^1, x^1, x^2)$ is the partition of representative components assigned and processed by both DM_1 and DM_2 then one possible solution is:

$$\pi^1_{*k_1} = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \pi^2_{*k_1} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

so that $\Pi = (\pi^1_{*k_1}, \pi^2_{*k_2})$. Another possible solution is:

$$\pi^1_{*k_3} = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \qquad \pi^2_{*k_4} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

so that $\Pi = (\pi^1_{*k_3}, \pi^2_{*k_4})$.

4.4 EXAMPLES

Several examples are presented to illustrate more clearly how the dimensionality of the problem is reduced significantly under combinations of the assumptions presented in the previous sections.

The first example to be considered illustrates the development in Section 4.2.1 while the second illustrates the case of groups of identical DMs, Section 4.2.2. The third example demonstrates the effect of components with identical probability distribution.

Example 4.1

This is the simplest possible case. Assume that

a) all components are of dimension one and are independent,

b) all components' alphabets possess identical probability distributions, and

c) all DMs are identical with respect to their processing time function.

Let the number of components be four (N = 4) and the parameters of the processing function be t(s) = 0.25s and c = 1.0. This example illustrates, under these simplifying assumptions, that the problem of selection and assignment of partitions of components to DMs does not require a GN algorithm or any other MP algorithm to solve it. A feasible solution which requires the minimum number of DMs can be obtained with a small number (3) of computations.

Entropies must be computed regardless of the method of solution. As was shown in Section 4.2, if all components are assumed independent and their alphabets possess identical probability distributions, then the entropy of any partition of dimension s is equal to s H_0, where

$$H_0 = H(x_n): \quad n = 1, 2, 3, 4$$

As was shown in Section 4.3, if all DMs are assumed identical, the processing time feasibility of a partition of components need only be checked for one DM.

The procedure for determining M^*, the minimum number of DMs necessary to process the entire input vector, is analogous to that presented in Section 4.2. However, the additional assumption of identical DMs eliminates the need to rank order the maximum number of components each DM can process in time δ or less; it is equal for all m (i.e., $s_m^* = s^*$) and thus s^* is computed only once. It can be found by solving the feasibility constraint on the processing time

$$\bar{\tau}_k^m \leq \delta \text{ , where } \delta = 7, \qquad (4.13)$$

for s^*.

$$\bar{\tau}_k^m = 0.25s + 1(sH_0) \leq 7$$

$$\Rightarrow s \leq 7(0.25 + H_0)^{-1}$$

$$\Rightarrow s^* = \lfloor 7(0.25 + H_0)^{-1} \rfloor$$

where $\lfloor \cdot \rfloor$ is defined as before.

If there are four elements in the alphabet of each component and if they are equiprobable, then

$$H_0 = -4(1/4 \log_2 1/4) = \log_2 4 = 2$$

It follows that the maximum value for s^* is three ($s^* = 3$) and, therefore, each DM can process at most three components. To find the minimum number of DMs, M^*, necessary to process all the components exactly once, the following expression is used,

$$M^* = \lceil N/s^* \rceil \qquad (4.14)$$

where $\lceil \cdot \rceil$ is defined as the smallest integer larger than or equal to the argument. Thus

$$M^* = \lceil 4/3 \rceil = 2$$

Two generic solutions exist. In the first one, one DM processes three components while the fourth component is sent to the other DM. The exact partitions, indicating which components are processed by which DM, need not be specified because the DMs are identical and the components are interchangeable with respect to who processes what component(s).

The number of possible assignments of three components to one DM and one component to the other DM is 2 (obtained from 4!/2(3!1!)) when no

distinctions are made between DMs. The generic information structure is

$$\Pi = (\pi^1_{k_{s_1}}, \pi^2_{k_{s_2}})$$

where $\pi^m_{k_{s_m}}$ is the k_{s_m}-th partitioning matrix of dimension $s_m \times 4$ with $m = 1,2$, and $s_m = 1,3$, $s_1 = s_2$.

Decision Maker 1 may be assigned $\underline{z}_{11} = (x_1, x_2, x_3)$ and DM_2 assigned $\underline{z}_4 = (x_4)$ to process. The corresponding partitioning matrices are

$$\pi^1_{11} = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \pi^1_4 = [0 \quad 0 \quad 0 \quad 1]$$

The corresponding information structure is

$$\Pi = (\pi^1_{k_{11}}, \pi^2_{k_4})$$

In the second generic solution each DM receives two components. For example, DM_1 receives the partition $\underline{z}_5$ containing components (x_1, x_2) and DM_2 receives the partition $\underline{z}_{10}$ containing components (x_3, x_4). The corresponding information structure is

$$\Pi = (\pi^1_{k_5}, \pi^2_{k_{10}}).$$

The number of possible assignments that can be considered is 3 (obtained from $4!/2(2!2!)$).

Feasible solutions, using the minimum number of DMs have been derived which involved:

1) computing one entropy
2) solving for s^* using the feasibility condition (4.13)
3) solving for M^* using the condition (4.14)
4) assigning the N components to the M^* DMs subject to the condition that each DM processes at most s^* components.

Table 3.3 shows that 1,805 variables and 280 constraints must be considered in solving the general problem with N = 8 and M = 3. The simplifying assumptions of this example have enabled the number of constraints to be reduced from 280 to 2.

Example 4.2

Assume a relaxation of assumption (c) of Example 4.1 so that groups of DMs can be formed, DMs within each group possessing identical processing time functions. Maintain assumptions (a) all components are mutually independent and have dimension one and (b) all components possess alphabets with identical probability distributions. Let the number of components be seven (N = 7), the number of groups of DMs be two (G = 2), the number of available DMs be four (M = 4), with two available DMs per group ($d^g = 2$, g = 1,2).

Let t(s) = 0.25s and c = 1.0 be the parameters of the processing time function associated with DMs in $\mathbf{M}^1$. Let t(s) = 0.50s and c = 1.75 be the parameters of the processing time function associated with DMs in $\mathbf{M}^2$. This example illustrates that under these assumptions, the problem of selection and assignment of partitions of components to DMs still need not require any MP algorithm to solve it. A feasible solution which requires the minimum number of DMs can be derived from a small number of computations (approximately 2G+1).

Although DMs between groups vary with respect to the number of components they can process in time δ, it is assumed that the costs associated with each DM are identical. The objective is to process all components using the minimum number of DMs.

Again, only one entropy, H_0, needs to be computed as a result of the identical probability distribution assumption for all components:

$$H_0 = H(x_n) = 2, \qquad n = 1,2,\ldots,7$$

as was presented in Example 4.1. The entropy of any partition of

dimension s is equal to $s\,H_0 = 2s$ since the components are independent.

DMs in $\mathbf{M}^1$ have an average processing time function

$$\bar{\tau}^{\mathbf{M}^1} = 0.25s + sH_0 = (2.25)s$$

while for those in $\mathbf{M}^2$,

$$\bar{\tau}^{\mathbf{M}^2} = 0.50s + 1.75\ sH_0 = 4s$$

A DM in group $\mathbf{M}^g$ can process a partition of components without overload if

$$\bar{\tau}^{\mathbf{M}^g} \leq \delta = 7, \qquad g = 1,2$$

Following the procedure of Example 4.1, the maximum number of components that can be processed by a DM in group $\mathbf{M}^g$, i.e., s_g^*, can be computed.

$$s_1^* = 3$$

$$s_2^* = 1$$

In order to minimize the number of DMs required to process the input vector, components are assigned to those DMs who can process the greatest number of components in time δ. Since DMs within a group have equal processing capabilities, the s_g^* are ranked in order of magnitude. The maximum number of components a DM in group $\mathbf{M}^g$ can process is denoted by s_{gf}^* and this number is the f-th largest, $f = 1,2,\ldots,G$. Thus $s_{11}^* = 3$, $s_{22}^* = 1$. If two or more groups have equal s^*, then they are ranked according to the dimension of each group from largest to smallest. If the group dimensions are also equal, then the ranking is arbitrary among these groups; they can even be combined into one larger group.

The maximum number of components that could be assigned to any group of DMs, $\mathbf{M}^g$, is $d^g s_g^*$, where d^g is the dimension of the group g. Thus DMs in $\mathbf{M}^1$ can process a maximum of 6 components and DMs in $\mathbf{M}^2$, can process a

maximum of 2 components. In this example, 6 components are assigned to the DMs in $\mathbf{M}^1$, each DM processing 3 components. The remaining component is assigned to one of the two DMs in $\mathbf{M}^2$ while the second DM will remain idle.

The information structure includes both DMs from $\mathbf{M}^1$ and one DM from $\mathbf{M}^2$. Without loss of generality, since the DMs in $\mathbf{M}^2$ are identical, DM_3 is selected.

The number of assignments of two three-dimensional partitions and one one-dimensional partition is 70 obtained from

$$7!/2(3!3!1!) = 70$$

One possible solution assigns

$$\underline{z}_{29} = (x_1, x_2, x_3) \text{ to } DM_1$$

$$\underline{z}_{30} = (x_4, x_5, x_6) \text{ to } DM_2$$

$$\underline{z}_{7} = (x_7) \text{ to } DM_3;$$

the corresponding partition matrices are

$$\pi^1_{29} = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \qquad \pi^2_{30} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\pi^3_7 = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

and the information structure is

$$\Pi = (\pi^1_{29}, \pi^2_{30}, \pi^3_7)$$

As stated earlier, five (2G + 1 = 5) computations were involved in solving this problem, if the assignment of specific components was not considered. Had this problem been formulated as a GN, 982 variables and constraints would have had to be taken into account.

Example 4.3

Assume a relaxation of assumption (b) of Example 4.1 so that groups of components can be formed, components within each group possessing alphabets with identical probability distributions. Assumptions (a) all components are mutually independent and of dimension one and (c) all DMs are identical are maintained.

Let the number of components be eight (N = 8), the number of groups of components be three (G = 3), with $s^1 = 3$, $s^2 = 3$ and $s^3 = 2$ and the number of available DMs be three (M = 3). Under these simplifying assumptions only one set, $\mathbf{K}^m$, need be defined. The dimension of the set is $\prod_{g=1}^{G} (s^g+1) = 48$. Therefore, only 48 partitions of representative components need be checked for overload.

Example 4.4

Assume a relaxation of assumptions (b) and (c) of Example 4.1 so that groups of components can be formed, components within each group processing alphabets with identical probability distributions, and groups of DMs can be formed, DMs within a group possessing identical processing time functions. The assumption that all components are mutually independent and of dimension one is maintained.

As in the previous example, let N = 8, G = 3, $s^1 = 3$, $s^2 = 3$ and $s^3 = 2$. Let the number of available DMs be four (M = 4) and the number of groups of DMs be two (G' = 2) so that $d^1 = 2$ and $d^2 = 2$. Under these simplifying assumptions only two sets $\mathbf{K}^m$ need be defined. The 48 possible partitions of components must be checked for overload by a representative DM from each group. The major computational savings occur in defining the sets $\mathbf{K}^m$.

4.5 SPECIALIZATION AMONG DECISION MAKERS

Decision makers often develop a specialization in a particular area of an overall task. An advantage of this is average processing times can be reduced as one acquires more expertise in a given area. Another form of specialization is that of machines which are often limited with respect to the types of data they can process. Specialization can be thought of in the context of this work as a constraint that allows the DM to process only certain components of data.

Assume that G groups of DMs exist, identical DMs within each group $\mathbf{M}^g$, and that each of these groups is specialized and can be assigned only components from a set of components $\mathbf{L}^g$, g = 1,...,G. Assume all components within a group $\mathbf{L}^g$, possess alphabets with identical probability distributions. Assume further that a one-to-one correspondence exists between the group of identical DMs, $\mathbf{M}^g$, and the group of components it can be assigned to process, $\mathbf{L}^g$, for all g.

It was shown in Section 4.3 that if groups exist consisting of components possessing alphabets with identical probability distributions then only one entropy need be computed for each group. Let H_g equal the entropy of a component in $\mathbf{L}^g$, g = 1,...,G. It was also shown in Section 4.2 that the assumption of identical DMs within a group requires that only one processing time feasibility constraint need be satisfied for each group. The assumption of a one-to-one correspondence between an $\mathbf{L}^g$ and $\mathbf{M}^g$ for all g allows the problem to be decomposed from one very large GN to G smaller decoupled GN formulations.

Figures 4.4 and 4.5 illustrate the decoupling effect for the case G = 2, $\mathbf{L}^1 \equiv (x_1, x_2)$, $\mathbf{L}^2 \equiv (x_3, x_4)$. One coupling constraint exists in the complete GN formulation,

$$Y_{1k_{16}} + Y_{2k_{16}} - D_{k_{16}N+1} = 0.$$

Since the empty set partition is only concerned with DMs not processing inputs, two empty set partitions can be created, one for each group of

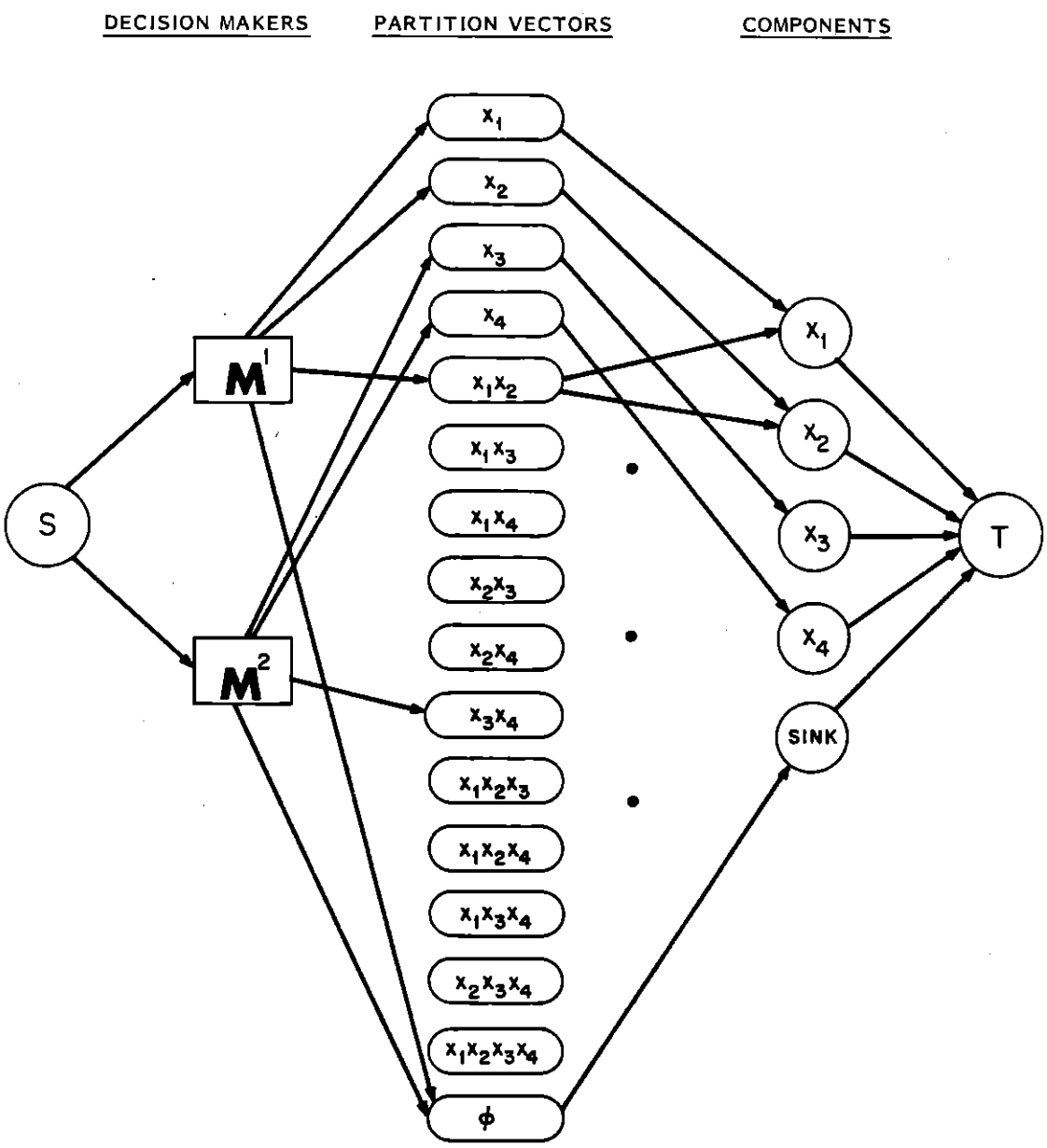


Figure 4.4. Specialization: Original Network

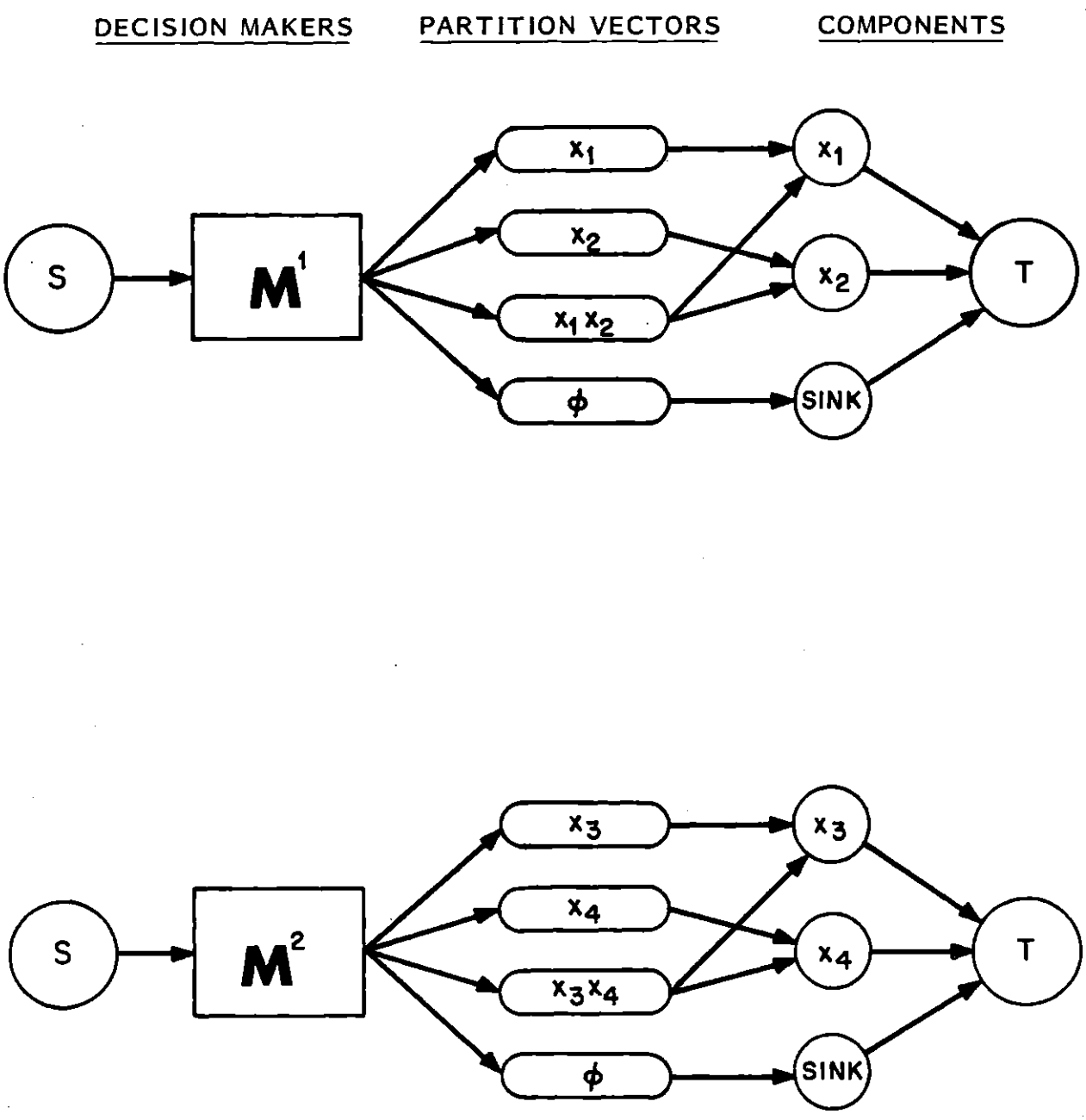


Figure 4.5. Specialization: Decoupling Effect

DMs. The problem is now completely decoupled. A decoupled GN formulation is presented in Table 4.5 for $(\mathbf{L}^1, \mathbf{M}^1)$. Variable definitions remain as before with M^* being the minimum number of DMs necessary to process the entire input vector. It can be obtained by summing M_g^*, $g = 1,2,\ldots,G$. the minimum number of DMs required to process the components in each decoupled problem

$$M^* = \sum_{g=1}^{G} M_g^*$$

It is not necessary to solve G Generalized Network problems to find M_g^* for each g. Each decoupled problem is identical in assumptions and format to the problem presented in Example 4.1 of Section 4.4. The procedure for solving for M^* in that example can be used to solve for M_g^* for each g. The information structures are constructed similarly.

As was shown in Example 4.1, the dimensionality of the problem is reduced significantly under these simplifying assumptions. The number of constraints is reduced from $2(1+M+N+2^{N-1})$ to $2G+1)$. For the case where $N = 10$, $M = 8$, and $G = 4$ this implies a reduction from 1,062 constraints to 9.

If the assumption of components possessing alphabets with identical probability distributions is removed, then 2^{d^g} partitions of components must be constructed for each $(\mathbf{L}^g,\mathbf{M}^g)$ group, and $\mathbf{K}^g$, the set of partitions of components that can be processed without overload by the DMs in $\mathbf{M}^g$, must be defined. While G GNs must be solved, the dimensionality reduction is still so great that the cumulative time required to solve each of these smaller GNs is still substantially less than that required to solve the one larger GN.

Table 4.6 presents a comparison of the dimensionalities of one large GN formulation and G smaller GNs. The dimensions of the component group $\mathbf{L}^1, \mathbf{L}^2, \mathbf{L}^3, \mathbf{L}^4$ are 3, 2, 2, and 3, respectively. The dimension of each group $\mathbf{M}^g$ of DMs is 2 for all g. This table reveals that rather than formulating a problem which involves 13,332 variables and 1,062 constraints, four

TABLE 4.5 GN FORMULATION: SPECIALIZATION - DECOUPLED PROBLEM

$$
\begin{array}{lllllllll}
-B_{S1} & & & & & & & = & -2 \\
B_{S1} & -Y_{1k_1} & -Y_{1k_2} & -Y_{1k_5} & -Y_{1k_{16}} & & & = & 0 \\
 & Y_{1k_1} & & & & -D_{k_1 1} & & = & 0 \\
 & & Y_{1k_2} & & & -D_{k_2 2} & & = & 0 \\
 & & & 2\,Y_{1k_5} & & -D_{k_5 1} & -D_{k_5 2} & = & 0 \\
 & & & & Y_{1k_{16}} & & -D_{k_{16} N+1} & = & 0 \\
 & & & & & D_{k_1 1} + D_{k_5 1} & -V_{1T} & = & 0 \\
 & & & & & D_{k_2 2} + D_{k_5 2} & -V_{2T} & = & 0 \\
 & & & & & & D_{k_{16} N+1} \quad -V_{N+1,T} & = & 0 \\
 & & & & & & V_{1T} & \geq & 1 \\
 & & & & & & V_{2T} & \geq & 1 \\
 & & & & & & V_{N+1,T} & \geq & 0
\end{array}
$$

$Y_{mk} = 0,1 \qquad m = 1,2,\ldots,M';\ k \in \mathbf{K}^m$

$D_{km} = 0,1 \qquad k \in \mathbf{K}^m;\ n \in \mathbf{L}_k$

TABLE 4.6 SPECIALIZATION: DIMENSIONALITY COMPARISON

N = 10 $d^1 = 3,\ d^2 = 2,\ d^3 = 2,\ d^4 = 3;\ G = 4$

M = 8 $d_1 = d_2 = d_3 = d_4 = 2$

NUMBER OF VARIABLES

				EXAMPLES			
VARIABLES	GENERAL CASE	EXAMPLE	SPECIALIZATION	GN1	GN2	GN3	GN4
B_{Sm}	M	8	1	1	1	1	1
Y_{mk}	$M\ 2^N$	8,192	2^{d^g}	8	4	4	8
D_{kn}	$2^{N-1}+1$	5,121	$(2^{d^g-1}d^g)+1$	13	5	5	13
V_{nT}	N+1	11	d^g+1	4	3	3	4
Total:	$(2M+N)2^{N-1}+M+N+2$	13,332	$(2+d^g)2^{d^g-1}+d^g+3$	26	13	13	26

NUMBER OF CONSTRAINTS

				EXAMPLES			
CONSTRAINTS	GENERAL CASE	EXAMPLE	SPECIALIZATION	GN1	GN2	GN3	GN4
(3.19a)	M	8	1	1	1	1	1
(3.19b)	M	8	1	1	1	1	1
(3.19c)	2^N	1,024	2^{d^g}	8	4	4	8
(3.19d)	N+1	11	d^g+1	4	3	3	4
(3.19e)	N+1	11	d^g+1	4	3	3	4
Total:	$2(1+M+N+2^{N-1})$	1,062	$2(2+d^g+2^{d^g}-1)$	18	12	12	18

decoupled GNs are formulated, each with at most 26 variables and 18 constraints. The larger GN would require approximately 10 times more computer time using the Network Code presented by Glover, Hultz and Klingman than each of the smaller GNs would require. [9]

4.6 PRESPECIFIED GROUPING OF COMPONENTS

Although components have been assumed mutually independent, the case could arise that certain components are still required to be processed together. These restrictions are imposed by forces in the organization. A DM may be assigned other components in addition to a specified group provided that the additional components do not overload him.

The concept of requiring components to be processed together was first presented in Chapter 2 under the guise of dependence. Now this concept is extended further. It should be kept in mind, however, that the dimensionality reductions resulting from the restrictions imposed are conceptually similar to the dimensionality reductions resulting from the dependence among components established in Chapter 2.

Let $L[g)$ represent the g-th group of components required to be processed together, $g = 1,2,\ldots,G$, and let $d[g)$ equal its dimension. Supercomponents $x[g)$, $g = 1,2,\ldots,G$ are again constructed. Each of these components consists of a group of individual components that are required to be processed together.

Now, however, because of the mutual independence among components, the construction of the new alphabets and their associated probabilities for each supercomponent can be done fairly easily. The entropy associated with each of these supercomponents, $H(g)$, is the sum of the entropies of the individual components comprising it, i.e.,

$$H(g) = \sum_{n \in L(g)} H_n .$$

The dimension of each supercomponent, $s[g)$, is equal to the sum of the dimensions of each of the components in the corresponding group $L[g)$,

i.e.,

$$s(g) = \sum_{n \in \iota(g)} s(n), \qquad g = 1,2,\ldots,G.$$

The set of components in the input vector ι is redefined to include the supercomponents, rather than the individual components comprising each $\iota(g)$. The dimension of ι is G.

The number of partitions that must be considered explicitly is reduced substantially (from 2^N to 2^G). Thus, the number of computations involved in defining K^m for each DM is also reduced. When determining whether the m-th DM can process partition $\underline{z}'_k$ without overload the true dimension of the partition $s(g)$, must be used and not the number of components comprising $\underline{z}_k$, i.e.,

$$\bar{\tau}^m_{k'}(s(g)) \leq \delta$$

The dimensionality of the GN formulation is reduced substantially. Figure 4.6 illustrates this when the input vector has dimension four but two groups of two components each are required to be processed together. The number of possible partitions reduces from 2^4 to 2^2, a reduction of 75 percent. It follows directly that the number of variables, Y_{mk} and D_{kn} is also reduced since both are directly related to the possible partitions: $(M \cdot 2^G)$ Y_{mk} variables and $(G \cdot 2^{G-1}+1)$ D_{kn} variables. The GN formulation is identical in structure to that presented in Table 4.1. It differs only in that K now equals 2^G and ι is the set of redefined components.

Table 4.7 compares the number of variables and constraints required with and without prespecified groupings of components. The number of variables is reduced by almost 75 percent and the number of constraints by 40 percent.

The information structure can be derived from the solution to this modified GN. Since the partitions are composed of supercomponents, however, the actual components comprising each supercomponent must be included in the information structure. If supercomponents $x_{g'}$ and $x_{g''}$ are

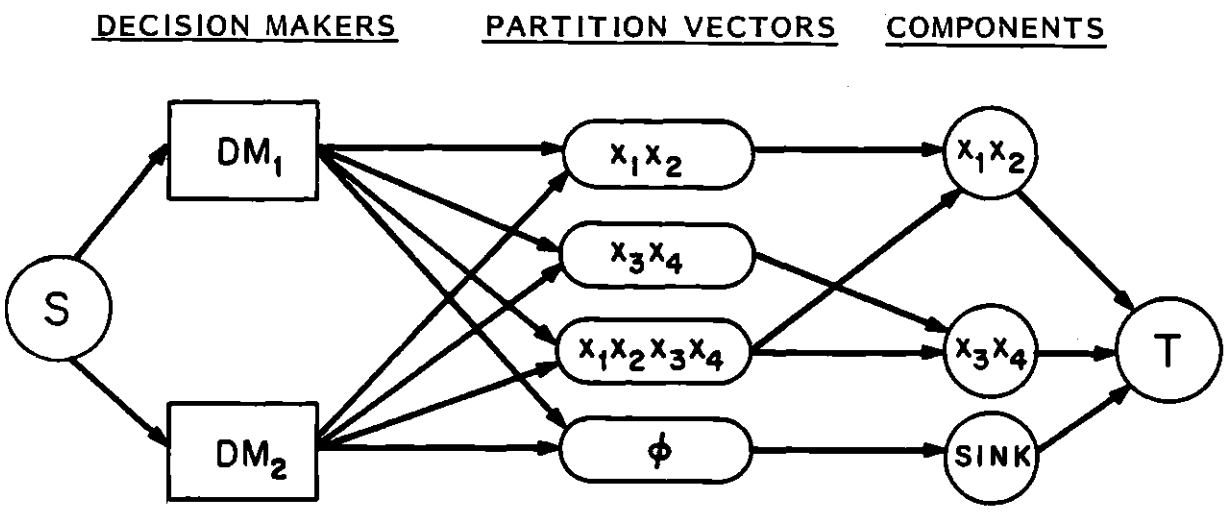


Figure 4.6. Prespecified Groups of Components

composed of components (x_1, x_2) and (x_3, x_4), respectively, and partition k, which is composed of $x_{g'}, x_{g''}$, is assigned to the m-th DM, then the corresponding partitioning matrix π^m_k is the 4 x N dimensional matrix:

$$\pi^m_k = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 & \cdots & x_N \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0 & \vdots & 0 \\ 0 & 0 & 0 & 1 & 0 & \vdots & 0 \end{bmatrix} \end{array}$$

rather than a 2 x G dimensional matrix. It is implicitly assumed, without loss of generality, that each component x_n, n = 1,2,...,N is of dimension one.

4.7 REDUNDANCY

The last special case addresses the issue of redundancy with respect to the processing of components. This case requires that the problem be viewed from a different perspective. The intent is to store processed data for use in future or other tasks. The concept of distributed database systems (DDSs) more clearly illustrates this idea.

TABLE 4.7 PRESPECIFIED GROUPING: DIMENSIONALITY COMPARISON

NUMBER OF VARIABLES

VARIABLES	WITHOUT PRESPECIFIED GROUPING (M = 2, N = G = 4)	WITH PRESPECIFIED GROUPING (M = 2, N = 4, G = 2)
B_{Sm}	2	2
Y_{mk}	32	8
D_{kn}	33	5
V_{nT}	5	5
Total:	72	20

NUMBER OF CONSTRAINTS

CONSTRAINTS	WITHOUT PRESPECIFIED GROUPING (M = 2, N = G = 4)	WITH PRESPECIFIED GROUPING (M = 2, N = 4, G = 2)
(4.9a)	2	2
(4.9b)	2	2
(4.9c)	16	4
(4.9d)	5	5
(4.9e)	4	4
(4.9f)	1	1
Total:	30	18

Distributed database systems evolved to satisfy several needs. One need was to have the data which required constant updating near the users. A second need was to reduce vulnerability and thus allow the uninterrupted flow of information, should one data center fail.

Redundancy, as interpreted with respect to S-E structures, can be integrated with the concept of DDSs. The concept of storing processed data requires a reexamination of what a S-E structure is. It was initially defined to be a group of DMs who (i) are not hierarchical with respect to each other and (ii) perform a complex task. Two types of tasks now concern the S-E. The first (original) task is to process the data of the input vector. The second task is to store the data, be it processed or not. This second task can be incorporated into the original problem by introducing constraints into the system which cause the original definition of efficiency to be violated. The original statement of objective required that a complex task be performed accurately using the minimum number of DMs while satisfying the constraints. Implicit in this objective was that each component of the input vector should be processed exactly once.

Each DM in the DDS processes and stores the entire partition of components he receives. All components are again assumed mutually independent. A subset of these processed components is also transmitted. This subset can range from the entire partition of processed components to the empty set. Time is required to process, store, and transmit components. The storage time and transmitting time functions are assumed identical and are imbedded in a single processing time function, τ_k^m.

The GN formultions corresponding to three different sets of conditions imposed on a DDS will be examined. The first case requires that each component n must be processed and stored R_n times. If $R_n = 1$ for all n, then this formulation defaults to the original GN formulation. However, if $R_n > 1$, for at least one component, then the original assumption that each partition can be assigned at most one time is no longer valid. This, in turn, prevents the problem of selecting and assigning partitions of components to DMs from being formulated as a GN

unless the size of the problem is increased significantly.

The reasoning is identical to that presented in Section 4.3 when partitions of representative components could be assigned to more than one DM. To formulate this problem as a GN, each partition k would have to be enumerated R times, where R equals the maximum number of times any component in the partition must be processed; partition k is repeated $(\max_{n \in \ell_k} R_n)$ times. Note that the minimum number of DMs, M^*, that could possibly perform the overall task is $(\max_{n \in \ell_k} R_n)$ since each DM processes any component at most one time. The maximum number of partitions that must be enumerated is $M^* \cdot 2^N$. The number of computations necessary to define $\mathbf{K}^m$ for each DM does not increase, however, since identical partitions are being enumerated.

The total number of variables, Y_{mk}, increases from $M\ 2^N$ to $M^2\ 2^N$ and the total number of variables, D_{kn}, increases from $(N\ 2^N+1)$ to $M(N\ 2^2+1)$. The problem cannot be decoupled into M separate GN problems because, as before, the decoupled problems would not guarantee every component n is processed R_n times. The information structure is constructed as was done previously. Two or more DMs may possess identical partitioning matrices, however.

Alternatively, the problem can be formulated as a mixed integer linear program. Minor modifications to the MILP presented in Table 3.1 are all that are required. The right hand side constants (equal to one) of constraint set (3.6c) are each replaced by R_n, the number of times each component is required to be stored. The dimensionality of this problem remains unaffected. The information structure is derived in an identical manner to that in Section 3.2.

The second case requires that each specified partition of components, k', be processed and stored $R^{k'}$ times. Within the set of specified partitions, $\mathbf{K}'$, the partitions are assumed disjoint, i.e., they contain no common components. This case is an extension of that presented in the previous section. Supercomponents are again created, composed of the

specified partitions of components that must be processed together. The difference is that now each of these groups of components must be processed more than one time. As in Section 4.6, $\mathbf{L}$ is redefined to be the set of supercomponents, $\mathbf{K}'$ is the set of partitions that can be constructed from these components and $\mathbf{K}^m$ is the set of partitions of components contained in $\mathbf{K}'$ that the m-th DM can process without overload.

Using these definitions, the problem of selecting and assigning partitions of components to DMs so that the minimum number of DMs is used to process each specified partition $R^{k'}$ times and all remaining components exactly once, is identical to that presented in the previous case of this section. The effects on the dimensionality of the problem resulting from the reduction in the number of partitions that must be considered explicitly is very significant, as was presented in Section 4.6. This reduction may be so significant as to warrant formulating the problem as a GN, using the procedure presented in the previous case since the algorithms for solving this class of problems are so efficient.

Using the GN formulation the information structure would be derived in an identical manner to that presented in Section 4.6. Using the MILP formulation, $Y_{mk} = 1$ would correspond to the k-th partition being assigned to the m-th DM. The corresponding partitioning matrices would be constructed in the identical fashion to those constructed using the GN formulation above.

In the third case, the specification of a group of DMs, $\mathbf{M}'$, to receive particular components to process and store, is considered. This may be necessary if certain strategic locations require having certain data always available. This case is an extension of the specialization case presented in Section 4.5. The simplifying assumptions of Section 4.5 that (a) DMs within a group possess identical processing time functions and (b) can only process those components specified, are maintained. The entropy for each group of components and the processing time required to process these specified components are computed. If overload does not occur, then these components are considered processed. If any components have not been specified, then a GN is formulated so as to assign these

components to DMs who are unrestricted as to the type of components they may process. The formulation is identical to that presented in Table 3.2 and the corresponding partitioning matrices are constructed in an identical fashion. The overall information structure would include these partitioning matrices as well as those associated with each DM in each specialized group.

4.8 SUMMARY

The primary purpose of this and the previous chapter was to provide an approach to model building, to the analysis of the partition selection and assignment process, and to the construction of information structures for parallel processing. The dimensionality issue evolving from the explicit enumeration of all possible partitions caused major concerns. Consequently, special cases were explored which revealed how the dimensionality of the problem may be made manageable under reasonably realistic assumptions. In the next chapter the assumption of mutual independence of the components of the input vector is exploited. Formulations are presented which consider all possible partitions implicitly. The special cases and other issues addressed previously are again considered from this alternative point of view.

CHAPTER 5

PARALLEL PROCESSING: IMPLICIT ENUMERATION

5.1 INTRODUCTION

The previous two chapters have explored procedures for selecting and assigning partitions of components to DMs so that all components are assigned and processed and no decision maker is overloaded. The explicit enumeration of all distinct partitions of components resulted in the problem having very high dimensionality. Fortunately, the mutual independence of the components of the input vector allows an alternative formulation to be developed which reduces significantly the size of the problem. This is a consequence of not having the distinct partitions enumerated explicitly. Rather, only the components and the DMs are considered explicity.

Since the partitions are not enumerated explicitly, the sets $\mathbf{K}$ and $\mathbf{K}^m$ do not have to be defined. The elimination of the set $\mathbf{K}^m$, for each m, makes checking the overload condition for each partition-DM pair unnecessary. Thus $M \cdot 2^N$ computations are eliminated.

The alternative implicit formulation still retains a generalized network structure. Therefore, the efficiency of the algorithms devised to solve this class of problems and the significant reduction in the dimensionality of the problem make this formulation very attractive.

5.2 NETWORK FORMULATION

In the implicit formulation, individual components, instead of partitions, are selected and assigned to each DM in the model. The group of components that are assigned to the m-th DM define the partition vector. In this way, the required M partition vectors are constructed implicitly. The conditions for selecting and assigning components are

a) every component is processed, and

b) no DM is overloaded.

Let Y_{nm} be a binary variable which equals one if the n-th component is assigned to the m-th DM, zero otherwise. To guarantee that every component is processed once and only once, the following set of constraints is established:

$$\sum_{m=1}^{M} Y_{nm} = 1 \qquad n = 1,2,\ldots,N \qquad (5.1)$$

where

$$Y_{nm} = 0,1 \qquad n = 1,2,\ldots,N;\ m = 1,2,\ldots,M \qquad (5.2)$$

A network structure which links every component to every DM is shown in Figure 5.1.

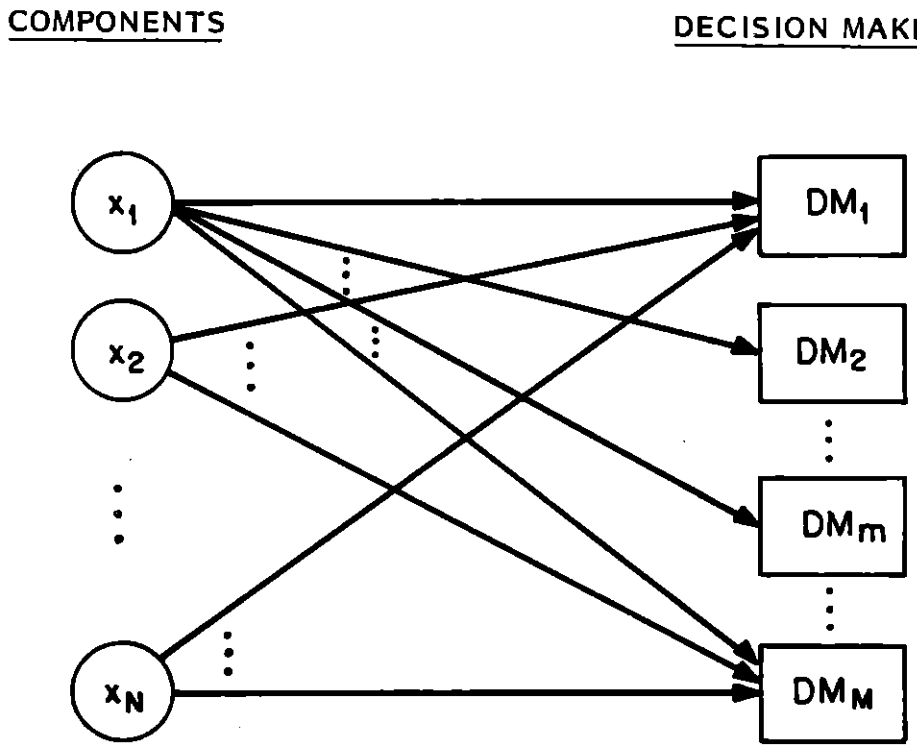


Figure 5.1. GN Formulation: Implicit Enumeration

To guarantee a DM is not overloaded, the time required to process the components assigned to him must be less than or equal to δ. The processing time function is assumed to have parameters $t^m(s) = st^m$ and c^m where s corresponds to the number of components assigned to the m-th DM. Since the components are mutually independent, the entropy of the s components assigned to a DM equals the sum of the entropies of those components. For this mean processing time function the time required to process the group of assigned components is the sum of the times required by the m-th DM to process each of the components separately. The components assigned to the m-th DM are obviously not known a priori. Using the variable Y_{nm} as an indicator variable the time required for the m-th DM to process component x_n is only included in his overall processing time if the component x_n is assigned to him. It follows directly that:

$$(t^m + c^m H_n)\, Y_{nm} = \begin{cases} (t^m + c^m H_n) & \text{if } Y_{nm} = 1 \\ 0 & \text{if } Y_{nm} = 0 \end{cases}$$

Furthermore:

$$\bar{\tau}^m = \sum_{n=1}^{N} (t^m + c^m H_n)\, Y_{nm}.$$

where $\bar{\tau}^m$ must be less than or equal to δ to guarantee the m-th DM is not overloaded. The following set of constraints must also be included in the formulation:

$$\sum_{n=1}^{N} (t^m + c^m H_n)\, Y_{nm} \leq \delta \qquad m = 1, 2, \ldots, M \qquad (5.3)$$

The coefficients of the variables, Y_{nm}, in constraint set (5.3) are clearly not restricted to unity or zero, implying that the formulation is not a pure network. It is a generalized network, however, since each variable appears in at most two constraint equations. In Table 5.1 the sets of constraints comprising this GN formulation are presented while in Table 5.2 the incidence matrix is given. The GN formulation itself is

TABLE 5.1 GN FORMULATION: IMPLICIT ENUMERATION

$$-\sum_{m=1}^{M} Y_{nm} = -1 \qquad n = 1,2,\ldots,N \qquad (5.4a)$$

$$\sum_{n=1}^{N} (t^m + c^m H_n)\, Y_{nm} \leq \delta \qquad m = 1,2,\ldots,M \qquad (5.4b)$$

$$Y_{nm} = 0,1 \qquad n = 1,2,\ldots,N;\ m = 1,2,\ldots,M \qquad (5.4c)$$

shown in Figure 5.1

The information structure can be constructed from the optimal solution to this problem.

Partitioning matrix, π^m_k corresponding to the set of components x_n for which $Y_{nm} = 1$, for each m, are inferred and then combined to define the information structure:

$$\Pi = (\pi^1_{k_1}, \ldots \pi^m_{k_m}, \ldots, \pi^M_{k_M})$$

In the following sections, the special cases presented in Chapter 4 are reexamined using the alternative formulation of the GN model to establish that an even larger reduction in the dimensionality of the problem is achieved.

5.3 IDENTICAL DECISION MAKERS

5.3.1 Single Group

As in Section 4.2.1, it is assumed that all components are one-dimensional and mutually independent and all DMs possess identical

TABLE 5.2 MATRIX STRUCTURE

Let $\underline{A}\underline{x} = \underline{b}$, where A is the matrix

$$\begin{bmatrix}
1 & 1 & \cdots & 1 & & & & & & & & & & & & & & & & \\
 & & & & 1 & 1 & \cdots & 1 & & & & & & & & & & & & \\
 & & & & & & & & \ddots & & & & & & & & & & & \\
 & & & & & & & & & 1 & 1 & \cdots & 1 & & & & & & & \\
 & & & & & & & & & & & & & \ddots & & & & & & \\
 & & & & & & & & & & & & & & 1_1 & 1 & \cdots & 1 & & \\
h_1^1 & & & & h_2^1 & & & & h_n^1 & & & & h_N^1 & & & & & & & \\
 & h_1^2 & \cdots & & & h_2^2 & \cdots & & & h_n^2 & \cdots & & & h_N^2 & \cdots & & & & \\
 & & \ddots & & & & \ddots & & & & \ddots & & & & \ddots & & & & & \\
 & & & h_1^M & & \cdots & & h_2^M & & \cdots & & & h_n^M & & \cdots & & & h_N^M & &
\end{bmatrix}$$

$\underline{x}$ is the vector

$$[Y_{11}, Y_{12}, \ldots, Y_{1M}\ Y_{21}, Y_{22}, \ldots, Y_{2M} \cdots Y_{n1}, Y_{n2}, \ldots, Y_{nM} \cdots Y_{N1}, Y_{N2}, \ldots, Y_{NM}]$$

$Y_{nm} = 0,1$ for all n, m: $\underline{b} = [\underbrace{1\ \ 1\ \ \ldots\ \ 1}_{N} \mid \underbrace{\delta\ \ \delta\ \ \ldots\ \delta}_{M}]$

where $h_n^m = t^m + c^m H_n$

properties. These simplifying assumptions do not lead to a reduction in the dimensionality of the GN. however. Each DM must be considered separately so that the particular assignments of components to DMs are obtained. This knowledge would be lost if a single representative DM was used instead.

There is an alternative approach, however, to solving the problem of selecting and assigning components to all DMS simultaneously, which takes advantage of these simplifying assumptions. This approach consists of

sequentially solving M^* MP problems belonging to a class known as knapsack problems. A knapsack problem (KP) is composed of exactly one constraint and variables which are all binary. The KP formulation for the m-th problem is:

$$\overline{\tau}^m_n \, Y_{mn} \leq \delta \qquad m = 1,2,\ldots,M^* : n \in L^m$$

where L^m is the set of components available for assignment; i.e., not assigned in one of the previous (m-1) KPs. The set L^m has dimension s_m where

$$s_m = N - \sum_{m'=1}^{m-1} s_{m'} \qquad m = 1,2,\ldots,M^*$$

The first KP assigns as many of the N components as possible (s_1) to a DM. The second KP assigns as many of the remaining ($N-s_1$) components to another DM. The number of components available for assignment in the m-th position is s_m. This process is repeated until all N components are assigned. Very efficient algorithms using Branch and Bound techniques exist to solve these KPs. [3] This leads to the possibility that the M^* problems, each with ($1+s_m$) variables and one constraint, may require cumulatively less total computer time than solving one GN problem with M N variables and (M+N) constraints.

The construction of the associated information structure proceeds in an identical fashion to that presented in Section 5.2. Each partitioning matrix specifies the components assigned to and processed by the m-th DM. Since the DMs are identical, the particular assignment of partitioning matrices to DMs is arbitrary.

5.3.2 Many Groups

The assumption that DMs in the g-th group, g = 1,2,...,G possess identical properties ($G \geq 2$), does not lead to a reduction in the dimensionality of the GN. The alternative approach of solving M^* KPs cannot be used either, unless the relative efficiency of DMs among groups

can be established and ranked, i.e. a DM from M^g can process any set of components more quickly than any DM from $M^{g'}$. If this dominance with respect to processing time functions exists, then the sequential approach used in the previous section could again be used. In particular, if g contains the most efficient DMs and d^g is its dimension, then the first d^g KPs would assign components to DMs from this g-th group. DMs from the second most efficient group would be considered next and this procedure would continue until all components were assigned.

5.4 IDENTICAL COMPONENTS

All components are assumed to be mutually independent and of dimension one. In Section 4.3.1 the case of all components having identical distributions (single group) was analyzed. It was shown that it is possible to obtain feasible solutions to the problem of assigning components directly to DMs (rather than paritions of components) and minimizing the total number of DMs in the S-E. Since this was essentially an implicit approach for deriving the solutions and constructing the information structure, it will not be repeated here. Instead, attention will be focused on the case of many groups of components. Let there be G groups of components with components within a group, M^g, $g = 1,2,\ldots,G$, possessing alphabets with identical probability distributions. All components are assumed mutually independent. As was done in Section 4.3.2, a representative component x^g can be selected for each group, L^g.

All possible partitions can be implicitly considered by using only the representative components and the dimension of each group. Rather than considering N components explicitly, G representative components need be considered with each one being assigned s^g times. It is possible that the same representative component, may be assigned to the same DM several times. In order to allow for this possibility, several of the constraints must be modified in the general formulation of the problem. In particular, since a representative component x^g can be assigned more than one time, Y_{gm} is no longer restricted to be binary. Rather it is restricted to be an integer with an upper bound of s^g. The modified constraint set of (4.4 c) is:

$$Y_{gm} = 0, 1, \ldots, s^g \qquad g = 1, 2, \ldots, G; \; m = 1, 2, \ldots, M. \quad (5.5)$$

To guarantee that every representative component is processed exactly s^g times constraint set (5.4a) is modified to

$$\sum_{m=1}^{M} Y_{gm} = s^g \qquad g = 1, 2, \ldots, G. \quad (5.6)$$

The modified GN formulation is presented in Table 5.3.

TABLE 5.3 GN FORMULATION: GROUPS OF IDENTICAL COMPONENTS

$$\sum_{m=1}^{M} Y_{nm} = s^g \qquad g = 1, 2, \ldots, G \quad (5.5a)$$

$$\sum_{g=1}^{G} (t^m + c^m H_g) Y_{nm} \leq \delta \qquad m = 1, 2, \ldots, M \quad (5.5b)$$

$$Y_{gm} = 0, 1, 2, \ldots, s^g \qquad g = 1, 2, \ldots, G; \; m = 1, 2, \ldots, M \quad (5.5c)$$

Figure 5.2 illustrates the reduction of the number of components and, consequently, of the overall dimensionality of the problem. As G, the number of distinct groups, decreases, the dimensionality of the problem is also reduced, although not as significantly as in the corresponding case of Section 4.3.2. This is due to the fact that the size of the current problem is only a linear, as opposed to exponential, function of the number of components.

The information structure can be obtained in a manner similar to that presented in Section 5.2. Each variable Y_{gm} which has a nonzero value R^g

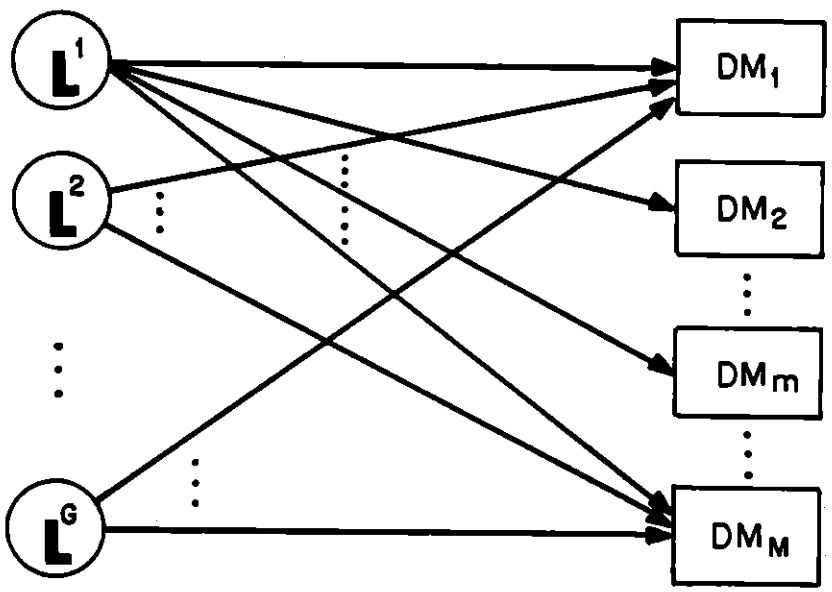


Figure 5.2. Groups of Identical Components

implies that R^g components from L^g, $g = 1,2,\ldots,G$, are assigned to the m-th DM. The assignment of the particular components from L^g to the appropriate DMs is arbitrary. For example, if $M^* = 2$, $L^1 = (x^1,x^2,x^3)$ and $L^2 = (x^4,x^5,x^6,x^7)$, $Y_{11} = 2$, $Y_{21} = 2$, $Y_{12} = 1$, and $Y_{22} = 2$, then the partition vectors $\underline{z}_k^m$, containing the representative components, which are processed by DM_1 and DM_2, respectively, are

$$\underline{z}_1^1 = (x^1,x^1,x^2,x^2), \text{ and } \underline{z}_2^2 = (x^1,x^2,x^2).$$

One possible assignment of the particular components is:

$$\pi_{k_1}^1 = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array} \qquad \pi_{k_2}^2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

so that

$$\Pi = (\pi^1_{k_1}, \pi^2_{k_2}).$$

Another possible solution is:

$$\pi^1_{k_3} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \pi^2_{k_4} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

so that

$$\Pi = (\pi^1_{k_3}, \pi^2_{k_4}).$$

5.5 EXAMPLES

In Section 4.4, examples were presented to illustrate the effects on the dimensionality of the GN problem resulting from various combinations of simplifying assumptions. These examples are explored again for the implicit enumeration formulation.

In the first two examples, all components were assumed identical; in Example 4.1 all DMs were also identical while in Example 4.2 identical DMs were placed in groups.

In both these examples the problem of selection and assignment of components to DMs did not require any mathematical programming algorithm to solve it.

Partitions of components were not considered explicitly since feasible solutions could be obtained with a minimal number of computations. These methodologies will not be reiterated.

Example 5.1

In this example it is assumed that

a) groups of components can be formed, with identical components within each

group possessing alphabets with identical probability distributions.

b) all components are mutually independent and of dimension one, and

c) all DMs are identical.

Let the number of components be eight (N = 8), the number of groups of components be three (G = 3), with $s^1 = 3$, $s^2 = 3$, and $s^3 = 2$, and the number of available DMs be three (M = 3). Since representative components can be defined for each group, only three components, rather than eight, need to be considered explicitly.

As shown in Section 5.3.1, the assumption of identical DMs does not cause a reduction in the number of DMs that need to considered explicitly.
It only allows the particular assignments of the specified components to be arbitrary.

The problem is still formulated as a GN. If the specified assignments of components are

$$Y_{11} = 2,\ Y_{21} = 1,\ Y_{22} = 2,\ Y_{32} = 1,\ Y_{13} = 1,\ Y_{33} = 1$$

then the partition vectors $\underline{z}_k^m$ are:

$$\underline{z}_1^1 = (x^1, x^1, x^2),\quad \underline{z}_2^2 = (x^2, x^2, x^3),\quad \underline{z}_3^3 = (x^1, x^3).$$

Since the DMs are identical, an alternative, equally feasible, solution is :

$$\underline{z}_2^1 = (x^2, x^2, x^3),\quad \underline{z}_3^2 = (x^1, x^3),\quad \underline{z}_1^3 = (x^1, x^1, x^2).$$

The information structure is constructed in an identical fashion to that presented in Section 5.4.

Example 5.2

Assumption (c) of the previous example is relaxed so that groups of DMs can be formed, DMs within a group possessing identical processing time functions.

The assumptions that a) all components are mutually independent and of dimension one and b) groups of components can be formed, components within each group possessing alphabets with identical probability distributions, are maintained.

As in the previous example, the assumption of groups of identical components allows the number of components that must be considered explicitly to be reduced from N to G. The assumption of groups of identical DMs has no effect on the dimensionality of the problem. Again, within a group of identical DMs, the specified assignments of components

$$\underline{z}^{m}_{k_m}, \quad m \in \mathbf{M}^g$$

is arbitrary; i.e.,

$$\underline{z}^{m'}_{k_{m'}} \text{ and } \underline{z}^{m}_{k_m} \qquad m, m' \in \mathbf{M}^g$$

can each process the other's set of components. Thus $\underline{z}^{m'}_{k_m}$ and $\underline{z}^{m}_{k_{m'}}$ is another feasible solution. Other than this arbitrariness, the information structure is constructed in an identical fashion to that presented in Section 5.4.

5.6 SPECIALIZATION AMONG DECISION MAKERS

As discussed in Section 4.5. specialization restricts the types of components that can be assigned to a particular DM. To imbed this constraint into the GN formulation developed in Section 5.2, it is only necessary to restrict the variables Y_{nm} included in the formulation. If n = 5 and DM_1 can only be considered for processing components x_1, x_2 and x_3 , then variables Y_{11}, Y_{21}, and Y_{31} would be included in the

formulation while Y_{41}, and Y_{51} would not be.

For each DM let $\mathbf{K}^m$ be the set of components that may be assigned to him.

The modified formulation is presented in Table 5.4.

TABLE 5.4 SPECIALIZATION OF DECISION MAKERS

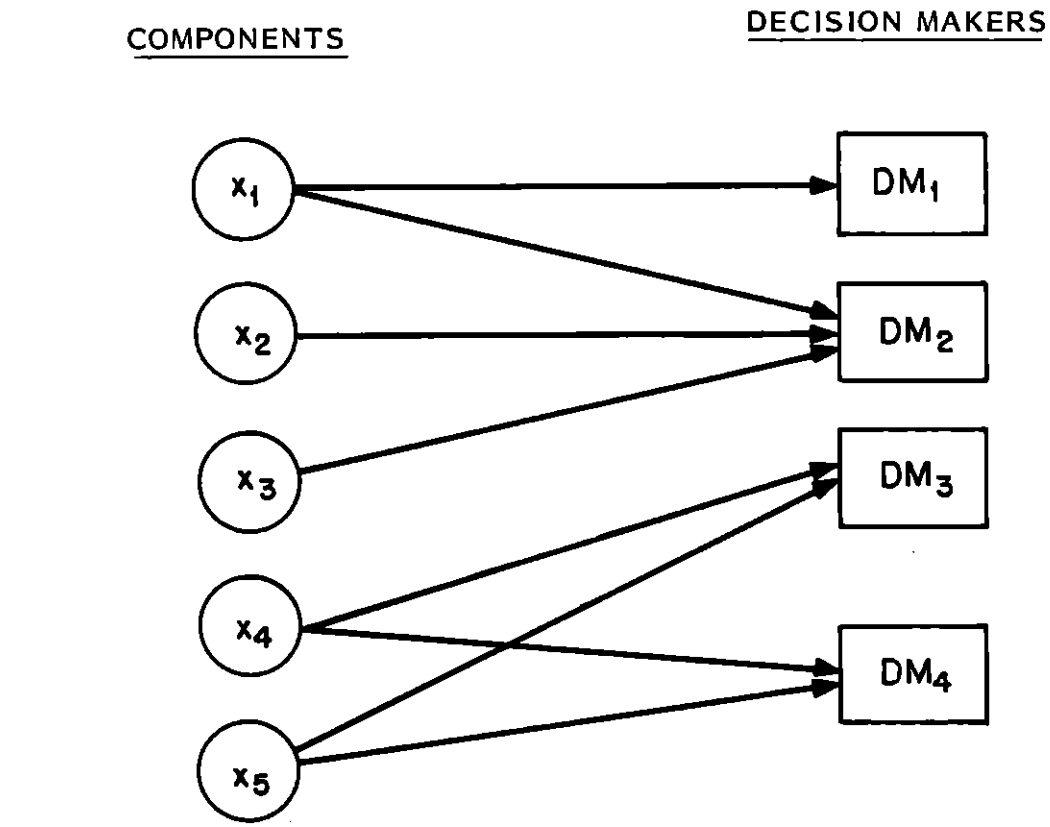
$$-\sum_{\substack{m=1 \\ n \in \mathbf{L}(m)}}^{M} Y_{nm} = -1 \qquad n = 1,2,\ldots,N \qquad (5.6a)$$

$$\sum_{n \in \mathbf{L}(m)} (t^m + c^m H_n)\, Y_{nm} \leq \delta \qquad m = 1,2,\ldots,M \qquad (5.6b)$$

$$Y_{nm} = 0,1 \qquad m = 1,2,\ldots,M;\ n \in \mathbf{L}(m) \qquad (5.6c)$$

The reduction in dimensionality is a function of the size of the sets $\mathbf{K}^m$ and the potential decoupling if sets of components can only be assigned to particular sets of DMs.

For example, let $\mathbf{M}^1 = (DM_1, DM_2)$, $\mathbf{M}^2 = (DM_3, DM_4)$, $\mathbf{L}^1 = \mathbf{L}^2 = (x_1, x_2, x_3)$ and $\mathbf{L}^3 = \mathbf{L}^4 = (x_4, x_5)$. Figure 5.3 illustrates the decoupling of this problem. The specialization restriction reduces the number of variables Y_{nm} that must be considered from 20 to 10. The decoupling effect allows two GNs to be solved, one with 6 variables, the other with 4 variables, rather than one GN with 10 variables. Since the complexity associated with the Branch and Bound algorithm used to solve this problem increases nonlinearly at a rate that is greater than unity, the ability to decouple the problem is a significant benefit.



(a) Original Problem

(b) Decoupled Problem

Figure 5.3 Specialization

5.7 PRESPECIFIED GROUPING OF COMPONENTS

As discussed in Section 4.6, although components may be mutually independent, reasons may exist which require certain components to be processed together. To guarantee these components were all assigned to the same DM, supercomponents were created. The set $\mathbf{L}(g)$ contained the set of components comprising this supercomponent, $H(g)$ was the set's entropy and $s(g)$ was the set's dimension. The set $\mathbf{L}$ of components comprising the input vector was redefined to be the set of supercomponents $\mathbf{L}(g)$ which has dimension G.

The number of components that need be considered explicitly is G. The only modification to be made to the GN concerns constraint set (5.4b) which is modified to account for the dimension of the supercomponents:

$$\sum_{g=1}^{G} (s(g)t^m + c^m H_g) Y_{gm} \leq \delta \qquad m = 1,2,\ldots,M \tag{5.7}$$

Table 5.5 presents the complete formulation.

TABLE 5.5 PRESPECIFIED GROUPING

$$-\sum_{m=1}^{M} Y_{gm} = -1 \qquad g = 1,2,\ldots,G \tag{5.8a}$$

$$\sum_{g=1}^{G} (s(g)t^m + c^m H_g) Y_{gm} \leq \delta \qquad m = 1,2,\ldots,M \tag{5.8b}$$

$$Y_{gm} = 0,1 \qquad g = 1,2,\ldots,G;\ m = 1,2,\ldots,M$$

The solution to the GN formulation is used to construct the information structure. As shown in Section 4.6, the actual components, not the supercomponents, must be included in the information structure. For example, if $Y_{11} = 1$, $Y_{21} = 1$, $Y_{32} = 1$, and $L^1 = (x_1, x_2, x_3)$, $L^2 = (x_4)$, and N = 4 then

$$\pi_1^1 = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \pi_2^2 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$$

so that

$$\Pi = (\pi_1^1, \pi_2^2).$$

5.8 REDUNDANCY

As discussed in Section 4.7, redundancy requires that one or more components be processed simultaneously by several DMs. Three different cases were explored in the context of Distributed Database Systems. The effect of redundancy on the ability to formulate the problem as a GN as well as the dimensionality issue was significant. These cases are reexamined now for the implicit enumeration approach.

The first case required that each component x_n be processed and stored R_n times. If R_n is unity for all n, then this formulation defaults to the original GN formulation presented in Section 5.2. If $R_n > 1$ for at least one component, then only a minor modification is necessary. Constraint set (5.4a) is modified so that

$$-\sum_m Y_{nm} = -R_n \qquad n = 1, 2, \ldots, N \qquad (5.9)$$

The information structure is constructed in an identical fashion to that presented in Section 5.2. The only difference is that partitioning matrices and their associated vectors will not necessarily contain distinct components.

The second case requires that R^g DMs must each process a specified set of components. In Section 5.7, the prespecification of groups of components was discussed. Supercomponents, composed of the specified sets of components, are again constructed. The formulation is identical to that presented in Table 5.5 except that the right hand side constants of constraint set (5.8a) are now replaced with R^g. The modified constraint set is

$$-\sum_{m=1}^{M} Y_{gm} = -R^g \qquad g = 1,2,\ldots,G \qquad (5.10)$$

The information structure is constructed in an identical fashion to that presented in Section 5.7. Again, the associated partitioning matrices and vectors will not necessarily have distinct components.

In the third case, a particular group of components must be processed and stored by each member of a prespecified group of DMs, **M'**. It is assumed that these components do not overload these DMs. If, however, the time required by the m-th DM to process his assigned components is strictly less than the available time δ, additional components may also be assigned to him provided they do not overload him. An available processing time, δ_m, is defined for each DM in **M'**, which is equal to the difference between the original available processing time and the time required to process the specified components.

Two modifications of the original GN formulation are made:

a) All components whose assignments to DMs have been prespecified are omitted. To guarantee that each of the remaining components is processed once and only once the following constraint set is formulated,

$$-\sum_{m=1}^{M} Y_{nm} = -1 \qquad n \in \mathbf{L}'' \qquad (5.11)$$

where **L"** is the set of unspecified components.

b) To guarantee that no DM is overloaded the following constraint set is formulated; each δ is replaced with the appropriate δ_m.

$$\sum_{n \in L'} (t^m + c^m H_n) Y_{nm} \leq \delta_m \qquad m = 1,2,\ldots,M \tag{5.12}$$

The variables, Y_{nm}, are again restricted to be binary

$$Y_{nm} = 0,1 \qquad n \in L'; \; m = 1,2,\ldots,M. \tag{5.13}$$

The partitioning matrix associated with each DM of the single-echelon (S-E) includes all components he was preassigned and all components for which $Y_{nm} = 1$, $n \in L'$. For example, if N = 3 and if DM_1 were preassigned components (x_1, x_2, x_3) and also were able to process component x_5, so that $Y_{51} = 1$, then his associated partitioning matrix would be:

$$\pi_1^1 = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

The information structure is constructed directly.

5.9 SUMMARY

This chapter has presented an alternative approach for modeling the allocation of data among decision makers using the parallel processing mode. This approach does not require each possible partition to be considered explicitly. As a result, the size of the problem has been reduced drastically from order 2^N to order N. The problem can still be formulated as a generalized network and solved using one of the efficient GN algorithms.

Special cases were examined: an occasion only a few straightforward computations were necessary to obtain feasible solutions.

It may occur, however, that even though the input vector has been partitioned to finest grain, no decision maker is able to process some component without being overloaded. Other information reduction strategies must be considered. In Chapter 6, the second information reduction strategy namely, the creation of slack resources (time), is discussed.

CHAPTER 6

ALTERNATE PROCESSING

6.1 INTRODUCTION

Information structures based on alternate processing are appropriate when the input symbol cannot be partitioned and no decision maker (DM) can process the input symbols without being overloaded. Although all components of the input vector have been assumed to be mutually independent, it is required that all the components be processed together. In order to allow each input symbol to be processed by a single DM without causing him to be overloaded, the DM must be given more time to process the symbol. This may be accomplished by using either stochastic or deterministic strategies.

6.2 STOCHASTIC STRATEGIES

A stochastic strategy assigns a generated input symbol to a particular DM m with a certain probability, q_m. Figure 6.1 illustrates this. The probability that the m-th DM does not receive an input symbol during the next i intervals is given by $(1 - q_m)^i$. The expected interarrival time for the m-th DM is found as follows:

$$\sum_{i=1}^{\infty} i\delta (1 - q_m)^{i-1} q_m = \delta/q_m$$

The problem then becomes one of determining simultaneously

a) the minimum number of DMs necessary to process the input symbols without any one being overloaded and

b) the frequency, q_m, with which each of these DMs receives an input symbol.

To guarantee each DM is not overloaded the mean processing time for the m-th DM must be less than or equal to the available time, i.e.,

$$\bar{\tau}^m \leq \delta/q_m \qquad m = 1,2,\ldots,M \qquad (6.1)$$

It follows directly from (6.1) that the frequency with which a DM receives an input symbol must be less than the ratio of the input generation time δ and the mean processing time,

$$q_m \leq \delta/\bar{\tau}^m \qquad m = 1,2,\ldots,M \qquad (6.2)$$

Furthermore, the ratio,

$$\zeta_m = \delta/\bar{\tau}^m, \qquad m = 1,2,\ldots,M \qquad (6.3)$$

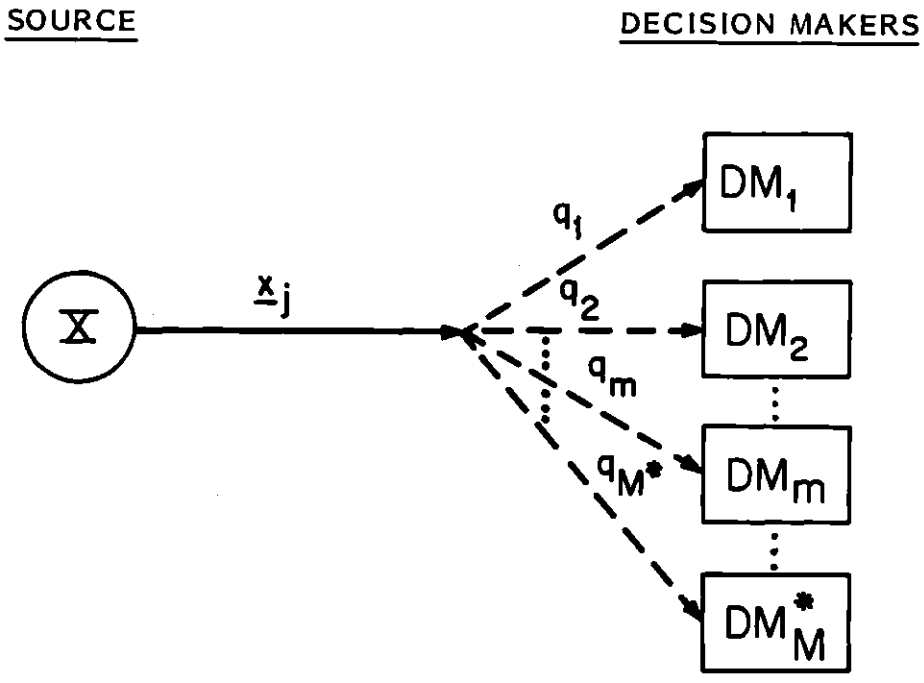
is strictly less than unity for all DMs, otherwise some DM would not be overloaded.

It is also necessary that the sum of the relative frequencies be unity, i.e.,

$$\sum_{m=1}^{M^*} q_m = 1 \qquad (6.4)$$

where M^* has not yet been determined.

For an efficient DM, $\bar{\tau}^m$ is small and therefore ζ_m is close to unity. The closer to unity this ratio is, the less extra time a DM needs to process an input symbol. Assume, as usual, that there are M DMs characterized by different values of $\bar{\tau}^m$ and that cost is not a factor. In order to use the minimum number of DMs, those DMs which are the most efficient are selected. The procedure for selecting them starts with setting the q_m's at their upper bound, ζ_m. The ζ_m are rank ordered from largest to smallest so that ζ_{mf} indicates that the m-th DM's ratio, ζ_m, is the f-th largest, f = 1,2,...,M. The ζ_{mf} are summed until the addition of the (f' + 1) ratio causes the sum to exceed unity, i.e.,



$\mathbf{X}$: set of alphabet symbols

$\underline{x}_j$: j-th element of $\mathbf{X}$

q_m: frequency of assignment of input symbol to the m-th DM

Figure 6.1. Alternate Processing: Stochastic Strategy

$$\sum_{f=1}^{f'+1} \zeta_{mf} > 1 \geq \sum_{f=1}^{f'} \zeta_{mf} \qquad (6.5)$$

If the right hand side of the (6.5) is an equality, then the minimum numbers of DMs, M^*, necessary to process the input symbols without overload is f', i.e.,

$$M^* = f'$$

and

$$q_m = \begin{cases} \zeta_{mf} & \text{if } 1 \leq f \leq f' \\ 0 & \text{otherwise} \end{cases} \qquad (6.6)$$

If the right hand side of (6.5) is a strict inequality, then the minimum number of DMs, M^*, is f'+1. However in this case it is incorrect to let $q_m = \zeta_{mf}$ if f is in the set [1, f'+1], since $\sum_{f=1}^{f'+1} \zeta_{mf} > 1$. In order to derive the appropriate values for the q_m so that the sum of the q_m's is unity, the values of one or more of the q_m's must be strictly less than their corresponding upper bounds ζ_{mf}. Several approaches exist.

Before discussing these approaches, however, an introduction of cyclical strategies is necessary. A cycle is defined to be the minimum length of time, δ', necessary for the assignment of input symbols to each DM in the S-E to be exactly in the proportion q_m, $m = 1,2,\ldots,M^*$. If every q_m is rational, i.e., the ratio of two integers, then the length of time δ' is an integer multiple of δ with the multiplier being the lowest common denominator of the q_m's. If any q_m is irrational, then the length of the cycle is infinite (acyclical). A stochastic cyclical strategy is a strategy in which, within each period, the proportion of the incoming symbols to the m-th DM is exactly q_m, $m = 1,2,\ldots,M$ but the ordering of the assignments to DMs is not specified and most likely will change from cycle to cycle. A stochastic acyclical strategy simply guarantees that in the "long run" the m-th DM will be assigned a proportion, q_m, of the total number of input symbols generated.

Depending on the nature of the organization, it may not matter whether the stochastic strategy is cyclical or acyclical. When $\sum_f \zeta_{mf}$ equals unity, the corresponding q_m's are uniquely defined. When $\sum_f \zeta_{mf}$ exceeds one, at least one DM's corresponding q_m is not set equal to its upper bound. Since the value of each q_m may be adjusted to any value less than or equal to its upper bound, it may be possible to have each q_m be rational so long as the sum of the q_m's is still one and each q_m does not exceed its upper bound.

If the ζ_{mf} are all rational, then one approach for determining rational values for the q_m, $m = 1,2,\ldots,M^*$, when the sum of the ζ_{mf} exceeds one, is to set each DM's q_m to an adjusted upper bound, ζ'_{mf}. This adjusted upper bound is the original upper bound reduced by an amount given by a rational number; the amount may be the same for all DMs.

Let δ^* denote the excess time derived from having the sum of the ratios just exceed unity, i.e.,

$$\delta^* = \sum_{f=1}^{f'+1} \zeta_{mf} - 1. \tag{6.7}$$

Divide δ^* by M^*, the number of DMs, and subtract this amount from each ζ_{mf}, $f = 1,2,\ldots,f'+1$, so that all q_m are defined as follows:

$$q_m = \begin{cases} \zeta_{mf} - \delta^*/M^* & f = 1,2,\ldots,f'+1 = M^* \\ 0 & \text{otherwise} \end{cases} \tag{6.8}$$

The information structure associated with either stochastic strategy is:

$$\Pi^m(t) = \begin{cases} I & \text{with probability } q_m, \quad m = 1,2,\ldots,M^* \\ 0 & \text{with probability } (1-q_m) \end{cases} \tag{6.9}$$

6.3 DETERMINISTIC STRATEGIES

A deterministic strategy is one in which the ordering of the assignment of input symbols to DMs within any period is specified. A specific sequence of assignments determines a set of relative frequencies just as in the stochastic case. However, these relative frequencies do not have a probabilistic interpretation. The computation of the values of the q_m's are done in an identical fashion to that presented in the previous section. The information structure differs however. The deterministic information structure is:

$$\Pi^m(t) = \begin{cases} I & \text{if } t = t^m_{jf} \quad m = 1,2,\ldots,M^* \quad j = 1,2,\ldots,q_m\delta' \\ & \qquad\qquad f = 1,2,\ldots,\delta' \\ 0 & \text{otherwise} \end{cases} \tag{6.10}$$

where t^m_{jf} is the j-th time the m-th DM is assigned an input symbol and it is the f-th assignment in the sequence of length δ'.

$$\sum_j t^m_{jf} / \delta' = q_m \qquad m = 1,2,\ldots,M^*.$$

A special case of the deterministic strategy is a periodic one characterized by the following properties:

a) the length of the sequence in exactly M^*, the number of decision makers, and

b) each DM is assigned exactly one symbol during the execution of the sequence.

The information structure for the periodic strategy is given by

$$\Pi^m(t) = \begin{cases} I & t = \theta M^* + m \\ 0 & t = \theta M^* + m \end{cases} \qquad m = 1,2,\ldots,M^*; \quad \theta = 0,1,2,\ldots \tag{6.11}$$

The relative frequency for a DM receiving a symbol for processing in the periodic case is

$$q_m = 1/M^* \text{ for all } m \tag{6.12}$$

In order that no DM be overloaded, the mean symbol processing time must satisfy the following inequality

$$\bar{\tau}^m \leq M^* \delta \qquad m = 1,2,\ldots,M^* \tag{6.13}$$

The relationship (6.13) can be used to derive the highest lower bound of the minimum number of DMs in the S-E that can process the input symbol without overload:

$$\max_m \bar{\tau}^m/\delta \leq M^* \tag{6.14}$$

6.4 SPECIAL CASES

In this section, special cases are again explored as in chapters 4 and 5 to provide insight into the effect various properties have on the dimensionality of the design problem.

6.4.1 Identical Decision Makers

In the first case, all DMs are assumed to possess identical properties:

a) the processing time functions of all the DMs are identical,

b) the DMs are able to process any input symbol, and

c) they have identical costs.

Assumption (a) implies that only one mean processing time, $\bar{\tau}$, need be computed. Since the DMs are identical it follows from (6.2) that

$$q_m = q \leq \delta/\bar{\tau} \qquad m = 1,2,\ldots,M. \tag{6.15}$$

It follows from equations (6.4) and (6.15) that

$$\sum_{m=1}^{M} q_m = M^* q = 1 \tag{6.16}$$

Therefore, M^* is the smallest integer that satisfies eq. (6.16) subject to the condition (6.15). The resulting strategy can be either stochastic or periodic. The solution to (6.15) and (6.16) only requires that the relative frequency of symbol processing by a DM be $1/M^*$.

The corresponding information structures are

Stochastic

$$\Pi^{m}(t) = \begin{cases} I & \text{with probability } q = 1/M^{*} \qquad m = 1,2,\ldots,M^{*} \\ 0 & \text{otherwise} \end{cases} \tag{6.17}$$

Periodic

$$\Pi^{m}(t) = \begin{cases} I & t = \theta M^{*} + m \qquad m = 1,2,\ldots,M^{*};\ \theta = 0,1,\ldots \\ 0 & \text{otherwise} \end{cases} \tag{6.18}$$

6.4.2 Groups of Identical Decision Makers

Let there be G groups of decision makers denoted by $\mathbf{M}^{g}$, $g = 1,2,\ldots,G$, with the DMs within each group possessing identical properties. Let d_g be the dimension of $\mathbf{M}^{g}$ and let ζ_g be computed for each group:

$$\zeta_g = \delta/\bar{\tau}^{g} \tag{6.19}$$

The ζ_g are then rank ordered, as was done in Section 6.2. Beginning with ζ_{g1}, the ratios are summed, by groups, until the sum exceeds unity, i.e.,

$$\sum_{f=1}^{f'+1} d_g \zeta_{gf} > 1 \geq \sum_{f=1}^{f'} d_g \zeta_{gf} \tag{6.20}$$

If the right hand side of (6.20) is an equality, then

$$q_m = \begin{cases} \zeta_{gf} & m \in \mathbf{M}^{g} \qquad g = 1,2,\ldots,f' \\ 0 & \text{otherwise} \end{cases} \tag{6.21}$$

If the right hand side of (6.20) is a strict inequality, then the excess time δ^{*} is defined by

$$\delta^* = 1 - \sum_{f=1}^{f'} d_g \zeta_{gf} \qquad (6.22)$$

Then a procedure identical to that described in Section 6.2 is used to determine the minimum number of DMs from group $\mathbf{M}^{f'+1}$ that is needed to make eq. (6.20) an equality. The resulting information structure based on the stochastic strategy is given by

$$\Pi^m(t) = \begin{cases} I \text{ with probability } q_m & m \in \mathbf{M}^g,\ g = 1,2,\ldots,f' \\ & m \in \tilde{\mathbf{M}}^{f'+1} \subset \mathbf{M}^{f'+1} \\ 0 \text{ otherwise} & \end{cases} \qquad (6.23)$$

where

$$q_m = \begin{cases} \zeta_{f'+1}^{g} & m \in \mathbf{M}^g \quad g = 1,2,\ldots,f' \\ & m \in \tilde{\mathbf{M}}^{f'+1} \\ 0 & \text{otherwise} \end{cases} \qquad (6.24)$$

Further refinements can be carried out to obtain cyclical strategies.

6.5 SPECIALIZATION

In Section 4.5 the concept of specialization was introduced for parallel processing. In that context, specialization implied that certain DMs were able to process only certain components of the input vector. This concept is inappropriate in the alternate processing case since DMs receive and process an entire input vector. However, specialization may be redefined to imply that a DM or group of DMs is able to process only certain elements from the alphabet set, X.

Let X^g, $g = 1,2,\ldots,G$ be the g-th subalphabet, i.e., the g-th group of elements of X. Let X^g have dimension γ^g, $g = 1,2,\ldots,G$. Let $\underset{\sim}{x}_j^g$ be the j-th element of X^g, $j = 1,2,\ldots,\gamma^g$, $g = 1,2,\ldots,G$. Let $\mathbf{M}^g$ correspond to the set of DMs of dimension d_g, $g = 1,2,\ldots,G$, available to process elements from X^g. As was shown in Section 4.5, a decoupling of the problem occurs. Although no mathematical programming formulations are

developed to solve the problems in this chapter, the decoupling effect is still relevant. G stochastic strategies must be constructed, rather than one, but each strategy involves fewer DMs. For each decoupled problem, the procedure for determining the optimal strategy is done in exactly the same fashion as was done in Section 6.2. The only modifications that must be done concern the probabilities associated with each of the elements in each subalphabet. These probabilities must be recomputed to include the fact that the probability that a DM from $\mathbf{M}^g$, g = 1,2,...,G receives the next generatd input symbol is equal to the probability that an input symbol from X^g is generated. This probability is equal to the sum of the probabilities of the elements in X^g, g = 1,2,...,G, i.e.,

$$p(X^g) = p^g = \sum_{j=1}^{\gamma^g} p(\underset{\sim}{x}_j^g).$$

Given that the next element generated is from X^g, the probability that it is $\underset{\sim}{x}_j^g$ is the conditional probability:

$$p(\underset{\sim}{x}_j^g | X^g) = p_j^g = p(\underset{\sim}{x}_j^g)/p^g, \quad g = 1,2,...,G; \quad \underset{\sim}{x}_j^g \in X^g.$$

Since these individual probabilities p_j^g, have increased, the entropy associated with any subalphabet is smaller. Figure 6.2 illustrates this.

Figure 6.2a illustrates the original problem, highlighting the assumptions particular to this special case. The source, which generates inputs from an alphabet set X, is now detailed to illustrate the G subalphabets considered. Input symbols are generated from the source at an average rate, δ^{-1}. If a symbol is generated from the subalphabet, X^g, it is assigned to the g-th set of DMs. Although M^* DMs compose the single-echelon they are explicitly grouped according to the set of input symbols they process.

Figure 6.2b presents the decoupled problems resulting from the one-to-one correspondence between subalphabet groups and decision maker groups. The probability that an input symbol is generated from the g-th group is p^g. Thus input symbols are generated from this group at a rate

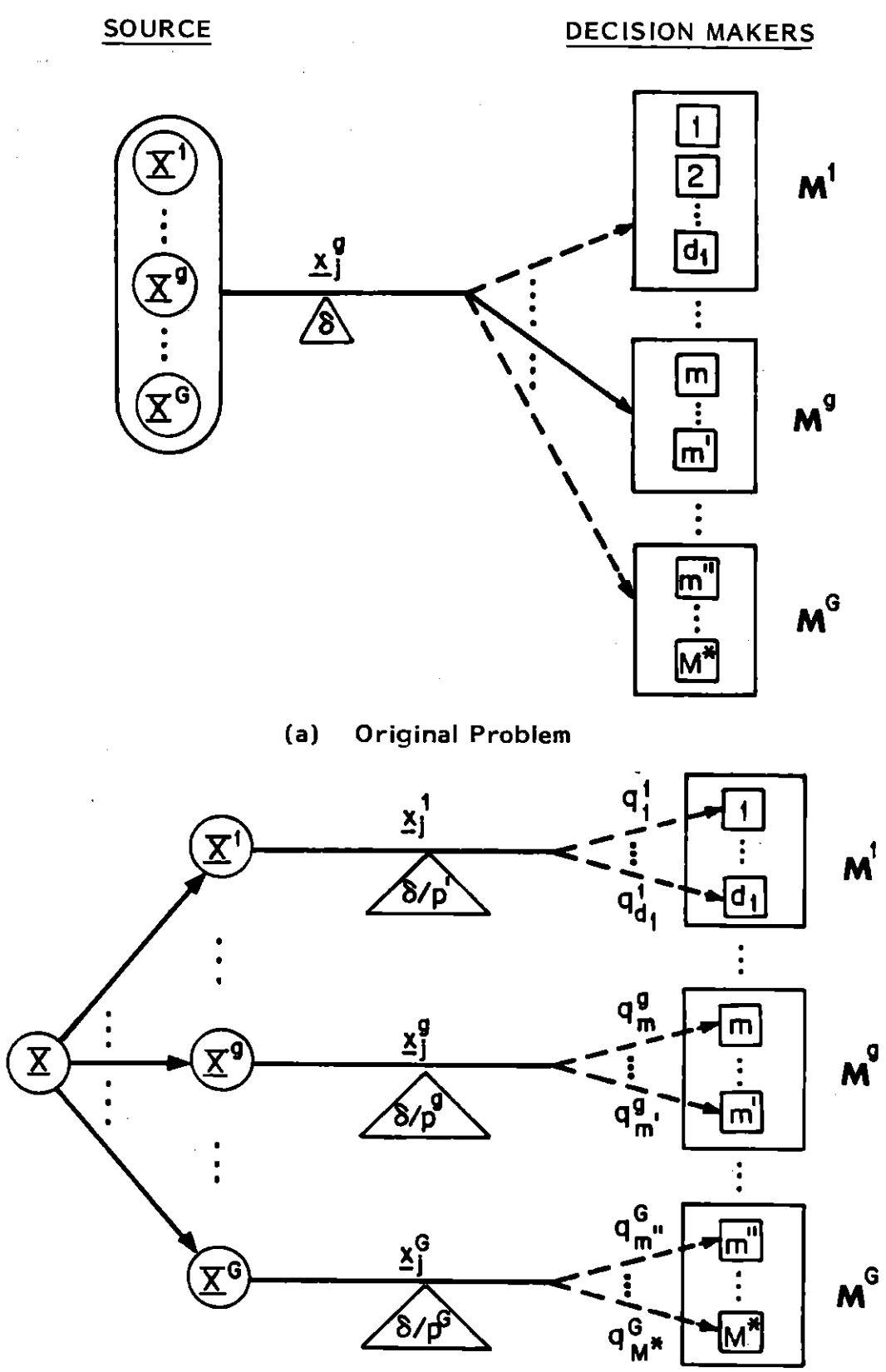


Figure 6.2. Specialization

δ/p^g. The assignment of the input symbol to the m-th DM in the g-th group of decision makers occurs with frequency q_m^g.

The information structure associated with each decoupled problem is constructed in an identical fashion to that presented in Section 6.2. In particular, the information structure for the m-th DM in the g-th group is:

$$\Pi_g^m(t) = \begin{cases} I & \text{with probability } q_m^g \\ 0 & \text{with probability } (1 - q_m^g) \end{cases} \qquad \begin{matrix} m = 1,2,\ldots,M^*; \\ g = 1,2,\ldots,G \end{matrix}$$

The alternate processing mode has been introduced as a means of implementing the second information reduction strategy, i.e., creation of a slack resource (time) to reduce information overload, as discussed in Chapter 1. Methodologies and the resulting information structures have been presented for increasing the available time a DM has to process each of his assigned input symbols.

CHAPTER 7

INFORMATION STRUCTURES FOR SINGLE-ECHELONS

7.1 INTRODUCTION

In the last four chapters a detailed analysis of pure parallel and alternate information structures for the single-echelon was presented. These two basic structures were explored under a variety of assumptions regarding the properties of the input symbols and of the decision makers. The emphasis in the development was on procedures for allocating the input symbols and for determining the minimum number of decision makers, given that a specific structure was desired. In the general case, however, the organization designer is given only the properties of the symbol source and a limited number of decision makers who could form the single-echelon. What he needs is a methodology for approaching the design of the information structure. Such a methodology, based on the results obtained so far and consisting of five steps, is presented in this section. It is then applied to three problems that require integration of both parallel and alternate processing modes.

STEP 1: Decomposition of Input

The first step in the procedure is the analysis of the properties of the inputs to the organization. As discussed in Chapter 2, the organization may be receiving inputs from a wide variety of sources; the sources may be physically similar, generating signals of different values (e.g., a set of sensors or radars) or dissimilar (e.g., some sensors, some sonars). Each one of the sources that constitute the supersource may be generating scalar signals or vector ones with components that would be related (e.g., position and velocity coordinates of an aircraft).

The basic model of the supersource requires that its components be mutually independent. A procedure for accomplishing this has already been described; it is based on replacing each value of a vector by a scalar

(or mapping) and determining the probabilities associated with the alphabet of the scalar. The components of the supervector are then analyzed to determine whether they can be arranged in groups, where each group consists of elements whose alphabets have identical probability distributions.

An alternative classification of the symbols generated by the supersource is in terms of subalphabets (see Section 6.5). Whether such a property is of use depends on the characteristics of the decision makers.

In summary, the first step consists of the following:

a) construction of the single supersource;

b) restructuring of the symbol vector so that its components are mutually independent;

c) identification of groups of components with alphabets that have identical probability distributions;

d) identifications of components that must be processed together, i.e., that cannot be partitioned further; and

e) identification of subalphabets (DM dependent).

STEP 2: Information Reduction Strategy

The two basic strategies are

a) creation of self-contained subtasks, and

b) creation of slack resources.

The first strategy is applicable when the input vector can be partitioned into subvectors. The second strategy is feasible when the organization can tolerate some delay (beyond the mean interarrival time δ) in the processing of the input vector. It is here that the organization designer's understanding of the task to be performed by the organization is crucial. He has to determine the extent to which parallel processing can be used and estimate the maximum tolerable delay which determines the extent to which alternate processing can be used. Assuming that the overall task can be accomplished by various combinations of alternate and parallel

processing, he then has to assess the trade-offs between subdivisions into smaller independent tasks and longer delays.

Once he selects an integrated information reduction strategy, the designer proceeds to Steps 3 to 5. At the end, when he evaluates the resulting design, he can return back to Step 2 and modify the strategy.

STEP 3: Mathematical Model

The next step consists of the formulation of the mathematical model that represents the integrated information reduction strategy selected in Step 2. There are four basic constraints, common to all strategies considered so far, that must be expressed analytically.

a) All components must be processed. It is a key assumption in the design procedure that all the data generated by the sources that have been included in the supersource must be processed. If any data were to be rejected, then their source has been eliminated from the supersource model.

b) No decision maker is overloaded. This condition requires that in the final design the mean processing time of each decision maker in the echelon is smaller than or equal to the mean interarrival time for symbols (or tasks) received by him.

c) Only decision makers who receive data from the supersource are members of the single-echelon. This constraint is introduced in the context of the optimization problem (Step 4) in which the minimum number of decision makers is sought for processing the incoming data. These decision makers are selected from a larger group of available DMs. The decision makers who are not used in the single-echelon are assigned the null or empty subtask.

d) Each DM is assigned at most one subtask. Additional constraints that are specific to the selected strategy can be introduced.

STEP 4: Optimization Problem

Given the mathematical formulation of the single-echelon model an objective function must be defined. A minimum cost, one in which each DM is assigned a cost that is related to his capabilities and limitations reduces the minimum number of DMs when they all have the same cost. The

resulting problem takes a number of different forms depending on the specific formulation.

In various cases, under particular simplifying assumptions, solutions were obtained from reasonably simple and straightforward computations. If this was not the case, mathematical programming techniques proved an attractive method for obtaining solutions. In particular, generalized network (GN) problems proved most attractive because of the efficient algorithms which exist to solve them. Knapsack problems, and mixed integer linear programs can also be considered.

STEP 5: Information Structures

The solution to the optimization problem yields the number (and identity) of the DMs comprising the single echelon and the assignment of a task or subtask to each DM. The results are expressed formally in terms of an information structure that consists of parallel or alternate processing, or a combination of both. The structure is evaluated to determine whether the tradeoffs between the number of DMs and delays are acceptable, if not, then the designer should return to step 2 and revise the information reduction strategy.

These five steps are applied now to two problems to illustrate the methodology.

7.2 EXAMPLES

7.2.1 Example 7.1

Consider G distinct sources, each source generating a vector of signals. The task is such that each source output has to be processed intact, i.e., it cannot be partitioned. There are M decision makers who can receive the generated signals and none of these DMs can process the output of any of the G sources without being overloaded. A parallel-alternate information structure will be designed using the methodology presented in Section 7.1.

STEP 1: Decomposition of Input

The supersource consists of G synchronized sources that generate vector signals. The mean signal generation rate is δ^{-1}. The elements of the input vector can be partitioned into G sets, each set corresponding to the output of each one of the individual sources. This is the finest grain decomposition of the input.

STEP 2: Information Reduction Strategy

The decomposition of the input vector allows for the parallel processing of the signals generated by the G sources, i.e., the overall task can be divided into G parallel subtasks. No further division into subtasks is possible. Since every one of the G subtasks arriving at a rate δ^{-1} cannot be processed by any DM without causing overload, the second information reduction strategy is involved...the creation of slack resource. Alternate processing of signals generated by each source would allow additional time for each DM to do the processing and therefore, overload may be avoided. The resulting processing mode is an integrated parallel-alternate processing.

STEP 3: Mathematical Model

a) Since alternate processing is assumed for the output of each source, the requirements that all signals be processed reduces to the condition that the sum of the symbol assignment frequencies for the output of each source, q g, m, must be equal to unity. This is expressed as eq. (7.1a) in Table 7.1.

b) In order that no DM be overloaded, the frequency with which each DM receives a signal for processing should be sufficiently low so that his mean processing time is less than or equal to the effective mean interarrival time signals. This condition is given by (7.1d) in Table 7.1.

c,d) Any DMs that receive input for processing with zero frequency are excluded from the single-echelon. Furthermore each DM is allowed to receive inputs from at most one of the G sources. Constraints (7.1b) and (7.1c) guarantee these conditions where the variable Y, g, m, is zero when the m-th decision maker is assigned the output of the g-th source.

TABLE 7.1 PARALLEL-ALTERNATE PROCESSING

Let:

q_{gm} be the frequency with which the m-th DM is assigned the g-th group of components for processing

y_{gm} be the binary variable which indicates whether the m-th DM is assigned the g-th group (y_{gm} = 0 when assigned, 1 when not)

Then:

$$\sum_{m=1}^{M} q_{gm} = 1 \qquad g = 1,2,\ldots,G \tag{7.1a}$$

$$\sum_{g=1}^{G} Y_{gm} = G-1 \qquad m = 1,2,\ldots,M \tag{7.1b}$$

$$Y_{gm} \cdot q_{gm} = 0 \qquad g = 1,2,\ldots,G; \quad m = 1,2,\ldots,M \tag{7.1c}$$

$$0 \leq q_{gm} \leq \zeta_{gm} = \delta/\bar{\tau}_g^m \qquad g = 1,2,\ldots,G; \quad m = 1,2,\ldots,M \tag{7.1d}$$

$$Y_{gm} = 0,1 \qquad g = 1,2,\ldots,G; \quad m = 1,2,\ldots,M \tag{7.1e}$$

STEP 4: Optimization Problem

In a structure such as this, the number of decision makers that can process the incoming signals is a reasonable objective function to be minimized. Note that since, in general, the DMs do not have identical properties, the problem cannot be decoupled into G distinct optimization problems even though no decision maker is allowed to process signals from more than one source. The resulting mathematical programming problem is difficult to solve because of the nonlinearity of one set of constraints.

STEP 5: Information Structures

The information structure can be constructed directly from the solution to the optimization problem (the nonlinear MP). The single-echelon is composed of only those DMs for which the corresponding frequency $q_{g,m}$ is strictly positive. The information structure, Figure 7.1, specifies the decision maker, the group of components g he processes and frequency with which he is assigned these inputs, i.e.

$$\Pi_m^f(t) = \begin{cases} I \quad \text{with probability } q_{f,m} & f = 1,2,\ldots,G \\ & m \in M^f \\ 0 \quad \text{otherwise} & \end{cases}$$

Note that while the inputs from the sources are received by the single-echelon simultaneously, the outputs are not synchronized. Indeed, each DM introduces a different delay; the maximum delay is given by the maximum value over m of

$$\delta(1-q_{gm}) \; / \; q_{gm}$$

If this delay is unacceptable, then more efficient DMs are needed.

7.2.2 Example 7.2

In the second example, a single source producing a high-dimensional vector signal is considered. The vector cannot be partitioned; i.e., the task of processing an input signal cannot be divided into subtasks. However, the alphabet of the vector is very large and can be partitioned into subalphabets. This is a case where an integrated subalphabet - alternate processing strategy can be used.

STEP 1: Decomposition of Input

The alphabet of the supersource is partitioned into G subalphabets, each consisting of different sets of values of the input vector $\underline{x}$. No further decomposition of the input is possible.

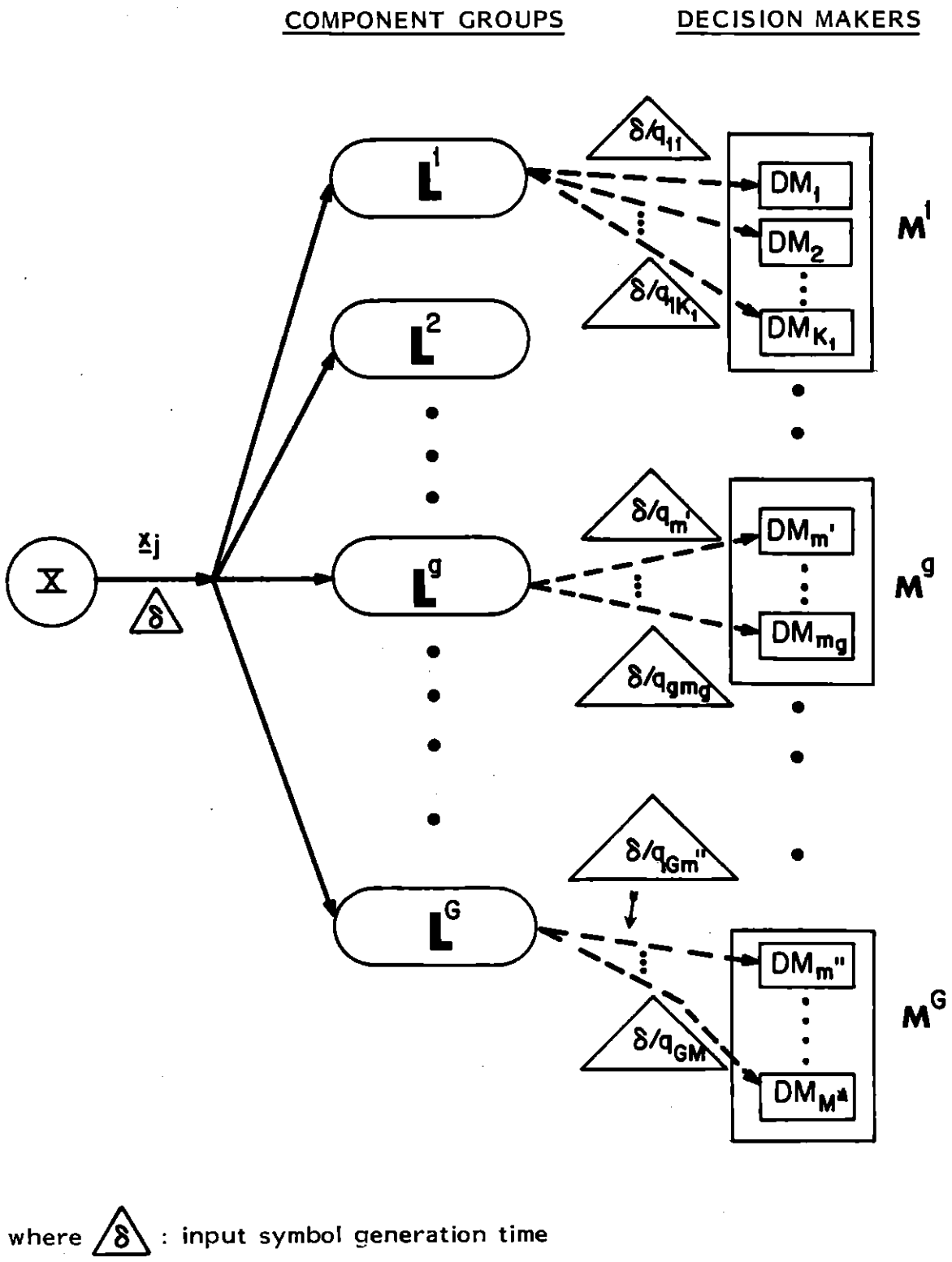


Figure 7.1 Parallel - Alternate Processing

STEP 2: Information Reduction Strategy

No decision maker can process full vectors arriving every δ units of time on the average without being overloaded. Since the input vector cannot be partitioned into subvectors, the alternate procesing mode is the feasible information reduction strategy. The integration of the subalphabet partition and alternate mode manifests itself in the requirement that each DM in the echelon receives inputs from a single subalphabet. Consequently, the solution to this problem leads to the specification of G groups of DMs, denoted by $\mathbf{M}^g$, with each group processing in an alternate mode one subalphabet.

STEP 3: Mathematical Model

a) To ensure that every vector that is generated by the source is processed, the sum of the assignment frequencies of the elements of each subalphabet f over all DMs must equal the probability that a generated vector belongs to that subalphabet (eq. (7. 2a)).

b) To ensure that no DM is overloaded constraints (7.2c) and (7.2e) are introduced. In the latter, the frequency of input assignment to each DM must be less than or equal to the ratio of the average available time and the average required time to process the arriving inputs.

c) A DM is not included in the single-echelon, if the frequency of receiving inputs from any subalphabet is zero. This statement is expressed analytically by constraint (7.2d).

d) Each member of the echelon must be processing elements from one subalphabet only; this is generated by eq. (7.2d).

STEP 4: Optimization Problem

An objective function which minimizes the number of DMs in the single-echelon is a desirable goal. The nonlinearity of two sets of constraints (7.2c,d) makes this problem difficult to solve, however. The binary restrictions on one set of variables introduce further complications. Mathematical programming algorithms exist which can approximate solutions. These include Lagrangian techniques as well as some heuristics. The structure of the single-echelon is shown in Figure 7.2. Note however that if the groups of DMs and the particular

TABLE 7.2 SUBALPHABET-ALTERNATE PROCESSING

Let:

q_{gm} be the frequency with which the m-th DM is assigned elements from the f-th subalphabet for processing

Y_{fm} be the binary variable which indicates whether the m-th DM is assigned inputs from the f-th subalphabet for processing ($Y_{fm} = 1$) or not ($Y_{fm} = 0$)

w_f be the probability that an input vector belongs to the f-th subalphabet

Then:

$$\sum_{m=1}^{M} q_{fm} = w_f \qquad f = 1,2,\ldots,G \tag{7.2a}$$

$$\sum_{g=1}^{G} Y_{fm} = 1 \qquad m = 1,2,\ldots,M \tag{7.2b}$$

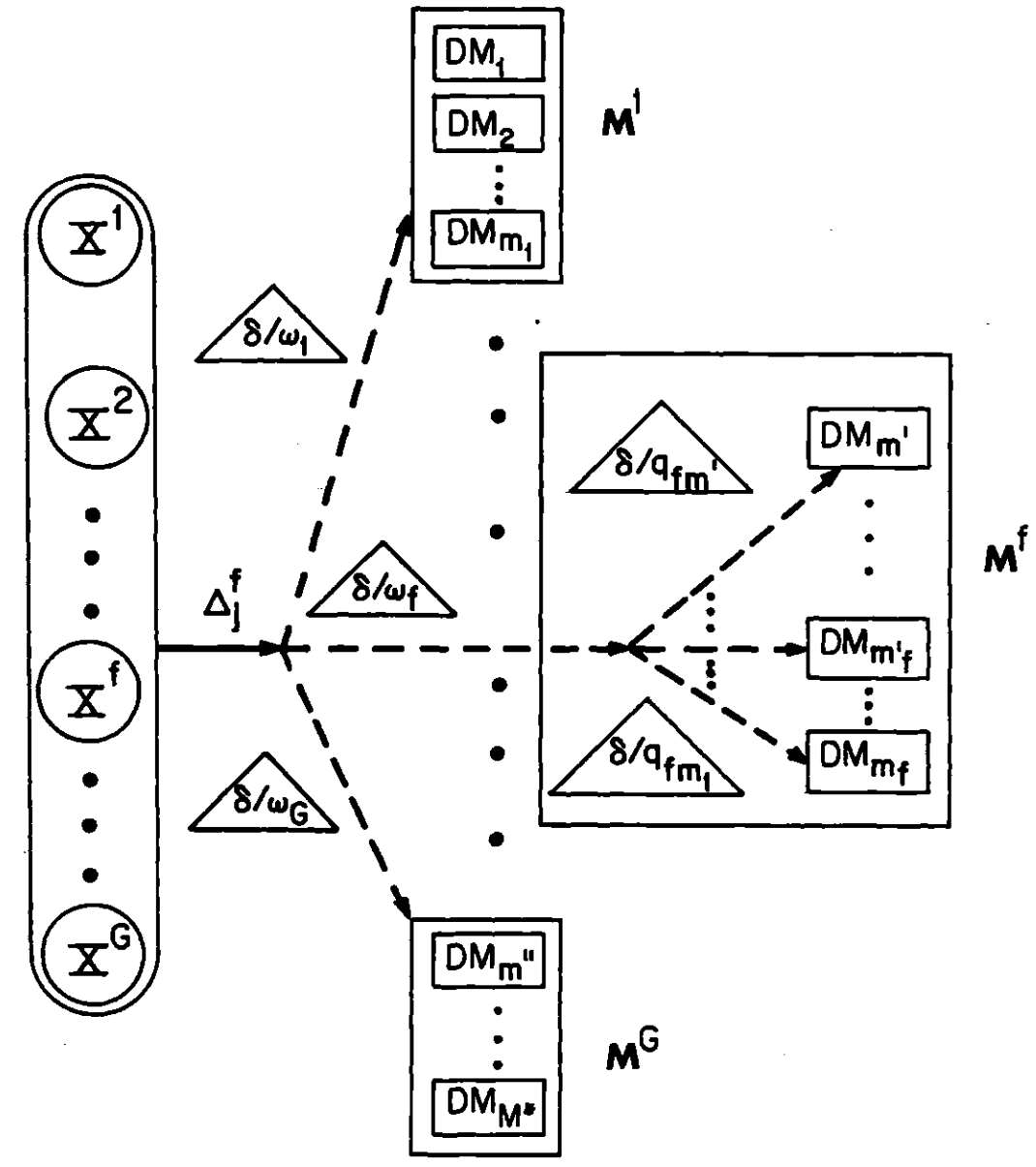
$$\left(\sum_{g=1}^{G} q_{fm} \cdot Y_{fm}\right) \left(\sum_{f=1}^{G} \bar{\tau}_f^m \cdot Y_{fm}\right) \leq \delta \qquad m = 1,2,\ldots,M \tag{7.2c}$$

$$q_{fm}\,(1-Y_{fm}) \doteq 0 \qquad f = 1,2,\ldots,G;\ m = 1,2,\ldots,M \tag{7.2d}$$

$$0 \leq q_{fm} \leq \zeta_{fm} = (\delta/\bar{\tau}_f^m) \qquad f = 1,2,\ldots,G;\ m = 1,2,\ldots,M \tag{7.2e}$$

$$Y_{fm} = 0,1 \qquad f = 1,2,\ldots,G;\ m = 1,2,\ldots,M \tag{7.2f}$$

subalphabets are prespecified this problem can be decoupled. (It is these coupling constraints that introduce the nonlinearity.) If the problem is decoupled, then it reduces to the one introduced in Section 6.5, Alternate



where δ : input symbol generation time

Figure 7.2 Subalphabet - Alternate Processing

Processing with Specialization. The latter, as shown in Section 6.5 can be solved directly.

STEP 5: Information Structures

The solution to the optimization problem the nonlinear MP) specifies which DMs are processing what subalphabets and with what frequency. Decison makers assigned to process elements from the f-th subalphabets can be grouped to form the set $\mathbf{M}^f$. For each DM in each of these groups the corresponding information structure is constructed as follows:

$$\Pi_m^f(t) = \begin{cases} I & \text{with probability } q_{f,m} \\ 0 & \text{with probability } (1-q_{f,m}) \end{cases} \quad \begin{matrix} f = 1,2,\ldots,G \\ m \in \mathbf{M}^f \end{matrix}$$

The maximum delay introduced is, again,

If this delay is not tolerable than more efficient DMs must be introduced into the S-E. It may be necessary to find efficient DMs in one group only. If the minimum q_{fm} for each f , excluding the group in which q^*_{fm} if found, is sufficiently large (implying the maximum delay in each of these groups is sufficiently small) then only minor modifications to the structure may permit this strategy to be imposed rather than seeking alternative ones.

7.3 CONCLUSIONS

In the previous section complex information structures were constructed. The more complex design of these structures resulted from the particular properties possessed by the associated DMs and inputs of each example.

In example 7.2 alternate processing was the information reduction strategy selected to avoid DM overload. If the delay time, introduced as a consequence of this strategy was unacceptably large the designer would

be required to consider other strategies. Had it been possible to partition the input vector, subtasks could have been created and a parallel processing mode implemented.

A distinct group of DMs would be responsible for processing inputs generated from a particular subalphabet only. The input symbols of the associated subalphabet would be partitioned among the DMs of the specified group. Each DM within this group would be assigned some partition of components of the input symbol to process. Figure 7.3 illustrates this integration of modes. A consequence of this mode integration is that no output delay is introduced. The number of DMs required to process may be large however.

Three directions for further research are: 1) the specification of the properties of the inputs 2) the characteristics of the DMs 3) the analysis of complex information structures. These directions will lead to refinement of the methodology for designing the information structures.

The next major step in this research is the integration of the single-echelon with other parts of the organization. The single echelon is responsible for transmitting the processed inputs to the appropriate destinations within the organization. This transmission of processed data to other members in the organization is referred to as serial processing [preprocessing]. Preprocessing was discussed briefly in Chapter 1; it is another method of coping with overload. [5]

The design of multiechelon structures must enable each echelon to process its information without overload. Additional constraints imposed by the echelons to which the information is transmitted further complicate the problem. Constraints are imposed on the entire organization. These include the allocation of time to the various echelons for performing specified tasks. Each echelon has its own internal constraints.

The two directions cited focus on the macroscopic part of an organization. Alternatively, the analysis of the individual DM, his

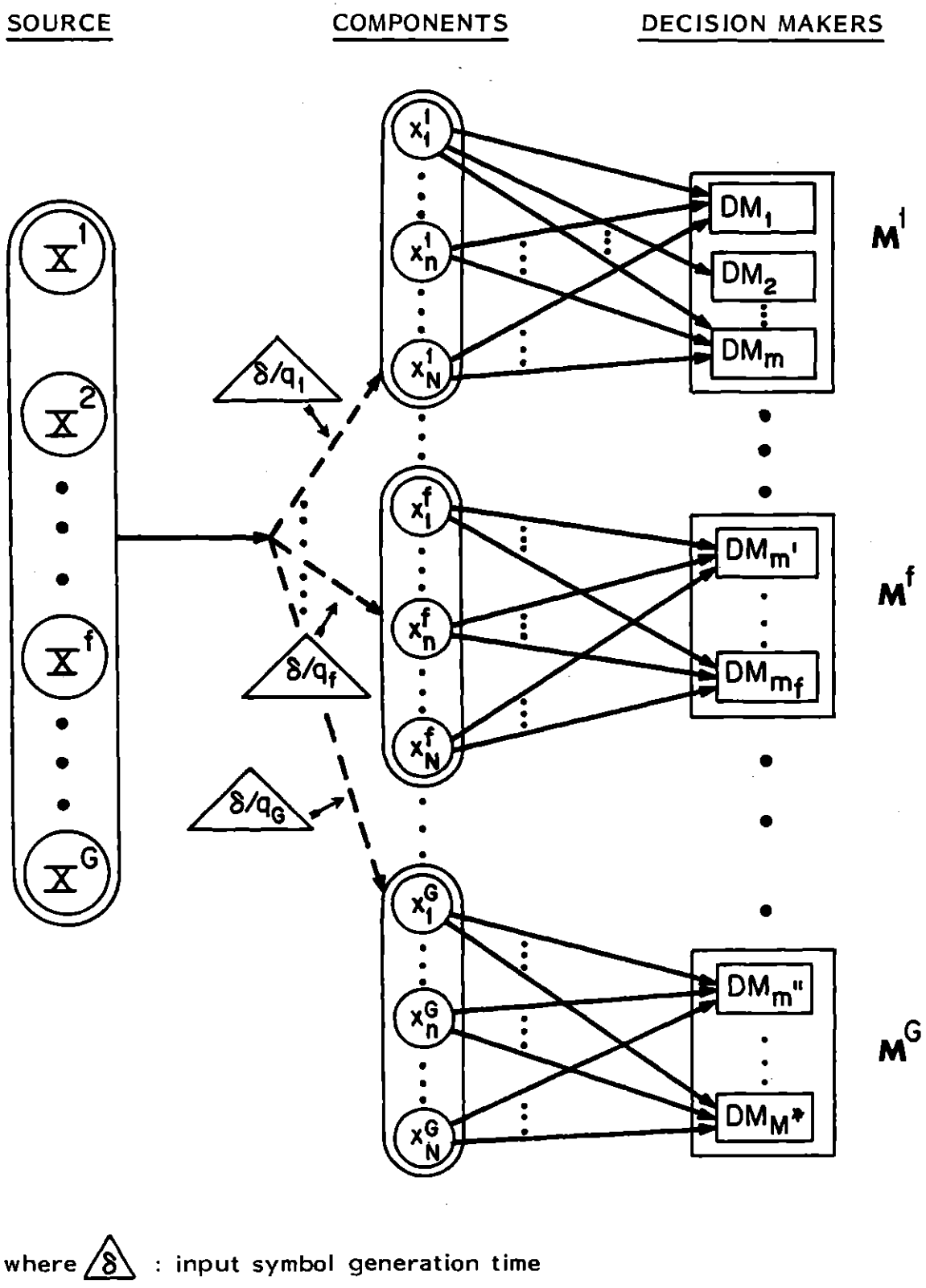


Figure 7.3 Subalphabet - Parallel Processing

processing capabilities and how to make his more efficient are other areas for further exploration For example the integration of man and machine to perform a task, tradeoffs and benefits in terms of costs and other factors remain to be more fully explored.

A methodology for designing the information structures who comprise the single echelon has been presented. This methodology is based on the properties of the inputs, the characteristics of the available DMs, and the constraints imposed by the task on the organization. These in turn restrict the number of feasible information structures. For example, if no delay in producing output can be tolerated, parallel processing must be included in the design strategy. However,, this processing mode can be implemented only if the input can be partitioned. If a delay can be tolerated then a strategy which includes the alternate processing mode is feasible also.

"Begin at the beginning...and go on til you come to the end: then stop"
Lewis Carroll

REFERENCES

1. R. F. Drenick, "Organization and Control," in Directions in Large Scale Systems Ho and Mitter, Eds. Plenum Press, N.Y., 1976.

2. J. Galbraith, Designing Complex Organizations, Addison-Wesley, New York, 1973.

3. D. Katz and R. L. Kahn, The Social Psychology of Organizations, John Wiley and Sons, New York, 1966.

4. E. E. Lawler III and J. G. Rhode, Information and Control in Organizations, Goodyear Publishing Company, Inc., Pacific Palisades, CA, 1976.

5. T. B. Sheridan and W. R. Ferrell, Man-Machine Systems, The M.I.T. Press, Cambridge, MA, 1974.

6. R. V. Hogg and A. T. Craig, Introduction to Mathematical Statistics, Macmillan, New York, 1968.

7. S. P. Bradley, A. C. Hax and T. L. Magnanti, Applied Mathematical Programming, Addison-Wesley, Reading, MA, 1977.

8. R. S. Garfinkel and G. L. Nemhauser, Integer Programming, John Wiley and Sons, New York, 1972.

9. F. Glover,J. Hultz,D. Klingman and J. Stutz," Generalized Networks: A Fundamental Computer-Based Planning Tool," Management Science, Vol. 24, No. 12, August,1978, pgs. 1209-1220.

10. B. L. Golden and T. L. Magnanti, Network Optimization, Unpublished Class Notes, M.I.T., Cambridge,MA, 1980.

11. W. S. Jewell, "Optimal Flow Through Networks with Gains," Operations Research, Vol. 10, 1962, pgs. 476-497.

12. R. E. Davis, D. A. Kendrick and M. Weitzman, "A Branch-and-Bound Algorithm for Zero-One Mixed Integer Programming Problems," Operations Research Vol. 19, 1969, pgs. 1036-1044.

13. F. Glover, J. M. Mulvey, "Equivalence of the 0-1 Integer Programming Problem to Discrete Generalized and Pure Networks," Research Report No. EES-79-5, Princeton University, Princeton, N.J., 1979.

14. J. F. Shapiro, A Survey of Lagrangean Techniques for Discrete Optimization, M.I.T., Cambridge, MA, 1977.

15. M. F. Fisher, "Lagrangian Relaxation Methods for Combinatorial Optimization," The Wharton School, University of Pennsylvania, 1978.

16. M. Eisen, Elementary Combinatorial Analysis, Gordon and Breach, New York, 1969.