A STUDY OF THE APPLICABILITY AND SYNTHESIS OF REDUNDANT, THRESHOLD LOGIC DECISION-MAKERS

By

JOHN L. YOUNGBLOOD

Bachelor of Science
Arlington State College
Arlington, Texas
1963

Master of Science
Oklahoma State University
Stillwater, Oklahoma
1965

Submitted to the faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the degree of
DOCTOR OF PHILOSOPHY
May, 1967

A STUDY OF THE APPLICABILITY AND SYNTHESIS OF

REDUNDANT, THRESHOLD LOGIC

DECISION-MAKERS

Thesis Approved:

A. M. Breipohl
Thesis Adviser

Richard L. Cummins

Kenneth A. McCollom

E. K. McFarlan

D. D. Dunham
Dean of the Graduate College

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my adviser, Professor A. M. Breipohl, for his untiring efforts in my behalf. He has given me invaluable encouragement, counsel, and aid during my research.

I also wish to thank the other members of my graduate committee, Professors W. A. Blackwell, K. A. McCollom, R. L. Cummins, and Jeanne L. Agnew for their assistance and encouragement. Professor E. K. McLachlan has also been of valuable service during the final stages of my work.

I am very grateful for the financial assistance which I have received during my graduate studies. Sandia Corporation has provided the funds necessary to perform the research reported here, and the Department of Health, Education, and Welfare has provided a three-year graduate fellowship.

Finally I wish to acknowledge my wife Sharon's patience, sacrifices, and encouragement.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Statement of the Problem. The basic concept which provided the inspiration for the research reported here is the concept of distributed redundancy. It is felt that this type of redundancy is achieved by devices in which each individual component participates in many (or all) functions, and all functions are produced by many (or all) components. If a component fails in a device so structured, no single function is totally lost; rather, all (or many) functions are incrementally degraded. This concept of redundancy has been considered in the majority of the developments used in the solution of the problem under investigation.

The basic problem can be divided into two major areas with the results of one being the justification for the effort spent on the other. The first problem area involves distinguishing between the areas of applicability of decision-makers possessing distributed redundancy characteristics and the areas of applicability of conventional redundant decision-makers. The results of this problem define a second problem which is involved with investigating a particular approach for enlarging the areas of applicability of decision-makers with distributed redundancy.

As a result of these two problem areas, the following chapters and appendices fall into one of two categories. Chapter II is devoted

exclusively to the comparison of the two decision-maker types. The remaining problem area is the topic for Chapters III and IV and the appendices.

The overall decision-making system model consists of the decision-maker itself and a set of n binary, sensor channels. These sensor channels measure traits of a physical phenomenon and convey this information to the decision-maker in an unreliable manner. It is assumed for simplicity that the channels are statistically independent and identical. For a given decision problem, the decision-maker is designed so that a risk function is minimized as the decision-maker makes either of two possible decisions. The entire system is binary.

The decision-maker which is considered to be conventional has certain basic characteristics which distinguish it from the decision-maker which, hopefully, models distributed redundancy. The conventional decision-maker makes binary decisions based on binary information as received through n inputs. Corresponding to each of these n inputs is at least one unreliable switching element. Each of these switching elements is interconnected with other switching elements and whether it is open or closed simply determines which of two possible inputs is fed into the interconnection. Thus, the designing of the decision-maker to make certain decisions is done by properly interconnecting the input switching elements and, possibly, other secondary elements. Of course, the interconnection may fail also; and it is therefore modeled by an unreliable channel. In the comparison which follows, it is assumed that the interconnection channel is significantly more reliable than that of the decision-maker with distributed redundancy. This is a fairly conservative assumption.

Another characteristic of the conventional network of switching elements is that it can implement any Boolean function with a relatively small variation in complexity. This is not the case for the type of decision-maker considered in Chapters III and IV.

The manner in which redundancy is incorporated in networks of switching elements is to use parallel and series combinations of redundant inputs. Basically, parallel combinations are used to prevent misfires and series combinations to prevent false alarms. Both of these failures must be taken into consideration in most decision systems. Consider, for example, the problem of fusing of nuclear weapons. It is obvious that failures to detonate and unwanted detonations are both failures that are of utmost importance. Examples such as this provide the reason for the forms of redundancy considered here. It should be emphasized that specific redundancy implementations are not in general considered in the comparison of Chapter II; rather, consideration is given to the overall or basic characteristics of decision-makers which can incorporate the types of redundancy being evaluated.

The unconventional decision-maker with distributed redundancy also is modeled in such a way that its inherent characteristics are exhibited to a certain degree. The most important characteristic is the lack of switching elements corresponding to each input to the decision-maker. As a result of this characteristic, the interconnection of the inputs must be more complex and is thus less reliable. Furthermore, this complexity is dependent upon specific problems. Some problems require a considerably less complicated interconnection of decision-maker inputs than for other problems. The more complicated ones provide some of the justification for the work presented in Chapters III and IV.

The form taken by the redundancy in decision-makers which are intended to possess distributed redundancy is considerably different from that of networks of switching elements. Of course, the repetition of input channels provides overall system redundancy. However, one of the most significant features of distributed redundancy is that it can be achieved through overdesign but without a blind repetition of components. The particular type of decision-maker proposed in Chapters III and IV possesses this feature to a certain degree as shown by Example 4.7.2. In this particular example it is necessary to use a relatively complex decision-maker. But, by a slight amount of overdesign in the nonredundant decision-maker, reliability of the desired level can be achieved far more easily than if the nonredundant realization had been minimal. This is a unique characteristic of distributed redundancy. A little overdesign buys a lot in terms of reliability.

The actual decision-maker which is proposed to model distributed redundancy consists of a network of threshold logic units (TLU's). A TLU is a device which presents a linear, weighted sum of its inputs to a threshold detector. The threshold detector produces a binary output depending upon the level of the weighted sum relative to a threshold level. Such a device is shown in Figure 1.1.1.

There are several reasons for the attention that TLU's have recently received. The reason which is relevant here is that the TLU is thought to possess distributed redundancy.

Rather than single TLU's, networks of TLU's are used primarily here because of the limitations of the use of single TLU's. The primary limitation lies in the fact that a single TLU cannot implement all Boolean functions. This is discussed in more detail in Chapter III.

The specific network configuration used is a two-layer network as shown in Figure 1.1.2. This configuration is sufficient to realize all Boolean functions.

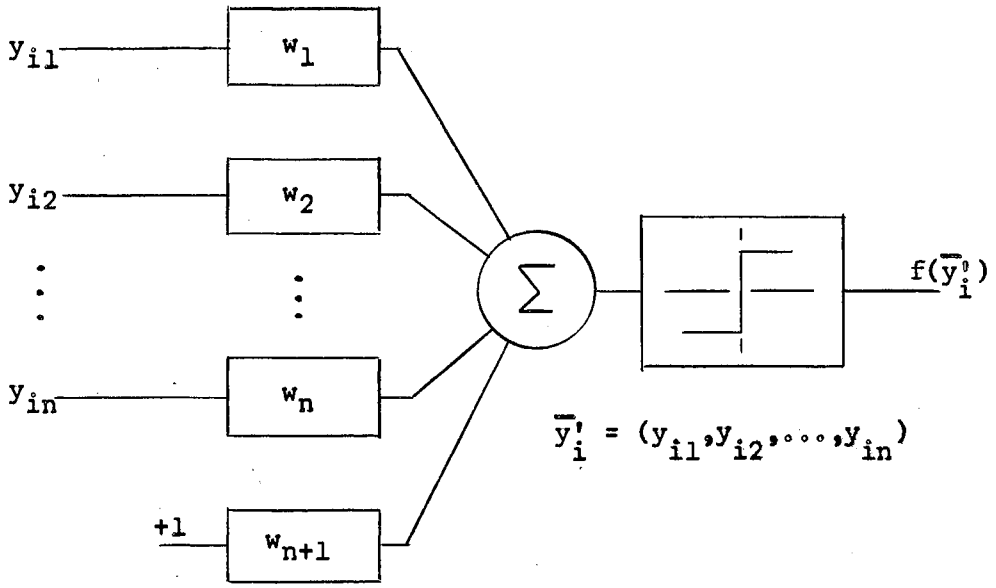


Figure 1.1.1. Threshold Logic Unit

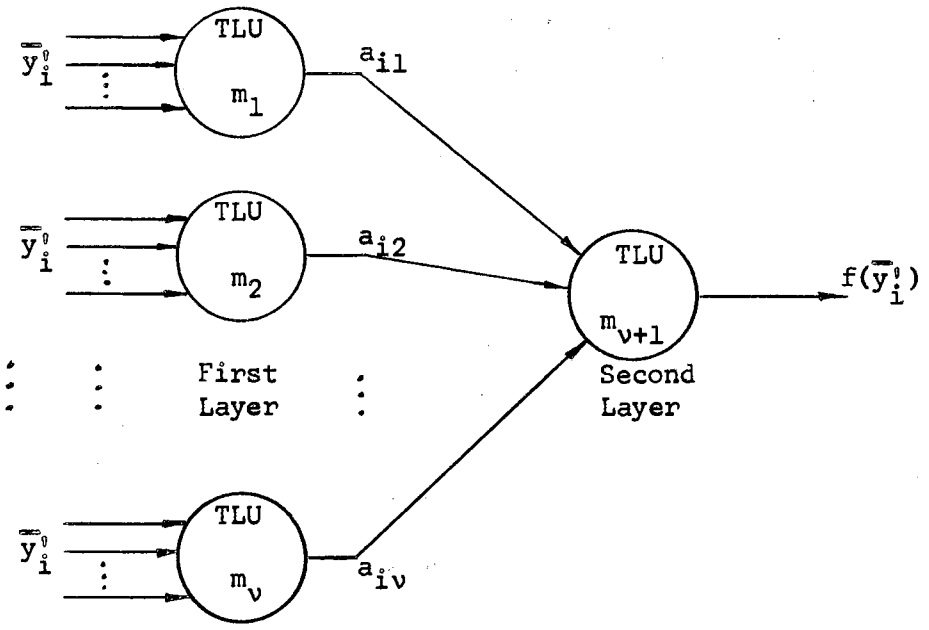


Figure 1.1.2. A Two-Layer TLU Network

As mentioned previously, a certain amount of overdesign in decision-makers of this type is significant to the amount of redundancy which is necessary to achieve a certain degree of reliability. The addition of redundancy to two-layer TLU networks is the specific topic for Chapters III and IV and the appendices. The problem reduces to synthesizing a set of γ redundant TLU's to add to a set of ν TLU's in the first layer as shown in Figure 1.1.3. These TLU's are synthesized so that single error-correction is achieved for errors between the first and second layers.

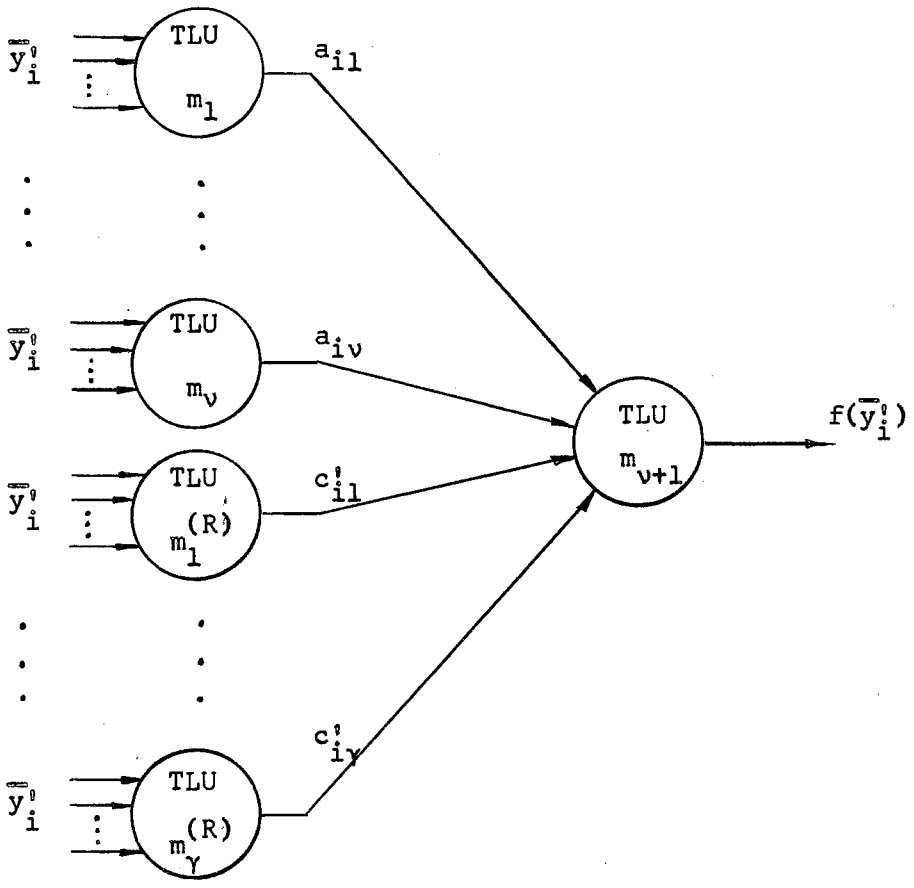


Figure 1.1.3. TLU Network With Redundancy

It is important to note that the justification for adding redundancy to the first layer rather than the second is that the reliability of any device is no greater than that of any of its series stages. For example, if the final layer is triplicated and a vote is taken on the three outputs, the reliability of the resulting system is no better than that of the device which performs the voting.

The technique developed here for adding redundancy to TLU networks is based strictly upon the system of linear inequalities for which the weights of the second layer TLU $m_{\nu+1}$ must be a solution. This system of linear inequalities, along with certain realizability requirements, is the foundation for the approach taken.

In summary, the first of the two problem areas here consists of comparing the two decision-maker types and finding the areas of applicability of each. The basic philosophy behind this comparison is that the decision-makers should be compared on the basis of their fundamental characteristics rather than on the characteristics of specific implementations of each.

The second problem area involves an investigation of the problem of redundancy in two-layer TLU networks. This investigation is justified by the conclusions of the first problem area.

1.2 Previous Work in the Area. The majority of the work done in the areas investigated here has been in the area of threshold logic and in redundant networks of computing elements. The theory of threshold logic and linear inequalities is of primary interest here because of the particular theoretical redundancy approach taken. Some of the contributions which are related to the theory used are those of Paull and

McCluskey (19), Gabelman (7), Chow (3), Hopcroft and Mattson (13), and Highleyman (12). In these papers the necessary and sufficient condition, upon which the redundancy technique is based, is mentioned either directly or indirectly. However, only Hopcroft and Mattson (13) have applied the theory with any similarity to the application here. To this author's knowledge there has been no direct application of the theory presented in these papers to the problem of introducing redundancy in TLU networks.

There are several redundancy techniques which have been applied to TLU networks. Perhaps the technique which has the greatest similarity to the technique used here is due to Bargainer and Coates (5). However, their technique is restricted in the sense that the second layer TLU has its design specified and thus restricts the freedom of choosing redundant TLU's which are best suited to the task.

Other techniques by Pierce (20, 21, 22), Jenson (14), and Knox-Seith (16), use adaption and additional layers of TLU's or other elements. These approaches still do not take into consideration the basic problem in linear inequalities as is done here.

Wilcox and Mann (25) have edited a collection of papers on redundancy which contains some work similar to that mentioned above.

1.3 Method of Solution. The solution to the problems consists mainly of establishing a model of the system and in investigating this model. This is true for the problem concerning the areas of applicability of decision-makers with distributed redundancy and for the problem of introducing redundancy in TLU networks.

For the problem of the areas of applicability of decision-makers,

the first step is to develop system models which demonstrate the desired characteristics of each decision-maker. Then, based on these models, risk functions are derived from a knowledge of the system parameters and a cost function. It turns out that only one general risk function is necessary and that this function can be altered to apply to either of the two decision-maker types by simply changing the form in which some of the parameters appear in the function.

In order to evaluate these risk functions, it is necessary to specify desired decision-maker outputs corresponding to each combination of inputs to the sensors for the system. As a result, two types of problems are considered which are representative of the extremes of the complexity of decision problems which can occur.

Corresponding to each of these two types of problems, the two risk functions can be evaluated as functions of the significant parameters in each of the decision-makers. This evaluation is used to compare the two decision-makers and to determine the areas of applicability on the basis of the number of input channels, the type of problem, the switching-element parameters, and the parameters associated with the assumed model for the interconnection of the decision-maker inputs.

Very briefly, this comparison reveals that the area of applicability of TLU decision-makers requires that networks of TLU's be used rather than a single TLU. This adds to the complexity of the TLU decision-maker and implies that it is necessary to consider ways to introduce redundancy in TLU networks in order to maintain any advantage that they may possess.

The problem of introducing redundancy in TLU networks is approached by first developing the mathematical model of the desired portion of the

network. This model as developed in Chapter III takes into consideration the possibility of failures in both the original TLU's in the non-redundant realization and in the redundant TLU's. Given the mathematical model, which consists of a system of linear inequalities, a basic theorem in the theory of linear inequalities is applied to the model. The result is an additional set of linear inequalities which place constraints on a vector which specifies the design of a redundant TLU.

If it is not possible to satisfy the inequality constraints with a single TLU, then the theory must be extended to obtain the constraints on more than one vector (thus, more than one TLU is added). In Chapter IV this extension is considered, and an iterative approach is suggested for selecting the redundant TLU's.

The entire development of the redundancy synthesis algorithm is done with the least complicated types of failures as possible. Due to the flexibility of the algorithm, it is possible to extend the allowable types of failures to more complicated situations. This extension occurring in Section 4.7 also permits the use of certain examples which better illustrate the utility of the synthesis algorithm.

The appendices contain detailed developments and computer programs which are not appropriate for including as a chapter. Appendix A presents the basic theory of linear inequalities which is applied here. A technique is developed for computing a set of vectors which are used in the inequality constraints mentioned above. Since the use of this technique is quite laborious, a computer program has been written to perform the necessary calculations. A flow-chart and a listing of this program are presented.

Appendix B presents the development of a technique for computing

the vectors used in the inequality constraints in a manner which greatly simplifies the use of the program in Appendix A. It is shown that certain redundant input data can be eliminated and that the appropriate compensation can be made on the output data.

In Appendix C a very special case of the technique developed in Appendix A is considered. This special case arises in a computer program in Appendix D.

Appendix D is a presentation of two computer programs which perform searches for vectors which specify the design of the redundant TLU's. These programs differ slightly in the criteria used in the searches.

1.4 Suggestions to the Reader. The manner of reading this report depends upon the reader's area of interest. If the reader is only interested in the areas of applicability of distributed redundancy, then Chapter II should receive the greatest attention with only brief attention being given to the remainder of the work presented. If the redundancy technique is to be considered, then Chapter III, Appendix A, and Chapter IV should be read in that order. The remaining appendices can be read as it becomes necessary.

In the notation used here a lowercase letter with an upper bar as with $\bar{z}_i$ represents a row vector whose components are $z_{i1}, z_{i2}, \cdots, z_{in}$ where n is the number of elements in the vector. If it is necessary to write such a vector as a column, it is written $\bar{z}_i^T$. A matrix is represented by an uppercase letter with a lower bar as in $\underline{A}$. The transpose of $\underline{A}$ is given by $\underline{A}^T$. A set of elements is denoted by the capital letter S with appropriate subscripts or superscripts for identification. Other less general notation is introduced as it becomes necessary.

CHAPTER II

A COMPARISON OF TWO TYPES OF BINARY DECISION-MAKERS

2.1 Introduction. This chapter is devoted to a comparison of decision-makers which contain switching elements on each input to decision-makers such as threshold logic unit networks. The latter do not contain switching elements on each input but have a more complicated interconnection of these inputs. Therefore, the comparison is based on the relative merits of these differences in the two types of decision-makers and on the corresponding variations in risk functions as the number, n, of input channels is varied.

The comparison is made in two distinct situations with regard to the nature of the basic decision problem. These two situations are explained in Section 2.2 along with a general description of the decision making system.

Section 2.3 presents the models of the two separate decision-maker systems upon which the risk functions used for the comparison are based. A general risk function R is developed in such a way that it can be applied to either of the two systems by simply making appropriate substitutions into R. Also presented are the conditions for optimality of R.

One of the two decision-making problems considered here is actually a special case of the other; therefore, in Section 2.4 the risk

function R, developed in Section 2.3, is restricted to apply to this particular problem. The resulting conditions for optimality are presented along with an example. The example considers a commonly used relay contact or switching element network and shows that it is sub-optimal except under certain specific conditions.

Using the restricted risk function developed in Section 2.4, the appropriate substitutions are made in Section 2.5 so that the risk functions for the two decision-maker types can be compared. Under the conditions of the particular decision-making problem of Section 2.4, the two risk functions are compared for large and small values of n separately. For small values of n, families of curves of risk as a function of n can be drawn to allow a determination of where the trade-offs exist between the two decision-maker types. For large n the risk functions tend to converge allowing the comparison of the decision-makers to be made without the aid of curves as in the case for small n.

Finally, in Section 2.6 the general risk function developed in Section 2.3 is adapted to each of the two decision-maker types; and a more general decision-making problem than that of Sections 2.4 and 2.5 is considered. This section establishes more distinct advantages of threshold logic decision-makers and at the same time reiterates a need for improving their reliability.

Thus this chapter establishes the areas of applicability of the two types of decision-makers and sets the stage for a detailed investigation of a technique for improving the reliability of threshold logic network decision-makers. This investigation is the topic for the remaining chapters.

2.2 The Binary Decision Problem. It is assumed that the decision-makers discussed here are intended to make binary decisions based on binary information as received through n statistically independent sensor channels as shown in Figure 2.2.1. The observed vector $\bar{z}_i$ has binary components (+1 or -1) $z_{i1}, z_{i2}, \ldots, z_{in}$, each of which represents some trait of the physical phenomenon. It is convenient to think of each of the vertices of the resulting n-cube as a state of nature. The task for the decision-maker is then to correctly classify the states, which can occur, into one of two categories and to do this in an optimum fashion. The criterion of optimality depends upon the agreement of the desired, $f_d(\bar{z}_i)$, and actual, $f_a(\bar{z}_i)$, Boolean functions associated, respectively, with the classification of the ith state of nature and with the output of the decision-maker.

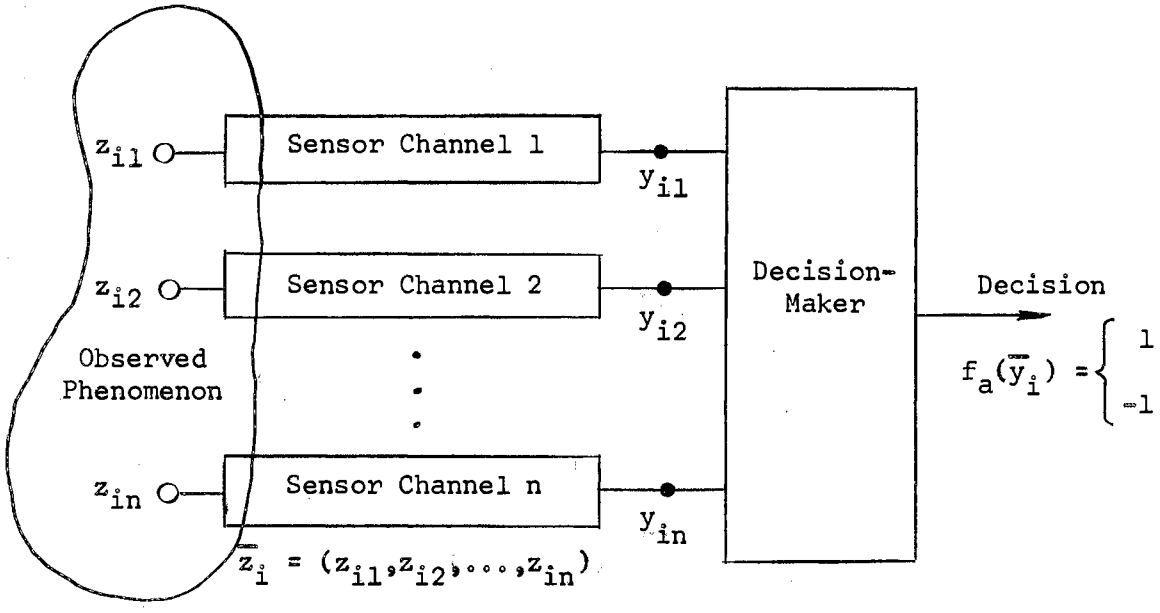


Figure 2.2.1. Model of the Decision-Maker

At any given time any $\bar{z}_i$ which can occur belongs to a set

$$S^{+}_{f_d} = \{\bar{z}_i \mid f_d = +1\}$$

or

$$S^{-}_{f_d} = \{\bar{z}_i \mid f_d = -1\}$$

depending upon whether the desired classification f_d is +1 or -1, respectively. Those states of nature which cannot occur belong to a set of "don't cares"

$$S^{d} = \{\bar{z}_i \mid \bar{z}_i \text{ never occurs}\} \quad .$$

Corresponding to a particular value of f_d, a vector $\bar{z}_i$ can occur with probability

$$P[\bar{z}_i \mid f_d = +1] = \begin{cases} P[\bar{z}_i \mid \bar{z}_i \in S^{+}_{f_d}] \\ 0 \text{ if } \bar{z}_i \notin S^{+}_{f_d} \end{cases} \qquad (2.2.1)$$

or

$$P[\bar{z}_i \mid f_d = -1] = \begin{cases} P[\bar{z}_i \mid \bar{z}_i \in S^{-}_{f_d}] \\ 0 \text{ if } \bar{z}_i \notin S^{-}_{f_d} \end{cases} \qquad (2.2.2)$$

where

$$\sum_{i=1}^{m^{+}} P[\bar{z}_i \mid \bar{z}_i \in S^{+}_{f_d}] = 1$$

and

$$\sum_{i=1}^{m^{-}} P[\bar{z}_i \mid \bar{z}_i \in S^{-}_{f_d}] = 1 \quad .$$

The integers m^+ and m^- are the numbers of elements in $S^+_{f_d}$ and $S^-_{f_d}$, respectively.

The use of the word "classification" stems from the fact that the decision problem is viewed as a pattern classification problem. The states of nature are actually patterns in an n-dimensional pattern space. The same is true of the vectors $\overline{y}_i$ on the outputs of the sensor channels.

There are two specific cases considered here with regard to the location of patterns in the sets $S^+_{f_d}$, $S^-_{f_d}$, and S^d. The first case called the "almost empty pattern set" is essentially a representation of a decision-maker and sensor system which has n redundant sensors. All n sensors are designed to measure the same trait of the physical phenomenon. In this case there are only two states of nature which can occur; therefore,

$$S^+_{f_d} = \{\overline{z}_{2^n}\} = \{(1, 1, \ldots, 1)\} \quad ,$$

$$S^-_{f_d} = \{\overline{z}_1\} = \{(-1, -1, \ldots, -1)\} \quad ,$$

and all other $\overline{z}_i$'s are in S^d. However, all 2^n allowable vectors $\overline{y}_i$ can occur since it is assumed here that the sensor channels are imperfect. The decision-maker must be designed to classify the $\overline{y}_i$'s such that an optimality criterion is satisfied.

An example of this case is a set of four sensors which are designed to detect the occurrence of one of the two possible states of nature and a set of four relay contacts interconnected so that a decision is made according to some specified decision law. The sensors might be radars, and the contacts could be arranged in the classic "quad" shown

in Figure 2.2.2. A detailed analysis of this particular example is presented in Section 2.4.

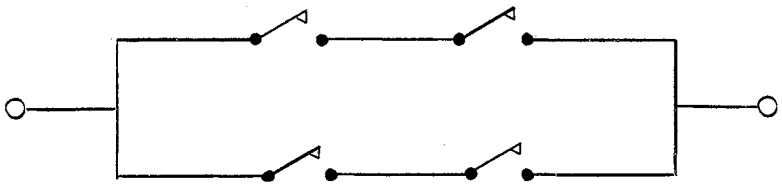


Figure 2.2.2. The "Quad" Switching Network

The second case considered here is the "full pattern set" case where all 2^n allowable states of nature can occur. It is assumed that the classification of the states of nature is a majority rule situation; that is, n is odd and those $\bar{z}_i$'s with (n + 1)/2 or more "+1" components are in the set $S^+_{f_d}$ while those with (n + 1)/2 or more "-1" components are in the set $S^-_{f_d}$.

2.3 The General System Models. The comparison which is made here is between the risk function R_S associated with a decision-maker which has a switching element associated with each input channel and the risk function R_T for a decision-maker without switching elements on each input. Figures 2.3.1 and 2.3.2 show the detailed decision-maker system models from which R_S and R_T are derived. Notice that in both figures there is no uncertainty involved in the classification of nature or in the decision-maker's designed classification. The first classification

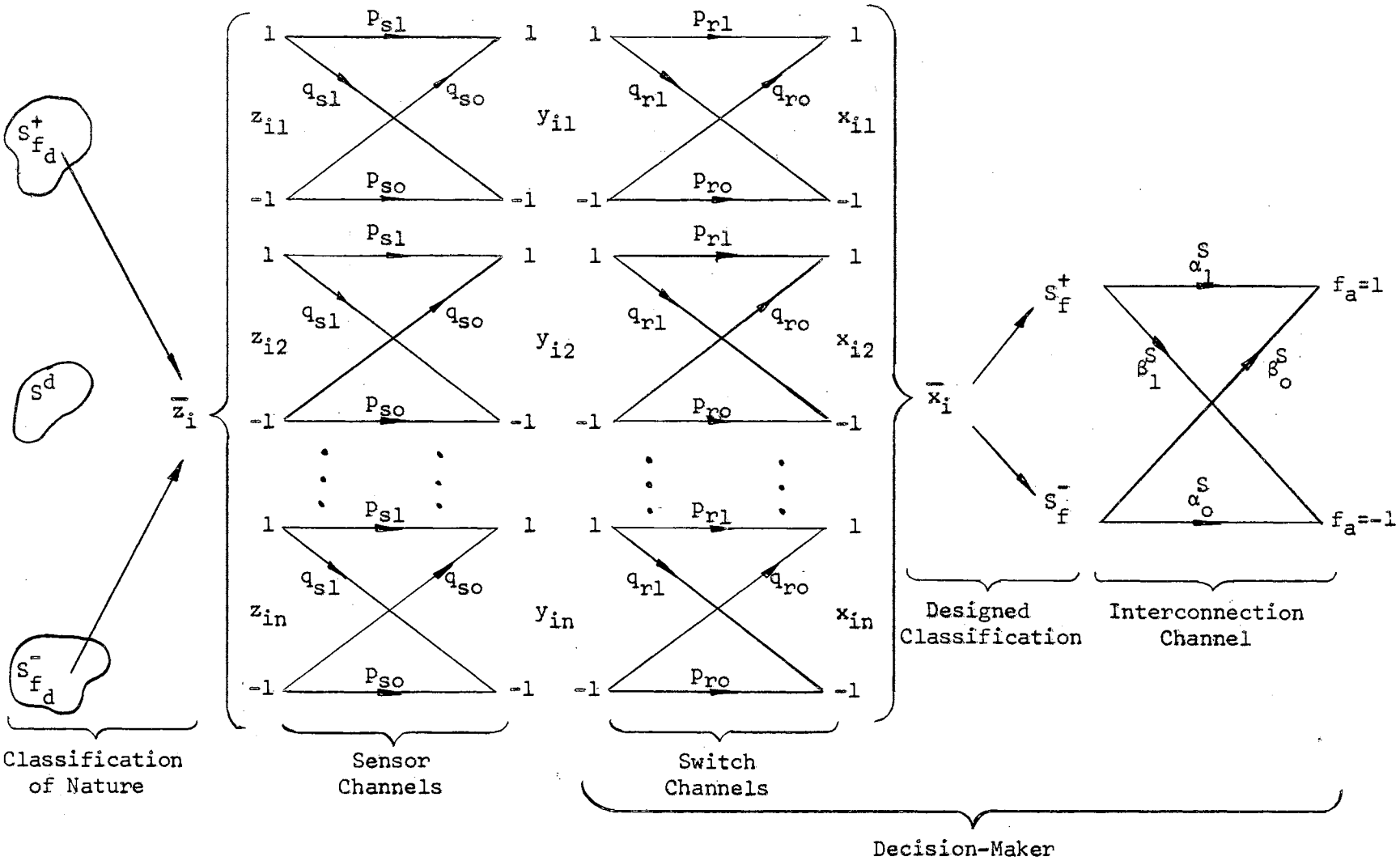


Figure 2.3.1. Model of Decision-Maker System With Input Switching Elements

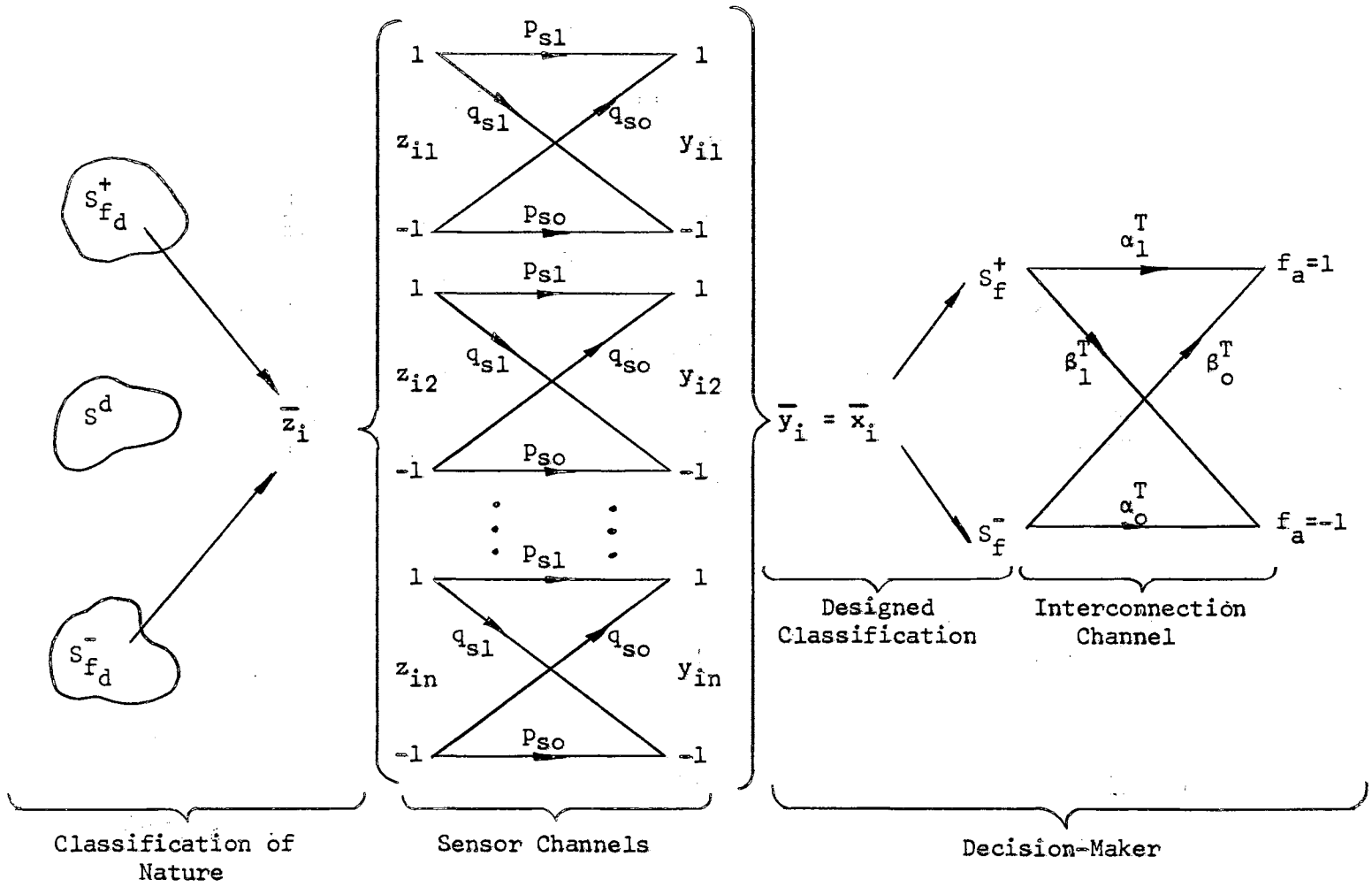


Figure 2.3.2. Model of Decision-Maker System Without Input Switching Elements

is known for a given problem, while the latter is selected or designed so that a risk function is minimized. The latter classification, the decision-maker's designed classification, classifies each of the vectors $\overline{x}_i$ into one of the sets

$$S_f^+ = \{\overline{x}_i \mid f(\overline{x}_i) = +1\}$$

or

$$S_f^- = \{\overline{x}_i \mid f(\overline{x}_i) = -1\}$$

where $f(\overline{x}_i)$ is equal to $f_a(\overline{z}_i)$ when no failures occur in the decision-maker interconnection channel. The Boolean function $f_a(\overline{z}_i)$ is the function appearing at the output of the decision-maker interconnection channel.

In a practical situation the parameters α_o^S, β_o^S, α_1^S, and β_1^S in Figure 2.3.1 or α_o^T, β_o^T, α_1^T, and β_1^T in Figure 2.3.2 may be difficult to determine. This model for the decision-maker interconnection channel is intended to simulate failures which cause all $\overline{x}_i$'s to produce the same output. For example, if the decision-maker is a relay contact network, any given set of contact positions ($\overline{x}_i \in S_f^+$ or $\overline{x}_i \in S_f^-$) might produce a zero output as the result of an open circuit on the output of the network. Similarly, one output might always be present because of a short circuit. This model does not take into consideration all types of failures; however, a model more general in terms of types of failures would be highly dependent upon the specific problem at hand. For example, the channel parameters could be dependent upon the particular vector $\overline{x}_i$ or the number of input channels.

The risk function for the comparison is developed in general so

that it can be applied to either the system in Figure 2.3.1 or Figure 2.3.2. This risk function is based on a loss function

$$L\,[f_a = +1\ ,\ \ f_d = -1] = C_f \quad \text{(false alarm)}$$

$$L\,[f_a = -1\ ,\ \ f_d = +1] = C_m \quad \text{(misfire)}$$

$$L\,[\text{otherwise}] = 0\ .$$

Although it is not assumed to be true here, this loss function could be a function of the state of nature $\overline{z}_i$. In other words, it could cost more to incorrectly classify certain vectors than for others.

The risk R (R_S or R_T), which is the expected value of the loss, is

$$R = E\,[L] = C_f\,P\,[f_a = +1\ ,\ f_d = -1] + C_m\,P\,[f_a = -1\ ,\ f_d = +1]\ .$$

Writing the joint probabilities above as

$$P\,[f_a = +1\ ,\ f_d = -1] = P\,[f_a = +1 \mid f_d = -1]\,P\,[f_d = -1]$$

and

$$P\,[f_a = -1\ ,\ f_d = +1] = P\,[f_a = -1 \mid f_d = +1]\,P\,[f_d = +1]\ ,$$

and defining

$$P = P\,[f_d = +1]$$

and

$$Q = P\,[f_d = -1]\ ,$$

then R is given by

$$R = QC_f\,P\,[f_a = +1 \mid f_d = -1] + PC_m\,P\,[f_a = -1 \mid f_d = +1] \qquad (2.3.1)$$

where

$$P + Q = 1 \, .$$

Notice that it is assumed that those $\bar{z}_i$'s in S^d cannot occur. The conditional probabilities in Equation 2.3.1 can be written in terms of the decision-maker interconnection channel parameters as

$$P\,[f_a = +1 \mid f_d = -1] = \alpha_1 \sum_{j=1}^{2^n} P\,[\bar{x}_j \in S_f^+ \mid f_d = -1] +$$

$$\beta_o \sum_{j=1}^{2^n} P\,[\bar{x}_j \in S_f^- \mid f_d = -1] \, , \qquad (2.3.2)$$

and

$$P\,[f_a = -1 \mid f_d = +1] = \alpha_o \sum_{j=1}^{2^n} P\,[\bar{x}_j \in S_f^- \mid f_d = +1] +$$

$$\beta_1 \sum_{j=1}^{2^n} P\,[\bar{x}_j \in S_f^+ \mid f_d = +1] \, . \qquad (2.3.3)$$

In order to simplify the risk expression define

$$A_o = \sum_{j=1}^{2^n} P\,[\bar{x}_j \in S_f^- \mid f_d = -1] \, ,$$

$$B_o = \sum_{j=1}^{2^n} P\,[\bar{x}_j \in S_f^+ \mid f_d = -1] \, , \qquad (2.3.4)$$

$$A_1 = \sum_{j=1}^{2^n} P\,[\bar{x}_j \in S_f^+ \mid f_d = +1] \, ,$$

and

$$B_1 = \sum_{j=1}^{2^n} P\left[\overline{x}_j \in S_f^- \mid f_d = +1\right] \qquad (2.3.5)$$

where

$$A_o + B_o = 1$$

and

$$A_1 + B_1 = 1 \ .$$

Notice that Equations 2.3.2 and 2.3.3 simplify to

$$P\left[f_a = +1 \mid f_d = -1\right] = \alpha_1 B_o + \beta_o A_o$$

and

$$P\left[f_a = -1 \mid f_d = +1\right] = \alpha_o B_1 + \beta_1 A_1$$

which can be visualized from Figure 2.3.3. The risk is simplified to

$$R = QC_f\left(\alpha_1 B_o + \beta_o A_o\right) + PC_m\left(\alpha_o B_1 + \beta_1 A_1\right)$$

or to

$$R = QC_f B_o\left(\alpha_1 + \alpha_o - 1\right) + PC_m B_1\left(\alpha_1 + \alpha_o - 1\right) +$$

$$QC_f\left(1 - \alpha_o\right) + PC_m\left(1 - \alpha_1\right) \ . \qquad (2.3.6)$$

The parameters which vary, depending upon which of the two decision-makers is used, are B_o, B_1, α_o, and α_1. The optimization of either decision-maker is done with respect to parameters contained within B_o and B_1. Equations 2.3.4 and 2.3.5 can be written

$$B_o = \sum_{j=1}^{2^n} \sum_{i=1}^{2^n} P\left[\overline{x}_j \in S_f^+ \mid \overline{x}_j\right] P\left[\overline{x}_j \mid \overline{z}_i\right] P\left[\overline{z}_i \mid f_d = -1\right]$$

$$B_1 = \sum_{j=1}^{2^n} \sum_{i=1}^{2^n} P\,[x_j \in S_f^- \mid \bar{x}_j]\, P\,[\bar{x}_j \mid \bar{z}_i]\, P\,[\bar{z}_i \mid f_d = 1]$$

or as

$$B_o = \sum_{j=1}^{2^n} \delta_j^+ \sum_{\bar{z}_i \in S_{f_d}^-} P\,[\bar{x}_j \mid \bar{z}_i]\, P\,[\bar{z}_i \mid \bar{z}_i \in S_{f_d}^-] \qquad (2.3.7)$$

$$B_1 = \sum_{j=1}^{2^n} \delta_j^+ \sum_{\bar{z}_i \in S_{f_d}^+} P\,[\bar{x}_j \mid \bar{z}_i]\, P\,[\bar{z}_i \mid \bar{z}_i \in S_{f_d}^+] \qquad (2.3.8)$$

where δ_j^- and δ_j^+ are defined by

$$\delta_j^- = P\,[\bar{x}_j \in S_f^- \mid \bar{x}_j] \qquad (2.3.9)$$

and

$$\delta_j^+ = P\,[\bar{x}_j \in S_f^+ \mid \bar{x}_j] \qquad (2.3.10)$$

and where $P\,[\bar{z}_i \mid f_d = -1]$ and $P\,[\bar{z}_i \mid f_d = +1]$ are given by Equations 2.2.1 and 2.2.2. The events $\bar{x}_j \in S_f^-$ and $\bar{x}_j \in S_f^+$ are deterministic since they are the specific events which the decision-maker is designed to perform. Therefore, δ_j^+ and δ_j^- are either 0 or 1 where $\delta_j^+ + \delta_j^- = 1$.

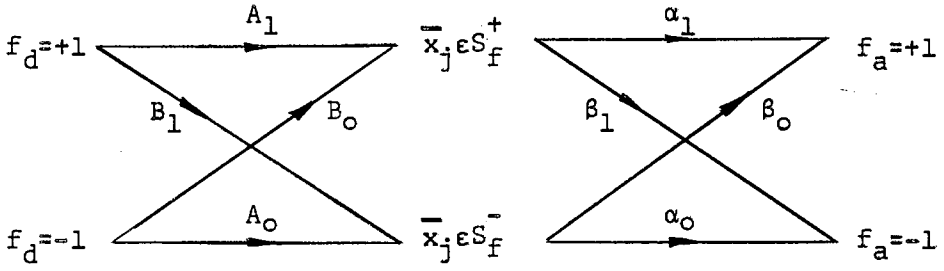


Figure 2.3.3. Simplified System Model

Combining Equations 2.3.6 through 2.3.10, the general risk function is given by

$$R = QC_f(\alpha_1 + \alpha_o - 1)\left\{\sum_{j=1}^{2^n} \delta_j^+ \sum_{\bar{z}_i \epsilon S_{f_d}^-} P[\bar{x}_j \mid \bar{z}_i] P[\bar{z}_i \mid z_i \epsilon S_{f_d}^-]\right\} +$$

$$PC_m(\alpha_1 + \alpha_o - 1)\left\{\sum_{j=1}^{2^n} \delta_j^- \sum_{\bar{z}_i \epsilon S_{f_d}^+} P[\bar{x}_j \mid \bar{z}_i] P[\bar{z}_i \mid \bar{z}_i \epsilon S_{f_d}^+]\right\} +$$

$$QC_f(1 - \alpha_o) + PC_m(1 - \alpha_1) .$$

Let

$$K_o = QC_f(\alpha_1 + \alpha_o - 1) ,$$

$$K_1 = PC_m(\alpha_1 + \alpha_o - 1) ,$$

and

$$K_2 = QC_f(1 - \alpha_o) + PC_m(1 - \alpha_1) ;$$

then

$$R = \sum_{j=1}^{2^n} \{\delta_j^+ K_o \sum_{\bar{z}_i \epsilon S_{f_d}^-} P[\bar{x}_j \mid \bar{z}_i] P[\bar{z}_i \mid \bar{z}_i \epsilon S_{f_d}^-] +$$

$$\delta_j^- K_1 \sum_{\bar{z}_i \epsilon S_{f_d}^+} P[\bar{x}_j \mid \bar{z}_i] P[\bar{z}_i \mid \bar{z}_i \epsilon S_{f_d}^+]\} + K_2 . \qquad (2.3.11)$$

This is the general risk function for both of the assumed channel models and for the assumed loss function. In order to minimize R,

$\delta_j^- = 1 - \delta_j^+$ is given by

$$\delta_j^- = \begin{cases} 1 \text{ if } K_3^{(j)} < 0 \\ 0 \text{ if } K_3^{(j)} > 0 \\ \text{arbitrary if } K_3^{(j)} = 0, \; j = 1, 2, \ldots, 2^n \end{cases},$$

where

$$K_3^{(j)} = K_o \sum_{\bar{z}_i \varepsilon S_{f_d}^-} P[\bar{x}_j \mid \bar{z}_i] \, P[\bar{z}_i \mid \bar{z}_i \varepsilon S_{f_d}^-] -$$

$$K_1 \sum_{\bar{z}_i \varepsilon S_{f_d}^+} P[\bar{x}_j \mid \bar{z}_i] \, P[\bar{z}_i \mid \bar{z}_i \varepsilon S_{f_d}^+] .$$

2.4 Development of the Risk Function for the Almost Empty Pattern Set Case. In Section 2.2 an example is cited of a system which has an almost empty pattern set. In this section this particular type of system is more fully described, and its optimality is considered. The mathematics is kept general so that the systems in either Figure 2.2.1 or Figure 2.2.2 can be used.

In order to simplify the algebra, it is assumed that each of the sensor and switch channels in Figure 2.2.1 and each of the sensor channels in Figure 2.2.2 can be represented by a composite channel as in Figure 2.4.1. The channel parameters are given by

$$p_1 = P[x_{ik} = +1 \mid z_{ik} = +1] ,$$

$$q_1 = P[x_{ik} = -1 \mid z_{ik} = +1] ,$$

$$p_o = P[x_{ik} = -1 \mid z_{ik} = -1] ,$$

and

$$q_o = P\,[x_{ik} = 1 \mid z_{ik} = -1]$$

where x_{ik} and z_{ik} are components of $\overline{x}_i$ and $\overline{z}_i$, respectively. It is assumed that the channels are identical and statistically independent.

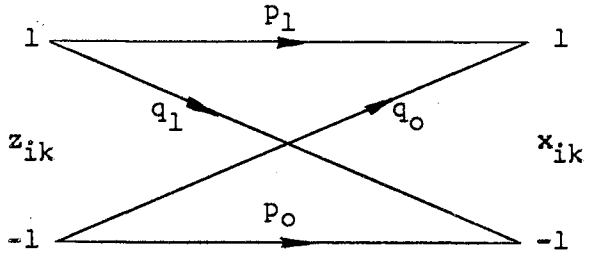


Figure 2.4.1. The ith Composite Channel

Referring to Equations 2.2.1 and 2.2.2, it is assumed that

$$P\,[\overline{z}_i \mid \overline{z}_i \in S^{+}_{f_d}] = \frac{1}{m^{+}} = 1 \qquad (2.4.1)$$

and

$$P\,[\overline{z}_i \mid \overline{z}_i \in S^{-}_{f_d}] = \frac{1}{m^{-}} = 1 \qquad (2.4.2)$$

where $m^{+} = 1$ and $m^{-} = 1$ for the almost empty pattern set case. As pointed out in Section 2.2

$$S^{+}_{f_d} = \{\overline{z}_{2^n}\} = \{(1, 1, \ldots, 1)\}$$

and

$$S^{-}_{f_d} = \{\overline{z}_1\} = \{(-1, -1, \ldots, -1)\} \quad .$$

Using the fact that there are only two states of nature and using Equations 2.4.1 and 2.4.2, Equation 2.3.11 can be simplified to

$$R = \sum_{\text{all } j} \{\delta_j^+ K_o P[\bar{x}_j \mid \bar{z}_1] + \delta_j^- K_1 P[\bar{x}_j \mid \bar{z}_{2n}]\} + K_2 \quad .$$

In order to evaluate the probabilities $P[\bar{x}_j \mid \bar{z}_i]$, $i = 1, 2, \ldots, 2^n$, an investigation must be made of the channels shown in Figure 2.4.1. Recalling that the n channels are identical and statistically independent, the probability of the occurrence of the jth vector $\bar{x}_j$, given that $\bar{z}_1 = (-1, -1, \ldots, -1)$ is sent, is

$$P[\bar{x}_j \mid \bar{z}_1] = p_o^{n-H_{1j}} q_o^{H_{1j}} \tag{2.4.3}$$

where H_{ij} is the Hamming distance between $\bar{x}_j$ and $\bar{z}_i$. Similarly,

$$P[\bar{x}_j \mid \bar{z}_{2n}] = p_1^{n-H_{2n_j}} q_1^{H_{2n_j}} \quad . \tag{2.4.4}$$

Since $H_{2n_j} = n - H_{1j}$,

$$P[\bar{x}_j \mid \bar{z}_{2n}] = p_1^{H_{1j}} q_1^{n-H_{1j}} \quad ;$$

thus

$$R = \sum_{\text{all } j} [\delta_j^+ K_o p_o^{n-H_{1j}} q_o^{H_{1j}} + \delta_j^- K_1 p_1^{H_{1j}} q_1^{n-H_{1j}}] + K_2 \quad .$$

Using $\delta_j^+ = 1 - \delta_j^-$, R can be written as

$$R = \sum_{\text{all } j} \delta_j^- [K_1 p_1^{H_{1j}} q_1^{n-H_{1j}} - K_o p_o^{n-H_{1j}} q_o^{H_{1j}}] + \sum_{\text{all } j} K_o p_o^{n-H_{1j}} q_o^{H_{1j}} + K_2 \quad . \tag{2.4.5}$$

Corresponding to each value of $H_{1j} = 0, 1, \ldots, n$ there are $\binom{n}{H_{1j}}$ values of j. If the values δ_j for these values of j are added to produce a term $k_{H_{1j}}$, $0 \leq k_{H_{1j}} \leq \binom{n}{H_{1j}}$, and if like terms in Equation 2.4.5 are collected then

$$R = \sum_{i=0}^{n} k_i [K_1 p_1^i q_1^{n-i} - K_o p_o^{n-i} q_o^i] + K_o \sum_{i=0}^{n} \binom{n}{i} p_o^{n-i} q_o^i + K_2$$

or

$$R = \sum_{i=0}^{n} k_i [K_1 p_1^i q_1^{n-i} - K_o p_o^{n-i} q_o^i] + K_o + K_2 \quad . \tag{2.4.6}$$

The integer k_i is the number of vectors $\bar{x}_j$ having i components equal to 1 and which are placed in the set S_f by the decision-maker. In other words k_i is the number of vectors $\bar{x}_j$ within a Hamming distance i of $\bar{z}_1 = (-1, -1, \ldots, -1)$ and which are placed in the set S_f by the decision-maker.

The optimum values of k_i are given by

$$k_i = \begin{cases} \binom{n}{i} \text{ if } K_4^{(i)} < 0 \\ 0 \text{ if } K_4^{(i)} > 0 \\ \text{arbitrary if } K_4^{(i)} = 0, \end{cases} \quad i = 1, 2, \ldots, n, \tag{2.4.7}$$

where

$$K_4^{(i)} = K_1 p_1^i q_1^{n-i} - K_o p_o^{n-i} q_o^i \quad .$$

The following example illustrates the optimality of a common decision maker.

Example 2.4.1. Letting $n = 4$ the common, redundant, relay contact

network in Figure 2.2.2 results. For convenience it is shown again in Figure 2.4.2 with labeling which corresponds to that in Figure 2.3.1.

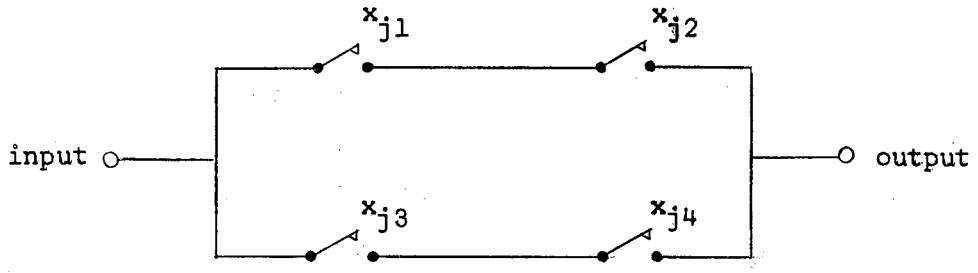


Figure 2.4.2. The "Quad" Switching Circuit

The component x_{jk} of $\bar{x}_j = (x_{j1}, x_{j2}, x_{j3}, x_{j4})$ is 1 if the kth switch is closed and 0 if it is open. The actual classification $f_a(\bar{x}_j)$ of a vector $\bar{x}_j$ is 1 if a path exists from the input of the quad to the output. Table 2.4.1 lists the components of the vectors $\bar{x}_j$ and the corresponding designed classifications $f(\bar{x}_j)$. The important information in this table is the classification of the sets of $\binom{4}{i}$ vectors with i components equal to one. Table 2.4.2 presents this condensed information. Notice that for i = 2, some of the vectors with two components equal to 1 are in S_f^- and some are in S_f^+. However, Equation 2.4.7 does not correspond to this distribution except in the case for $K_3^{(2)} = 0$; that is, the quad is not optimum except when

$$K_1 p_1^2 q_1^2 = K_o p_o^2 q_o^2 \quad .$$

Substituting in the original parameters, the condition for optimality of a quad is

TABLE 2.4.1

CLASSIFICATION OF VECTORS BY A QUAD

j	x_{j1}	x_{j2}	x_{j3}	x_{j4}	$f(\bar{x}_j)$
1	-1	-1	-1	-1	-1
2	-1	-1	-1	1	-1
3	-1	-1	1	-1	-1
4	-1	-1	1	1	1
5	-1	1	-1	-1	-1
6	-1	1	-1	1	-1
7	-1	1	1	-1	-1
8	-1	1	1	1	1
9	1	-1	-1	-1	-1
10	1	-1	-1	1	-1
11	1	-1	1	-1	-1
12	1	-1	1	1	1
13	1	1	-1	-1	1
14	1	1	-1	1	1
15	1	1	1	-1	1
16	1	1	1	1	1

TABLE 2.4.2

CLASSIFICATION OF SETS OF VECTORS BY A QUAD

Number of 1 Components	Location of Vectors $\bar{x}_j$ $j = 1, \ldots, 16$	
i	S_f^-	S_f^+
0	1	---
1	2, 3, 5, 9	---
2	6, 7, 10, 11	4, 13
3	---	8, 12, 14, 15
4	---	16

$$\sqrt{PC_m}\, p_1 q_1 = \sqrt{QC_f}\, p_o q_o \quad . \tag{2.4.8}$$

If the quad is drawn as in Figure 2.4.3, a similar situation arises. This configuration is also suboptimal except under the conditions of Equation 2.4.9. The only difference is in the location of the vectors $\bar{x}_j$ for i = 2. In this case S_f^- contains two vectors for which i = 2 and S_f^+ contains four.

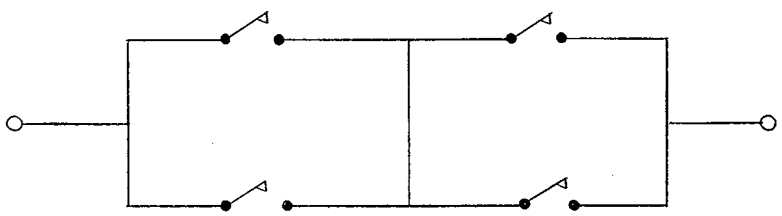


Figure 2.4.3. Alternate Quad Configuration

2.5 Comparison of R_S to R_T for an Almost Empty Pattern Set. In the previous section the conditions for optimality of decision-makers, either with or without switching elements on each input, is derived for the almost empty pattern set case. In order for a designer to decide which type of decision-maker to use, he needs a comparison based on the significant parameters associated with the risk functions R_S and R_T corresponding to the two types of decision-makers. The significant parameters in R_S are assumed to be p_{ro}, q_{ro}, p_{r1}, q_{r1}, α_o^S, β_o^S, α_1^S, and β_1^S while those of R_T are α_o^T, β_o^T, α_1^T, and β_1^T. The remaining parameters P, Q, p_{so}, q_{so}, p_{s1}, q_{s1}, C_m, C_f, and n in R are assumed to be the same for either R_S or R_T.

From Figure 2.5.1 the probabilities p_o^S, q_o^S, p_1^S, and q_1^S in R_S given by Equation 2.4.6 can be written as

$$\begin{aligned} p_o^S &= p_{so}p_{ro} + q_{so}q_{r1} \\ q_o^S &= p_{so}q_{ro} + q_{so}p_{r1} \\ p_1^S &= p_{s1}p_{r1} + q_{s1}q_{ro} \\ q_1^S &= p_{s1}q_{r1} + q_{s1}p_{ro} \end{aligned} \qquad (2.5.1)$$

where the superscripts correspond to the subscript on R_S. Similarly from Figure 2.5.2 the corresponding parameters in R_T are

$$\begin{aligned} p_o^T &= p_{so} & p_1^T &= p_{s1} \\ q_o^T &= q_{so} & q_1^T &= q_{s1} \end{aligned} \quad . \qquad (2.5.2)$$

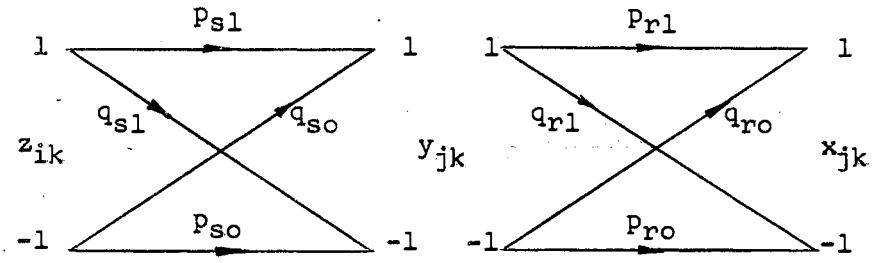
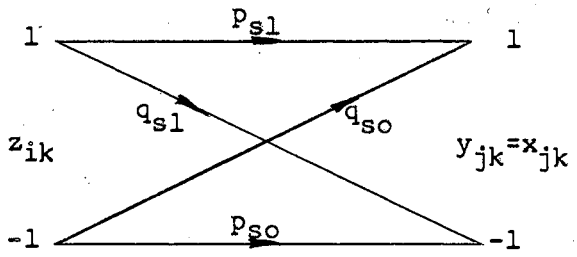


Figure 2.5.1. The ith Composite Channel for Decision-Maker With Switching Elements



$i = 1, 2^n$

$j = 1, 2, \ldots, 2^n$

$k = 1, 2, \ldots, n$

Figure 2.5.2. The ith Composite Channel for Decision-Maker Without Switching Elements

The two risk functions can now be written. R_S is given by

$$R_S = \sum_{i=0}^{n} \left\{ k_i^s [K_1^S (p_{s1}p_{r1}+q_{s1}q_{ro})^i (p_{s1}q_{r1}+q_{s1}p_{ro})^{n-i} + \right.$$
$$\left. - K_o^S (p_{so}p_{ro}+q_{so}q_{r1})^{n-i} (p_{so}q_{ro}+q_{so}p_{r1})^i] \right\} + K_o^S + K_2^S \qquad (2.5.3)$$

where the optimum k_i^s is

$$k_i^s = \begin{cases} \binom{n}{i} \text{ if } K_s^{(i)} < 0 \\ 0 \text{ if } K_s^{(i)} > 0 \\ \text{arbitrary if } K_s^{(i)} = 0, \; i = 1, 2, \ldots, n, \end{cases}$$

where

$$K_s^{(i)} = K_1^S(p_{s1}p_{r1}+q_{s1}q_{ro})^i(p_{s1}q_{r1}+q_{s1}p_{ro})^{n-i} +$$

$$- K_o^S(p_{so}p_{ro}+q_{so}q_{r1})^{n-i}(p_{so}q_{ro}+q_{so}p_{r1})^i ,$$

$$K_o^S = QC_f(\alpha_1^S+\alpha_o^S-1),$$

$$K_1^S = PC_m(\alpha_1^S+\alpha_o^S-1),$$

and

$$K_2^S = QC_f(1-\alpha_o^S) + PC_m(1-\alpha_1^S) .$$

R_T is given by

$$R_T = \sum_{i=0}^{n} \left\{ k_i^T[K_1^T p_{s1}^i q_{s1}^{n-i} - K_o^T p_{so}^{n-i} q_{so}^i] \right\} + K_o^T + K_2^T \qquad (2.5.4)$$

where the optimum value of k_i^T is

$$k_i^T = \begin{cases} \binom{n}{i} \text{ if } K_T^{(i)} < 0 \\ 0 \text{ if } K_T^{(i)} > 0 \\ \text{arbitrary if } K_T^{(i)} = 0 \end{cases}$$

where

$$K_T^{(i)} = K_1^T p_{s1}^i q_{s1}^{n-i} - K_o^T p_{so}^{n-i} q_{so}^i ,$$

$$K_o^T = QC_f(\alpha_1^T+\alpha_o^T-1),$$

$$K_1^T = PC_m(\alpha_1^T+\alpha_o^T-1),$$

and

$$K_2^T = QC_f(1-\alpha_o^T) + PC_m(1-\alpha_1^T).$$

It turns out that Equations 2.5.3 and 2.5.4 converge rapidly with n to functions which are independent for some range of values of the sensor and switching element channel parameters. Because of this behavior the comparison of the decision-makers is broken into two parts as indicated by the two regions in Figure 2.5.3. The risk R as given by Equation 2.4.6 with $\bar{k}_i$ optimum and with the following constant parameters is plotted with n as the independent variable:

$p_o = 0.995$ $\quad$ $P = 0.5$

$q_o = 0.005$ $\quad$ $Q = 0.5$

$p_1 = 0.995$ $\quad$ $C_m = 10.0$

$q_1 = 0.005$ $\quad$ $C_f = 1.0$.

As shown in the figure the values of n are divided into two regions; the first part of this comparison is concerned with Region I while the second part is restricted to Region II.

For values of n in Region I a comparison between R_S and R_T is made by making the assumption that the decision-maker interconnection channel corresponding to R_S is perfect; that is, $\alpha_o^S = \alpha_1^S = 1$ and $\beta_o^S = \beta_1^S = 0$ in Figure 2.3.1. Therefore the parameter $p_{ro} = p_{r1}$ forms a family of curves for R_S as a function of n while $\alpha_o^T = \alpha_1^T$ forms a family for R_T as a function of n. These curves are shown in Figure 2.5.4 for the following constant parameter values:

$p_{so} = 0.99$ $\quad$ $P = 0.5$

$q_{so} = 0.01$ $\quad$ $Q = 0.5$

$p_{s1} = 0.99$ $\quad$ $C_m = 10.0$

$q_{s1} = 0.01$ $\quad$ $C_f = 1.0$

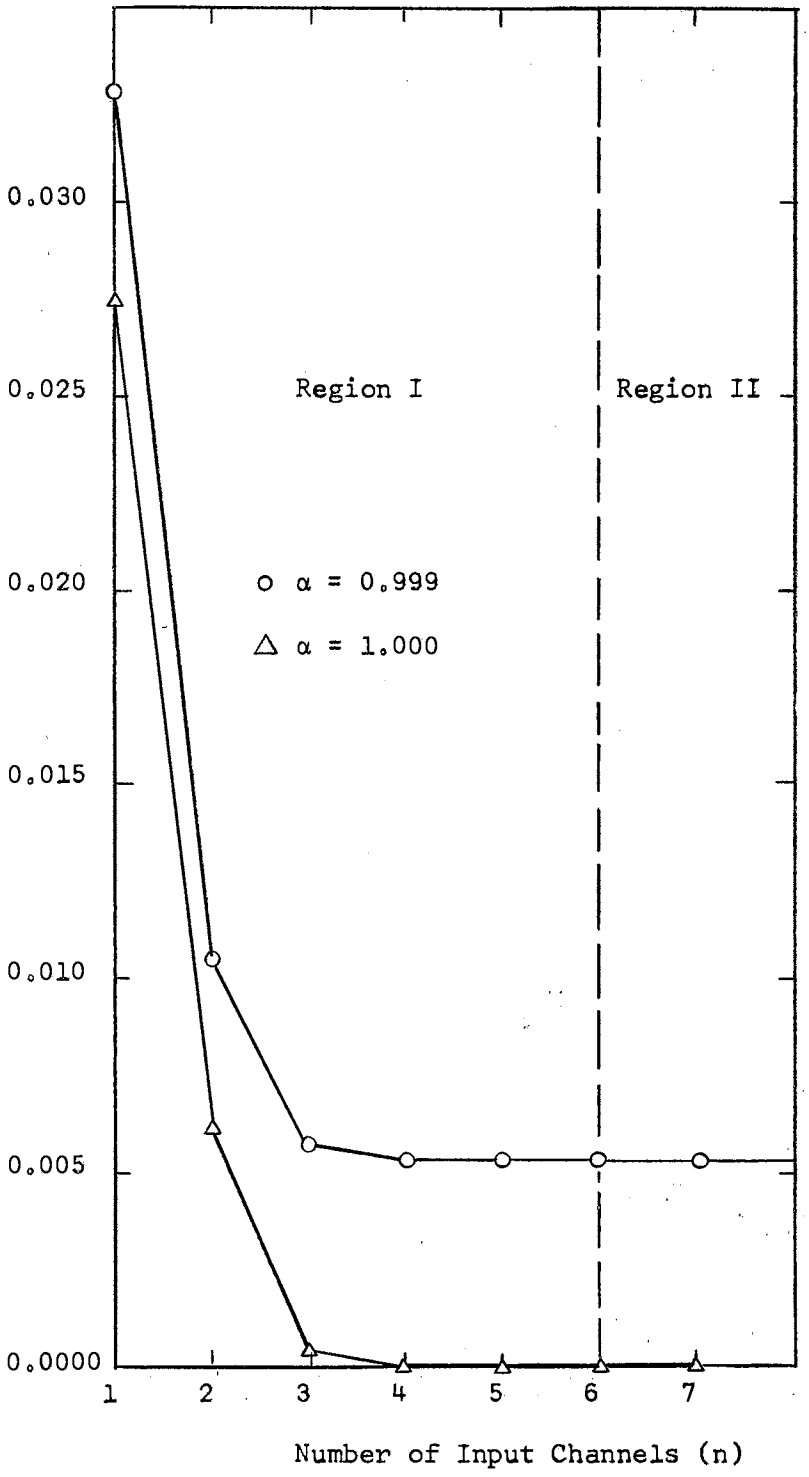


Figure 2.5.3. Variation of R w.r.t. n for the Almost Empty Pattern Set Case

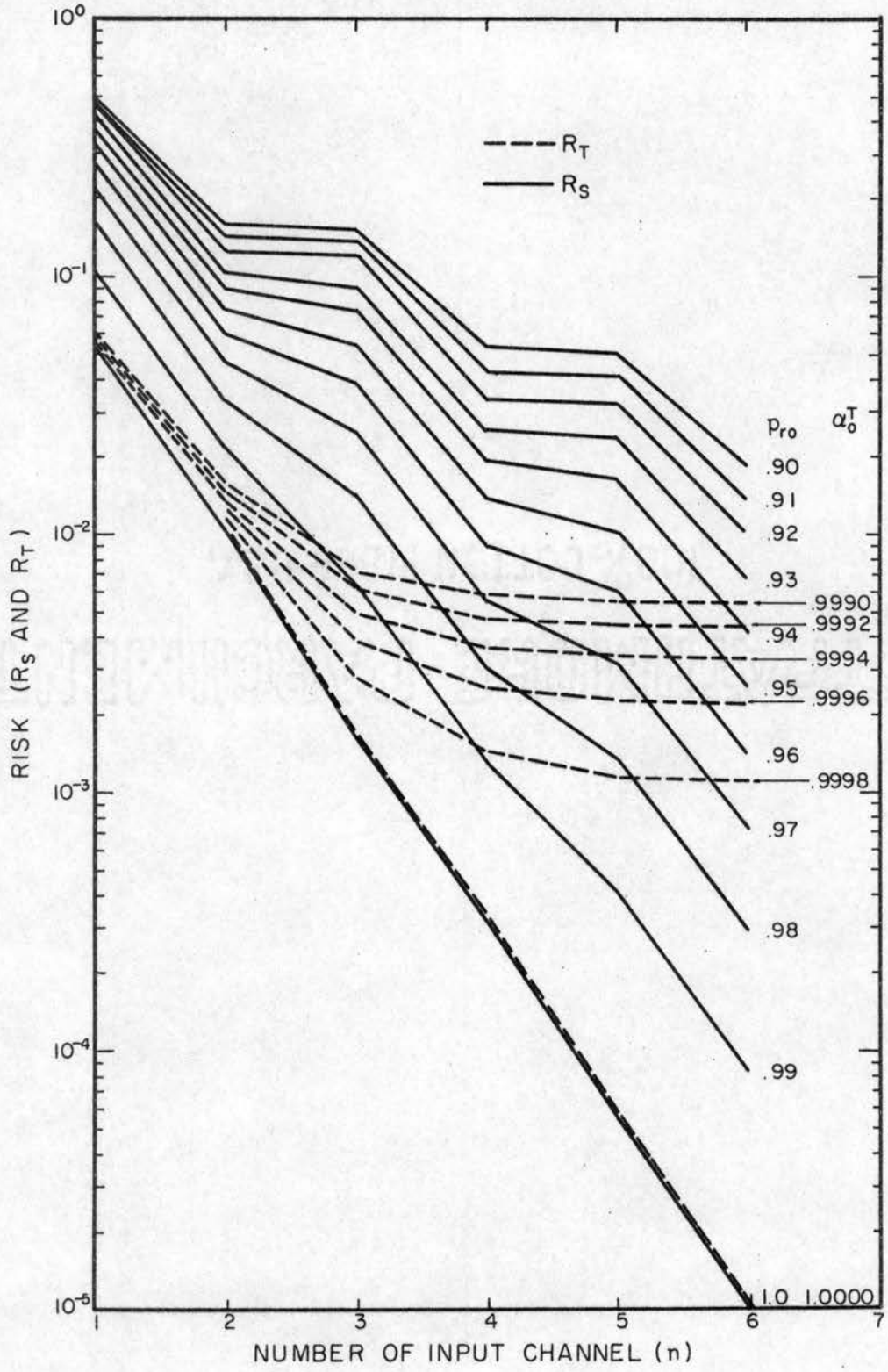


Figure 2.5.4. Risk Curves for an Almost Empty Pattern Set

For a decision-maker system with these parameter values, Figure 2.5.4 can be used to determine when either of the two decision-maker types is justified in terms of the risk involved. For example, if n = 4 and if a relay contact network is available for which $p_{ro} = p_{r1} = 0.97$, then $\alpha_o^T = \alpha_1^T$ must be greater than 0.999. This means that a lower bound is set for the interconnection channel parameters for a decision-maker of the type shown in Figure 2.3.2 to be better than a relay contact network modeled in Figure 2.3.1.

An important observation from Figure 2.5.4 is that R_S approaches zero more rapidly as p_{ro} increases. On the other hand the parameter α_o^T merely changes the asymptote for R_T as n increases. Thus the convergence demonstrated by Figure 2.5.3 is demonstrated further in Figure 2.5.4.

A more careful examination of the risks R_S and R_T for the particular case of n = 3 is presented in Figure 2.5.5. The same constant parameter values used for Figure 2.5.4 are used here. These curves are representative of the risk variations for other values of n; however, the variations with p_{ro} and α_o^T become greater as n increases as shown in Figure 2.5.4.

If n is restricted to Region II of Figure 2.5.3, some approximations can be made for values of p_{so}, p_{s1}, p_{ro}, and p_{r1} close to one and for PC_m and QC_f approximately equal. The second part of this comparison is concerned with this case, where the effects of variations in the decision maker interconnection channel parameters are more clearly demonstrated.

Consider the general risk R from Equation 2.4.6 which is

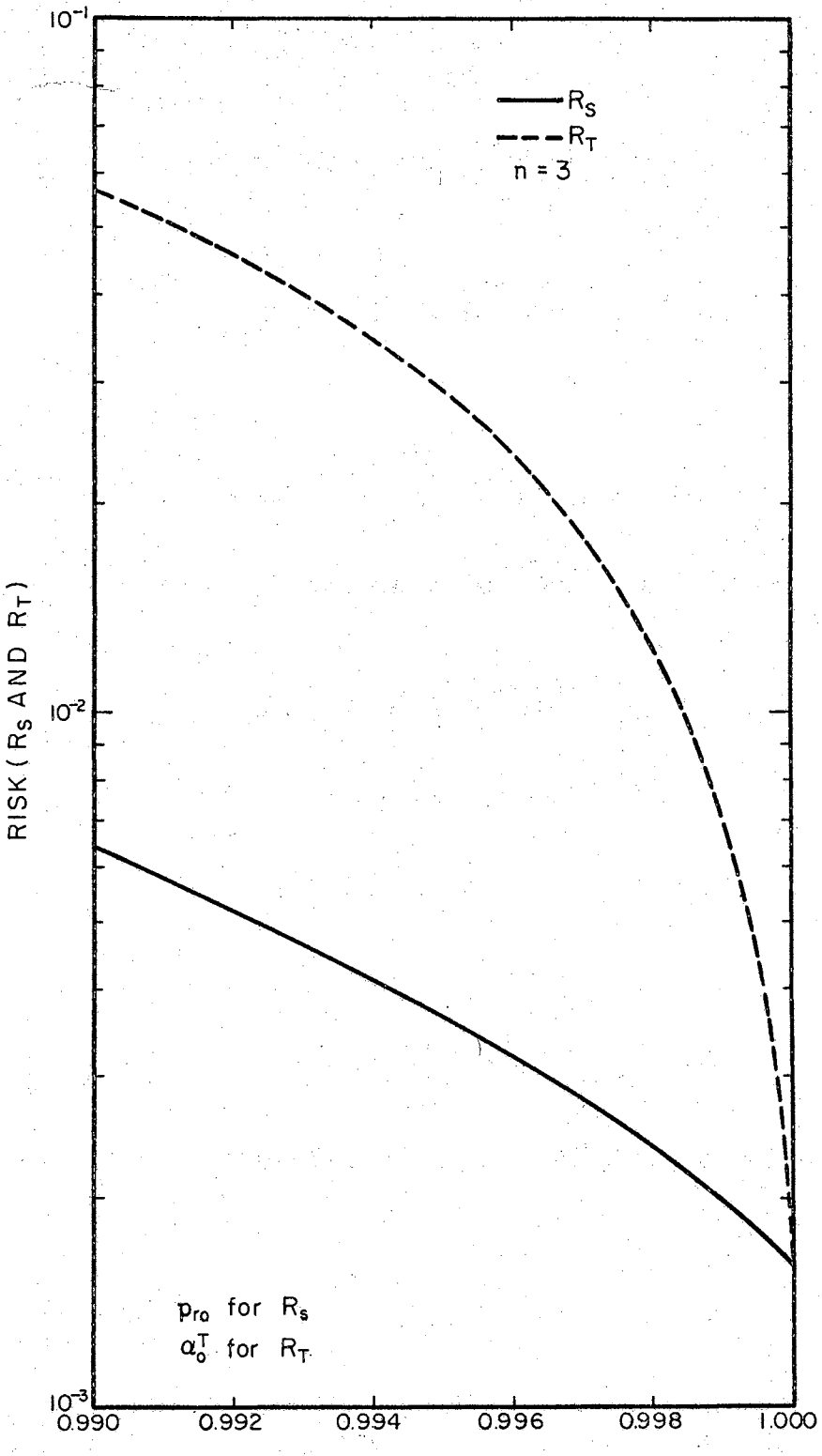


Figure 2.5.5. Risk Curves for an Almost Empty Pattern Set With n=3

$$R = \sum_{i=0}^{n} k_i[K_1 p_1^{-i} q_1^{n-i} - K_o p_o^{n-i} q_o^{i}] + K_o + K_2 \quad .$$

This function can be simplified by an examination of the binomial distribution

$$F(\tfrac{n}{2}) = \int_{-\infty}^{\frac{n}{2}} f(x;n,q)dx$$

where

$$f(x;n,q) = \sum_{i=0}^{n} \binom{n}{i} p^{n-i} q^{i} \delta(x-i)$$

is its density and $\delta(x-i)$ is the delta function. Notice that

$$F(\tfrac{n}{2}) = \sum_{i=0}^{\frac{n}{2}} \binom{n}{i} p^{n-i} q^{i} \quad . \tag{2.5.5}$$

The expected value and variance for this density are

$$E[x] = nq$$

and

$$E[(x-nq)^2] = npq \ ,$$

respectively. Papoulis (18) states on page 163 that the following inequalities are true:

$$E^2[\tfrac{n}{2} - X \mid x \leq a] \leq E[(a - X)^2 \mid x \leq a] \tag{2.5.6}$$

$$E[a - X \mid x \leq a]F(a) \geq a - E[X] \tag{2.5.7}$$

$$E[(a - X)^2 \mid x \leq a]F(a) \leq E[(X - nq)^2] + (a - E[X])^2. \tag{2.5.8}$$

The constant "a" is in general arbitrary. Letting a = n/2 and

substituting in the mean and variance shown above, it can be shown that

$$F(\tfrac{n}{2}) \geq \frac{(\frac{n}{2} - nq)^2}{npq + (\frac{n}{2} - nq)^2}$$

by properly manipulating Inequalities 2.5.6, 2.5.7, and 2.5.8. As n approaches infinity the result is

$$\lim_{n\to\infty} F(\tfrac{n}{2}) \geq \lim_{n\to\infty} \frac{(\frac{1}{2} - q)^2}{\frac{pq}{n} + (\frac{1}{2} - q)^2} = 1 \quad .$$

However, since $F(a) \leq 1$ for all values of the variable "a", the binomial distribution in Equation 2.5.5 for large n is

$$\lim_{n\to\infty} F(\tfrac{n}{2}) = 1 \quad .$$

From an examination of the binomial distribution tables by Weintraub (24) it can be seen that this convergence is very rapid for small q. For example, for q = 0.01 and n = 6, $F(\frac{6}{2})$ differs from 1 by 0.0000195536 while n = 12 yields a difference of approximately 9.0×10^{-10}.

The result of this convergence is that the summations

$$\sum_{i=\frac{n}{2}}^{n} \binom{n}{i} p_o^{n-i} q_o^i$$

and

$$\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i} p_1^i q_1^{n-i}$$

approach zero as n gets large for small values of q_o and q_1. Therefore, using the definition of the optimum $\bar{k}_i$ and including in the summations

in R the terms above which approximately sum to zero, R can be written as

$$R \simeq (\alpha_1+\alpha_o-1) \sum_{i=0}^{n} (0)PC_m p_1^i q_1^{n-i} - \sum_{i=0}^{n} \binom{n}{i} QC_f p_o^{n-i} q_o^i$$

$$+ QC_f(\alpha_1+\alpha_o-1) + QC_f(1-\alpha_o) + PC_m(1-\alpha_1) \ .$$

This equation simplifies to

$$R \simeq QC_f(1-\alpha_o) + PC_m(1-\alpha_1) \ .$$

Therefore, with the appropriate approximations, the risks are

$$R_S \simeq QC_f(1-\alpha_o^S) + PC_m(1-\alpha_1^S)$$

and

$$R_T \simeq QC_f(1-\alpha_o^T) + PC_m(1-\alpha_1^T)$$

for large values of n. If the interconnection channels are assumed to be symmetric for both types of decision-makers, the risks simplify to

$$R_S \simeq (1-\alpha^S)(QC_f + PC_m) \qquad (2.5.9)$$

and

$$R_T \simeq (1-\alpha^T)(QC_f + PC_m) \qquad (2.5.10)$$

where $\alpha^S = \alpha_o^S = \alpha_1^S$ and $\alpha^T = \alpha_o^T = \alpha_1^T$. A significant observation here is that as the number of redundant channels increases for either type of decision-maker the risk R (R_S or R_T) approaches zero.

Certain conclusions can be drawn from the comparison of the two types of decision-makers which are used in a system with an almost empty

pattern set. These conclusions are drawn with the thought in mind that a decision-maker which has no switching elements on each input (Figure 2.3.2) has a more complicated interconnection channel than a decision-maker without these switching elements (Figure 2.3.1). This means that α_o^T and α_1^T are smaller than α_1^S and α_o^S and that in the comparison of the two types of decision-makers this difference may be offset by the presence of input switching elements and by variations in n.

As shown by Figure 2.5.4 and Equations 2.5.5 and 2.5.6, the value of n determines the method of evaluation. For small n in Region I the choice of decision-makers can be made by a comparison of the risks R_S and R_T as in Figure 2.5.4. For n in Region II, Equations 2.5.9 and 2.5.10 indicate that a decision-maker with input switching elements (relay network) is superior to the other type of decision maker (threshold logic unit network).

2.6 Comparison of R_S to R_T for a Full Pattern Set. The case of a full pattern set does not permit the simplifications in the risk function, given by Equation 2.3.11, that were permitted in Sections 2.4 and 2.5 for the almost empty pattern set case. The case considered in this section permits all 2^n $\bar{z}_i$'s to occur in such a way that the states perform a majority rule. It is assumed that n is odd and that the $\bar{z}_i$'s with (n+1)/2 or more "+1" components are in the set $S_{f_d}^+$ while those with (n+1)/2 or more "-1" components are in the set $S_{f_d}^-$. It is also assumed that the probabilities given in Equations 2.2.1 and 2.2.2 are

$$P[\bar{z}_i \mid \bar{z}_i \epsilon S_{f_d}^+] = \frac{1}{2^{n-1}}$$

and

$$P[\bar{z}_i \mid \bar{z}_i \epsilon S^-_{f_d}] = \frac{1}{2^{n-1}} .$$

This means that, for any value of odd n, a consistent classification of the 2^n states of nature exists; therefore R_S and R_T can be evaluated for various values of n.

Besides n, the sensor and switch channel parameters and the decision-maker interconnection channel parameters are important. The intent here is to show where the trade-off comes with respect to n when increased decision-maker interconnection channel complexity (lower values of α_o^T and α_1^T in R_T) is contrasted with the reliability of switching elements on all n inputs (the inclusion of p_{ro}, q_{ro}, p_{r1}, and q_{r1}, in R_S).

The risks R_S and R_T are given by Equation 2.3.11 with the appropriate changes in parameters. For R_S these changes result in

$$R_S = \frac{2}{2^n} \sum_{j=1}^{2^n} \left\{ \delta_j^+ K_o^S \sum_{\bar{z}_i \epsilon S^-_{f_d}} \prod_{k=1}^{n} P[x_{jk} \mid z_{ik}] + \right.$$

$$\left. + \delta_j^- K_1^S \sum_{\bar{z}_i \epsilon S^+_{f_d}} \prod_{k=1}^{n} P[x_{jk} \mid z_{ik}] \right\} + K_2^S$$

where

$$P[x_{jk} = +1 \mid z_{ik} = +1] = p_1^S = p_{s1}p_{r1} + q_{s1}q_{ro}$$

$$P[x_{jk} = -1 \mid z_{ik} = +1] = q_1^S = p_{s1}q_{r1} + q_{s1}p_{ro}$$

$$P[x_{jk} = -1 \mid z_{ik} = -1] = p_o^S = p_{so}p_{ro} + q_{so}q_{r1}$$

$$P[x_{jk} = 1 \mid z_{ik} = -1] = q_o^S = p_{so}q_{ro} + q_{so}p_{r1}$$

as in Equations 2.5.1. Similarly

$$R_T = \frac{2}{2^n} \sum_{j=1}^{2^n} \left\{ \delta_j^+ K_o^T \sum_{\underline{z}_i \in S_{f_d}^-} \prod_{k=1}^{n} P[x_{jk} \mid z_{ik}] + \right.$$

$$\left. + \delta_j^- K_1^T \sum_{\underline{z}_i \in S_{f_d}^+} \prod_{k=1}^{n} P[x_{jk} \mid z_{ik}] \right\} + K_2^T$$

where

$$p_1^T = p_{s1}$$

$$q_1^T = q_{s1}$$

$$p_o^T = p_{so}$$

$$q_o^T = q_{so}$$

as in Equations 2.5.2. Using the risks R_S and R_T, a comparison similar to the one in Section 2.6 can be made.

Consider the particular case where $\alpha_o^S = \alpha_1^S = 1$, $\beta_o^S = \beta_1^S = 0$, $p_{ro} = p_{r1}$, and where $\delta_j^- = 1 - \delta_j^+$ is optimum for both R_S and R_T. Using the following parameter values

$p_{so} = 0.99$	$P = 0.5$
$q_{so} = 0.01$	$Q = 0.5$
$p_{s1} = 0.99$	$C_m = 10.0$
$q_{s1} = 0.01$	$C_f = 1.0$

the family of curves shown in Figure 2.6.1 results.

These curves illustrate the conclusions which can be drawn regarding the trade-offs between the two types of decision-makers considered here. For the larger values of p_{ro} and α_o^T the curves for R_S tend to become steeper than the lines for R_T as n increases. It is this feature which indicates that it becomes more critical that p_{ro} be larger than α_o^T

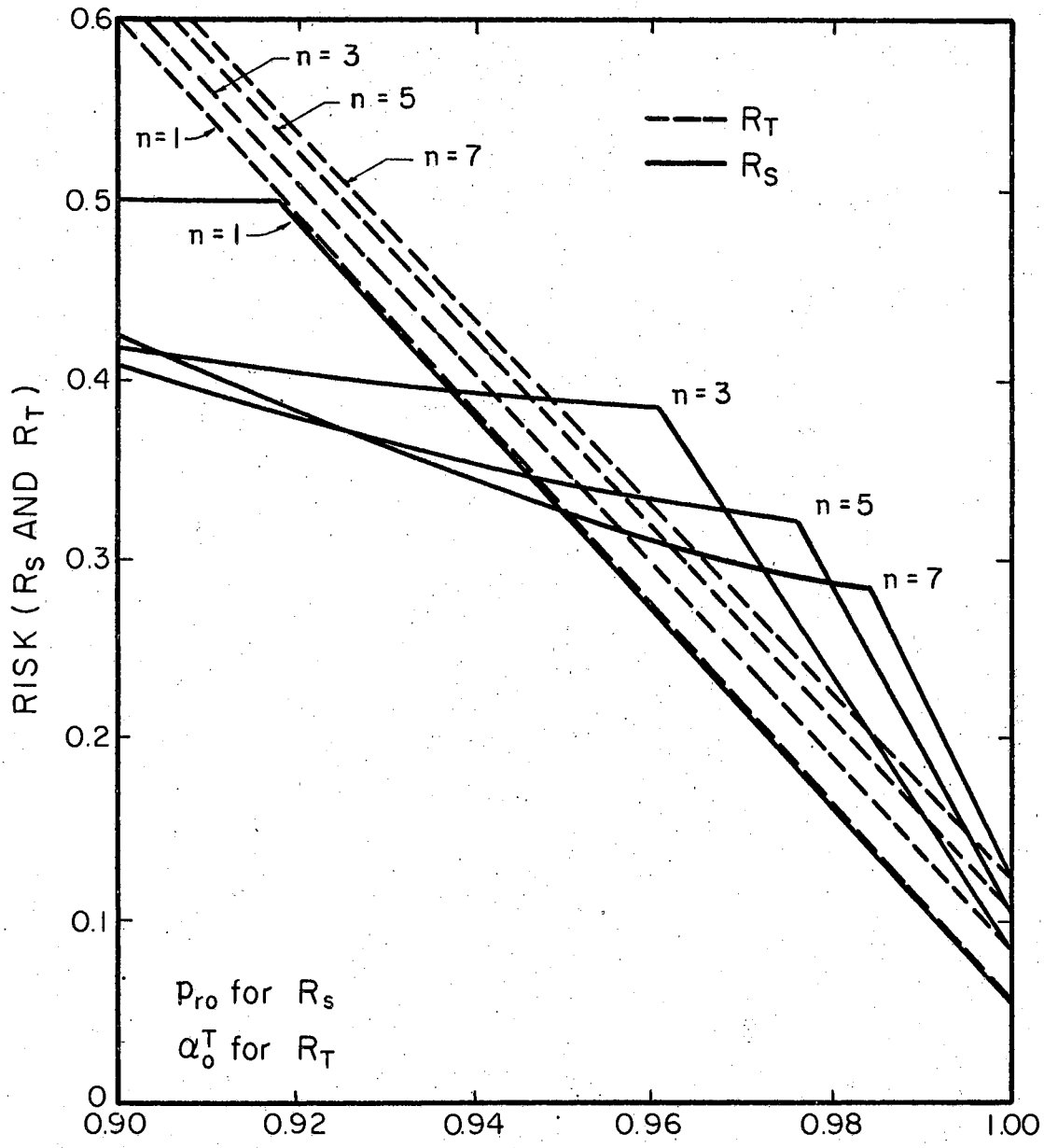


Figure 2.6.1. Risk Curves for a Full Pattern Set

for large values of n. This means that, for a full pattern set, a decision-maker with switching elements on every input can be improved upon by using a decision-maker such as a threshold logic unit network which has a more complicated interconnection channel ($\alpha_o^T < \alpha_o^S$). The improvement of this interconnection channel is the fundamental topic of the investigation whose results are presented in the remaining chapters and appendices.

One peculiarity of the curves for R_S shown in Figure 2.6.1 is the abrupt break in the curves. These breaks are caused by the reclassification of certain patterns $\overline{x}_i$ to permit minimization of R_S. In other words, as p_{ro} decreases, there results an increase in the probability that certain vectors $\overline{z}_i$ in one classification will be transformed into vectors $\overline{x}_i$ in the other classification. Therefore the decision-maker must reclassify the latter as dictated by the loss function. Notice that this reclassification can cause the patterns in $S_{f_d}^+$ and $S_{f_d}^-$ to be considerably different from those in S_f^+ and S_f^- respectively. Recall that the sets $S_{f_d}^+$ and $S_{f_d}^-$ are the classifications of nature and S_f^+ and S_f^- are the decision-maker's designed classification.

In summary the full pattern set case is a situation where a decrease in reliability of input switching elements justifies the use of a decision-maker such as a threshold logic network. The reason that this situation occurs with a full pattern set and not with the almost empty pattern set is that in the former some of the patterns in $S_{f_d}^+$ are within very small Hamming distances of patterns in $S_{f_d}^-$; thus it is more likely in this case that a state of nature in one classification will appear to the decision-maker to be in the opposite classification. However, it is this possible reclassification that forces the use of more versatile

networks of threshold logic units rather than single threshold logic units. The versatility of these units is considered indirectly in Chapter III.

2.7 Conclusion. The conclusions drawn from the investigation presented in this chapter are the justification for the work presented in the remaining chapters. The most important conclusion is that, if the interconnection channel of a threshold logic network can be made reasonably reliable, then these networks are better in terms of risk for the full pattern set case than decision-makers with switching elements on every input. Therefore for the full pattern set case the key problem in reducing risk is to improve the reliability of threshold logic decision-makers.

It is shown in Section 2.5 that, for the almost empty pattern set case, the area of applicability of threshold logic decision-makers is limited to small numbers, n, of input channels. Even if n is small the decision-maker interconnection channel parameters must be roughly two orders of magnitude better than the channel parameters of switching elements for decision-makers such as relay contact networks. This is evidenced by the curves in Figure 2.5.4.

In general it can be said that, for the almost empty pattern set case, the best decision-maker is a conventional network of switching elements. For the full pattern set case threshold logic networks are sufficiently versatile and are superior in terms of risk than networks of switching elements. The versatility of threshold logic networks as opposed to single threshold logic units is required by the complexity of the decision problems that can arise with a full pattern set and

unreliable sensors. Chapter III explains the reason for the difference in versatility between networks of threshold logic units and single units.

CHAPTER III

THRESHOLD LOGIC DECISION-MAKERS

3.1 Introduction. In the previous chapter the general area of applicability of threshold logic unit (TLU) decision-makers is defined. In this chapter the primary goal is to develop a mathematical model to aid in a scheme for improving the reliability of TLU decision-makers and thus increasing their applicability.

Section 3.2 is devoted to a presentation of the fundamentals of threshold logic and to pointing out more specifically some of the properties of TLU's. An explanation is presented of the role in decision making of single TLU's as opposed to networks of TLU's. The justification for the research on TLU network reliability is also discussed.

A general discussion of TLU networks is presented in Section 3.3 along with the particular network considered in the remainder of the work presented here. Figure 3.3.5 illustrates this network and some of the notation used. Sections 3.4 and 3.5 present the mathematical model which is used in Chapter IV to introduce redundancy in a TLU network.

3.2 Threshold Logic Units. A TLU is a device which produces a linear weighted sum of n inputs and subjects this sum to a threshold detector as shown in Figure 3.2.1. A vector $\bar{y}' = (y_1, y_2, \ldots, y_n)$ in the pattern space will produce function values $f(\bar{y}')$ as follows:

$$f(\overline{y}') = \begin{cases} 1 \text{ if } \overline{w}' \cdot \overline{y}' > -w_{n+1} \\ \\ -1 \text{ if } \overline{w}' \cdot \overline{y}' < -w_{n+1} \end{cases} \quad . \tag{3.2.1}$$

The vector $\overline{w}' = (w_1, w_2, \ldots, w_n)$ consists of the first n weights of the TLU. It is convenient to use an augmented vector $\overline{y} = (y_1, y_2, \ldots, y_n, 1)$ rather than $\overline{y}'$ as above. Similarly let $\overline{w} = (w_1, w_2, \ldots, w_{n+1})$. The vectors $\overline{y}$ and $\overline{w}$ are called "augmented pattern vector" and "weight vector" respectively while $\overline{y}'$ is called "pattern vector." It is assumed that the components of $\overline{y}'$ are binary; that is, $y_k = -1$ or $y_k = 1$ for $k = 1, 2, \ldots, n$. Also it is assumed that the TLU is designed such that $\overline{w}' \cdot \overline{y}' = -w_{n+1}$ never occurs so that the system of <u>strict</u> inequalities in Equation 3.2.1 can be used to generate a mathematical model.

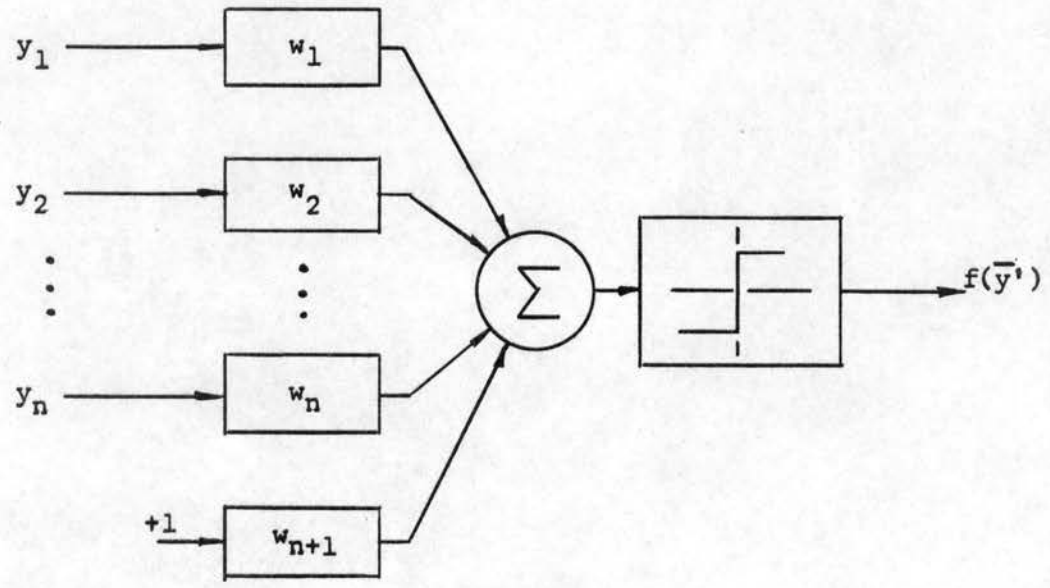


Figure 3.2.1. Threshold Logic Unit

Basically the task of a TLU is to dichotomize two sets of pattern vectors

$$S_f^+ = \{\bar{y}' \mid f(\bar{y}') = 1\}$$

and

$$S_f^- = \{\bar{y}' \mid f(\bar{y}') = -1\} \quad .$$

In order to better visualize this dichotomization the geometric representation of Equation 3.2.1 is appropriate. Consider a two-dimensional pattern space as shown in Figure 3.2.2.

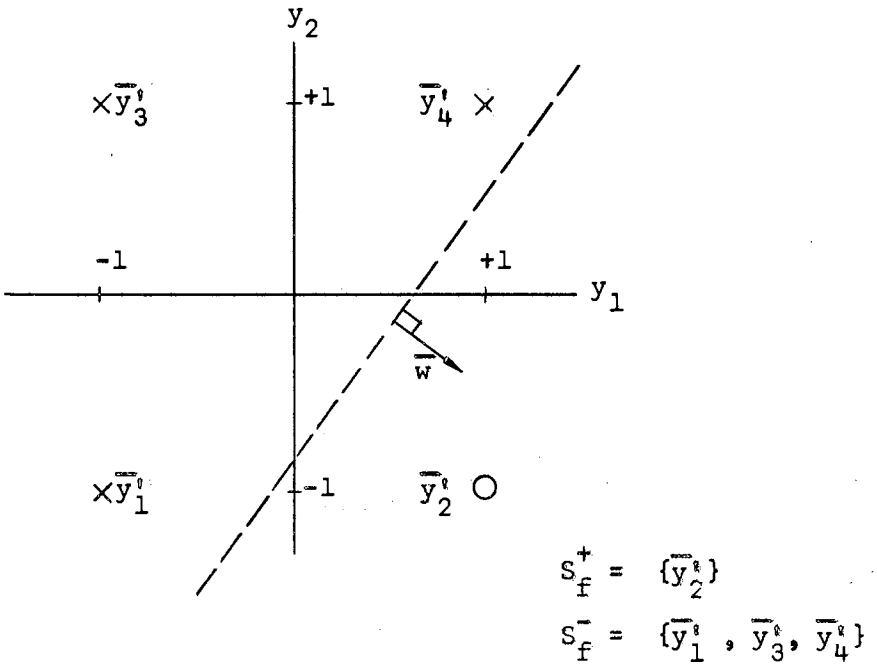


Figure 3.2.2. Two-Dimensional Pattern Space

The equality

$$\overline{w}' \cdot \overline{y}' = - w_{n+1} \qquad (3.2.2)$$

describes a hyperplane (in this case a line) passing through the space such that it is a distance of $|w_{n+1}|/||\overline{w}'||$ from the origin. The notation $||\overline{w}'||$ means the magnitude of the vector $\overline{w}'$ and $|w_{n+1}|$ means the magnitude of the scalar w_{n+1}. In this particular example the pattern vectors in S_f^+ are on one side of the line and those in S_f^- are on the other. Notice that the vector $\overline{w}'$ points in the positive direction with respect to the hyperplane.

The situation discussed above can be extended to n dimensions; and, if there exists a vector $\overline{w}'$ such that Equation 3.2.2 describes a hyperplane where

$$\overline{w} \cdot \overline{y} > 0 \quad \text{if } \overline{y}' \epsilon S_f^+ \qquad (3.2.3)$$

and

$$\overline{w} \cdot \overline{y} < 0 \quad \text{if } \overline{y}' \epsilon S_f^- \quad , \qquad (3.2.3)$$

then the two sets of patterns S_f^+ and S_f^- are said to be "linearly separable" (LS). Obviously linear separability is a necessary and sufficient condition for the existence of a vector $\overline{w}$ such that Inequalities 3.2.3 are true.

Given that a set of pattern vectors is LS, the vector $\overline{w}$ can be found in several ways. The more rigorous techniques involve the solution of a system of linear inequalities constructed from the pattern vectors in the sets S_f^+ and S_f^- and from $\overline{w}$. For each pattern vector, $\overline{y}_i$, one inequality results; if $\overline{y}_i \epsilon S_f^-$, the corresponding inequality is multiplied through by minus one so that all the inequalities in the system

are "greater than." If there are m^+ vectors in S_f^+ and m^- in S_f^- then the system can be written

$$\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} & 1 \\ y_{21} & y_{22} & \cdots & y_{2n} & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ y_{m^+1} & y_{m^+2} & \cdots & y_{m^+n} & 1 \\ -y_{(m^+\!+1)1} & -y_{(m^+\!+1)2} & \cdots & -y_{(m^+\!+1)n} & -1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ -y_{m1} & -y_{m2} & \cdots & -y_{mn} & -1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \cdot \\ \cdot \\ \cdot \\ w_{n+1} \end{bmatrix} > \begin{bmatrix} 0 \\ 0 \\ \cdots \\ 0 \\ 0 \\ \cdots \\ 0 \end{bmatrix}$$

where $m = m^+ + m^-$ and the vectors $\bar{y}_i$ are ordered such that $\bar{y}_1', \bar{y}_2', \ldots, \bar{y}_{m^+}'$ are in S_f^+ and $\bar{y}_{m^+\!+1}', \bar{y}_{m^+\!+2}', \ldots, \bar{y}_m'$ are in S_f^-. This system can be solved by linear programming, by relaxation techniques, or by techniques presented by Coates and Lewis (4), Dertouzos (6), Ho (11), Kaszerman (15), or Stokes (23) to mention a few.

Some other methods of finding a vector $\bar{w}$ rely upon the learning machine properties of TLU's. Nilsson (17) discusses learning machines quite extensively and presents training procedures (solution techniques) with convergence proofs for finding $\bar{w}$.

Threshold logic units possess some interesting properties that have helped to arouse interest in them. As pointed out by Brain (1) the simulation techniques used for neural elements have certain properties which are possessed by TLU's; thus the TLU is thought of by some as a neural element model. Even though the adaptability and other electrical properties of TLU's are used for brain simulations, the brain's redundancy properties are not considered in the simulations.

Consider the concept of distributed redundancy introduced in Section 2.6. Obviously each input weight of a TLU has an effect on every output when the input components are +1 or -1. In order to insure that the failure of one of the input weights (say an open circuit which changes the weight to zero) does not produce an overall failure, the TLU must be designed to correctly classify the resulting patterns as seen at the input to the summing device. The synthesis technique presented in Chapter IV can be extended to apply in this situation; however, this extension is not presented here.

Another important property of a TLU is the simplicity with which it can be implemented. Corresponding to the switching elements on each input of a network of switching elements, the elements on each input of a TLU can be simple resistors with the entire weighting and summing portion being constructed as a Kirchoff adder shown in Figure 3.2.3. In Chapter II the decision maker to which the network with input switching elements is compared is assumed to have perfect input elements. The relative simplicity of the input elements (resistors in the example shown in Figure 3.2.3) for the TLU decision-maker does not appreciably invalidate the comparison.

Perhaps the most serious limitation of TLU decision-makers is their lack of versatility. A single TLU is capable of dichotomizing only sets of pattern vectors which are LS. Consider for example the nonlinearly separable (NLS) pattern set shown in Figure 3.2.4. In this case it is not possible to pass a line through the pattern space such that the sets S_f^+ and S_f^- are separated, therefore a single TLU cannot dichotomize the patterns.

There are certain decision problems which are LS. The almost empty

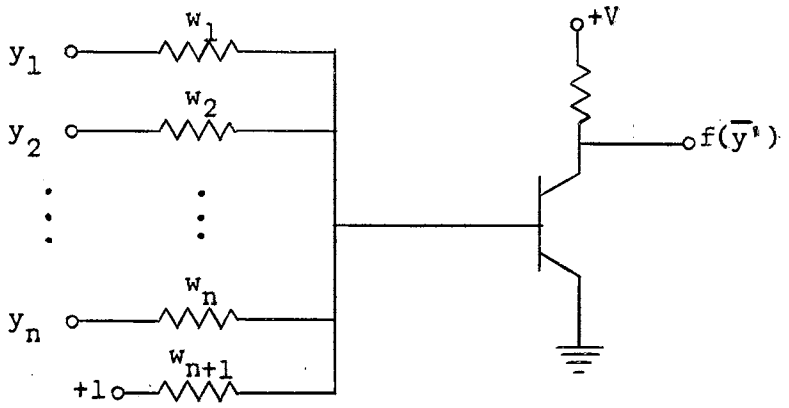


Figure 3.2.3. TLU Implementation

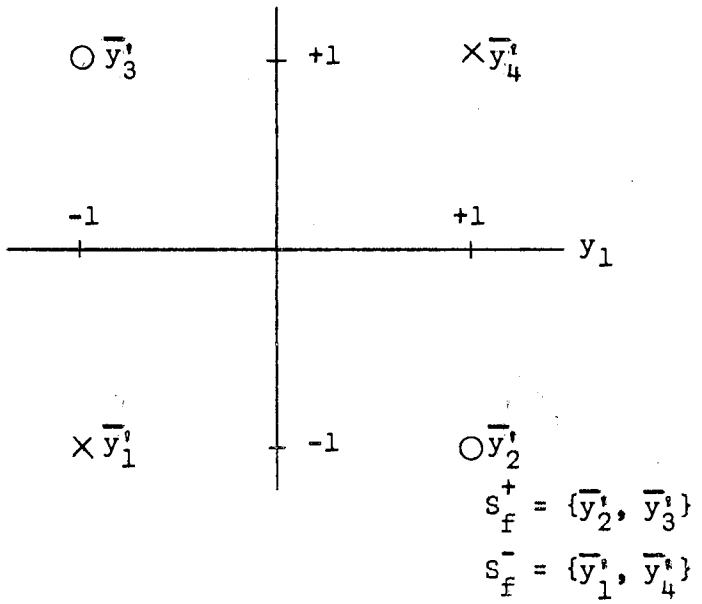


Figure 3.2.4. Nonlinearly Separable Pattern Set.

pattern set case is an outstanding example. In this case the optimum decision-maker classification is essentially a majority rule type of decision as shown by the optimum value of k_i^- in Equation 2.4.7. A pattern classification which is a majority rule is a very simple LS

problem, thus allowing the use of a single TLU. However, it is pointed out in Chapter II that for an almost empty pattern set case a conventional switching element network is better in terms of risk unless highly reliable TLU's can be designed.

TLU decision-makers are not applicable unless more complex decision problems occur. The full pattern set case is shown in Chapter II to justify the use of TLU decision-makers. However, a full pattern set problem cannot be depended upon to result in a LS decision-maker's optimum classification. For this reason a single TLU cannot in general be used in those situations which justify TLU decision-makers. Therefore networks of TLU's must be used and the improvement of their reliability is a valid problem.

3.3 Networks of Threshold Logic Units. In general TLU networks can be arranged in a variety of ways; however, so little is known about their synthesis that the configurations are limited. One synthesis procedure developed by Hopcroft and Mattson (13) is based on fundamental mathematical principles and is very promising provided that the amount of computation required can be reduced. This procedure provided the idea behind the redundancy synthesis procedure presented in Chapter IV.

There are other techniques suggested in Nilsson (17) which are intuitive and more easily understood than that of Hopcraft and Mattson. Basically Nilsson's procedures amount to transforming NLS patterns in one space into another space (image space) so that the patterns in the latter are LS. Figure 3.3.1 shows a representation of such a situation. The ν first layer TLU's are synthesized so that the patterns presented to TLU $m_{\nu+1}$ are LS. The total network is called a "two-layered machine."

The problem with this configuration (or with any other besides a single TLU) is that there are no known, convergent training procedures. However, there is a training procedure which has been successful and is used to synthesize a two-layered machine called a "committee machine." The reader is referred to page 97 of Nilsson (17) for a very concise explanation of this procedure.

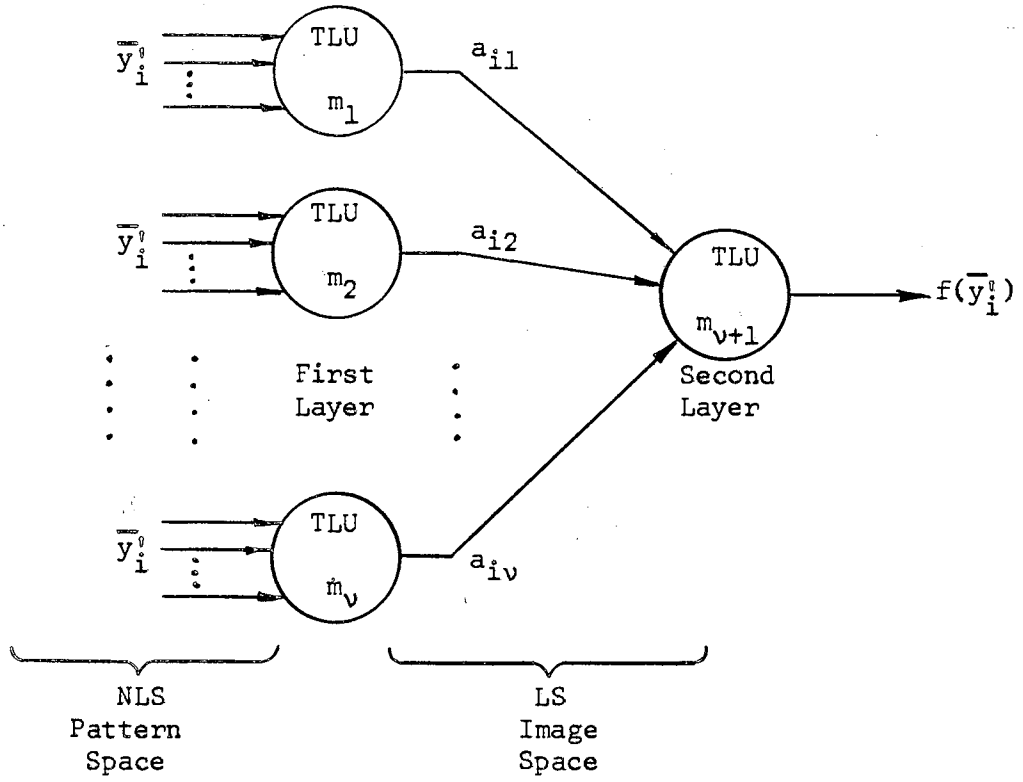


Figure 3.3.1. Two-Layer TLU Network

There is another synthesis procedure which works quite well for pattern spaces which can be visualized. With this method parallel hyperplanes are passed through the pattern space such that the regions

(cells) between the hyperplanes contain patterns belonging exclusively to either S_f^+ or S_f^-. Nilsson (17) shows on page 108 that this is a sufficient condition for linear separability of the image space. An example of this procedure is given in Chapter IV.

Since the primary concern here is with redundancy, the synthesis of nonredundant TLU networks is not pursued further. It is assumed here that a two-layered TLU network can be found and that, once it is found, the problem is to introduce redundancy into the network.

3.4 Mathematical Model of Two-Layer TLU Networks. Given the TLU shown in Figure 3.2.5, the system of linear inequalities to be solved by the weight vector $\bar{w}$ of TLU $m_{\nu+1}$ can be written using the output vectors $\bar{a}_i'$ whose components are the outputs $a_{i1}, a_{i2}, \ldots, a_{i\nu}$ of the first layer TLU's. Corresponding to the ith pattern vector $\bar{y}_i'$ there is an image vector $\bar{a}_i'$. Since the TLU network is assumed to be designed such that the NLS sets S_f^+ and S_f^- are transformed into the LS sets

$$S_g^+ = \{\bar{a}_i' \mid g(\bar{a}_i') = 1;\ \bar{a}_i' \text{ produced by } \bar{y}_i';\ \bar{y}_i' \varepsilon S_f^+;\ i = 1, 2, \ldots, m^+\}$$

and

$$S_g^- = \{\bar{a}_i' \mid g(\bar{a}_i') = -1;\ \bar{a}_i' \text{ produced by } \bar{y}_i';\ \bar{y}_i' \varepsilon S_f^-;\ i = m^+ + 1, m^+ + 2, \ldots, m\},$$

then there exists a weight vector $\bar{w}$ such that

$$\bar{w} \cdot \bar{a}_i' > -w_{\nu+1} \quad \text{if } \bar{a}_i' \varepsilon S_g^+$$

and

$$\bar{w} \cdot \bar{a}_i' < -w_{\nu+1} \quad \text{if } \bar{a}_i' \epsilon S_g^- \ .$$

Letting

$$\bar{a}_i^p = (\bar{a}_i' \mid 1) = (a_{i1}, a_{i2}, \ldots, a_{i\nu}, 1) \text{ if } \bar{a}_i' \epsilon S_g^+$$

and

$$\bar{a}_i^p = -(\bar{a}_i' \mid 1) = -(a_{i1}, a_{i2}, \ldots, a_{i\nu}, 1) \text{ if } \bar{a}_i' \epsilon S_g^- \ ,$$

the system of linear inequalities can be written as

$$\underline{A}_p \bar{w}^T > \bar{0}^T \qquad (3.4.1)$$

where

$$\underline{A}_p = \begin{bmatrix} \bar{a}_1^p \\ \bar{a}_2^p \\ \cdots \\ \bar{a}_{m^+}^p \\ \bar{a}_{(m^+\!+1)}^p \\ \cdots \\ \bar{a}_m^p \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1\nu} & 1 \\ a_{21} & a_{22} & \cdots & a_{2\nu} & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{m^+1} & a_{m^+2} & \cdots & a_{m^+\nu} & 1 \\ -a_{(m^+\!+1)1} & -a_{(m^+\!+1)2} & \cdots & -a_{(m^+\!+1)\nu} & -1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ -a_{m1} & -a_{m2} & \cdots & -a_{m\nu} & -1 \end{bmatrix}_{m\times(\nu+1)} \qquad (3.4.2)$$

$$\bar{w} = (w_1, w_2, \ldots, w_\nu, w_{\nu+1}) \ ,$$

and

$$\bar{0} = (0, 0, \ldots, 0) \ .$$

The rows of the matrix $\underline{A}_P$ are not necessarily unique. Consider, for example, the parallel hyperplane synthesis technique discussed in Section 3.3. All of the patterns $\overline{y}'_i$ in a particular cell between two hyperplanes are mapped into the same point in the image space. Therefore, corresponding to each of these $\overline{y}'_i$'s there is one vector $\overline{a}_{i'}$ in the image space. The matrix $\underline{A}_P$ is the basis for the generation of additional matrices which simulate failures among the first layer TLU's.

3.5 Redundancy for Fallible, Two-Layer, TLU Networks. The types of failures which are permitted are restricted here to two types. One of the two types of failures, "type $\mu = -1$" or "type $\mu = +1$", occurs when some input to a weight w_i of TLU $m_{\nu+1}$ is equal to μ for all pattern vectors $\overline{y}'_i$.

An alternate error simulation technique, which is not used here, is to let the output of some first layer TLU vary from 1 to -1 or -1 to 1 depending upon whether the correct state for a particular $\overline{y}_i$ is 1 or -1 respectively. This method of failure simulation is not satisfactory because a particular first layer TLU would still have to vary its output corresponding to different patterns $\overline{y}'_i$. In effect the output of this TLU would be negated while the TLU continued to function properly.

The result of a type 1 or type -1 failure is to increase the complexity of the system of linear inequalities which the weights w_i of TLU $m_{\nu+1}$ must solve. Assume that TLU m_1 experiences a type 1 failure, then the vectors

$$(1, a_{i2}, a_{i3}, \ldots, a_{i\nu}, 1), \quad i = 1, 2, \ldots, m$$

must be classified into the same catagories as the failure-free vectors

$$(a_{i1}, a_{i2}, \ldots, a_{i\nu}, 1), \quad i = 1, 2, \ldots, m$$

if no overall failures are permitted. Thus, a type 1 failure in a specific TLU has added m rows to $\underline{A}_p$. If an arbitrary failure type μ (μ equal to -1 or 1) is possible in any of the ν first layer TLU's and if no overall failures are permitted, then the system of linear inequalities to be solved by $\overline{w}$ is

$$\underline{A}_o \overline{w}^T > \overline{0}^T \tag{3.5.1}$$

where A_o is given in Equation 3.5.2. As in $\underline{A}_p$, $\underline{A}_o$ may contain repeated rows. These repetitions will be used to provide some simplification in the synthesis algorithm in Chapter IV.

It is assumed that Equation 3.4.1 has a solution; however, it is obvious that Equation 3.5.1 may not have a solution. The basic idea used here is to force Equation 3.5.1 to have a solution and was presented by Hopcroft and Mattson (13). The idea is to add columns to $\underline{A}_o$ and this idea is developed further in Appendix A. The technique amounts to adding additional TLU's in the first layer of the network and thereby increasing the dimension of the image space. One of the most significant differences between the work presented here and that done by Hopcroft and Mattson is that the additional (or redundant) TLU's are fallible themselves.

Let the binary outputs of the redundant TLU's, $m_k^{(R)}$, $k > \nu + 2$, be represented by $c'_{i1}, c'_{i2}, \ldots, c'_{i\gamma}$, $i = 1, 2, \ldots, m$, where γ is the number of redundant TLU's. For the kth redundant TLU, $m_k^{(R)}$, a vector

$$
\underline{A}_o = \left[\begin{array}{ccccc}
a_{11} & a_{12} & \cdots & a_{1\nu} & 1 \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
a_{m^+1} & a_{m^+2} & \cdots & a_{m^+\nu} & 1 \\
-a_{(m^++1)1} & -a_{(m^++1)2} & \cdots & -a_{(m^++1)\nu} & -1 \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
-a_{m1} & -a_{m2} & \cdots & -a_{m\nu} & -1 \\
\hline
\mu & a_{12} & \cdots & a_{1\nu} & 1 \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
\mu & a_{m^+2} & \cdots & a_{m^+\nu} & 1 \\
-\mu & -a_{(m^++1)2} & \cdots & -a_{(m^++1)\nu} & -1 \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
-\mu & -a_{m2} & \cdots & -a_{m\nu} & -1 \\
\hline
\vdots & \vdots & \vdots & \vdots & \vdots \\
\hline
a_{11} & a_{12} & \cdots & \mu & 1 \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
a_{m^+1} & a_{m^+2} & \cdots & \mu & 1 \\
-a_{(m^++1)1} & -a_{(m^++1)2} & \cdots & -\mu & -1 \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
-a_{m1} & -a_{m2} & \cdots & -\mu & -1
\end{array}\right]_{m(\nu+1)x(\nu+1)}
= \left[\begin{array}{c} \underline{A}_P \\ \hline \underline{A}_R \end{array}\right] \qquad (3.5.2)
$$

$$\bar{c}'_k = (c'_{1k}, c'_{2k}, \ldots, c'_{mk}) \tag{3.5.3}$$

is used to describe the m outputs of the TLU corresponding to the m inputs $\bar{y}'_i$. Notice that this vector differs from $\bar{a}'_i$ and $\bar{y}'_i$ since its components correspond to different inputs to $m_k^{(R)}$ occuring at different times rather than components of a pattern or image space vector occuring at the same time. Letting $\gamma = 1$ the system of linear inequalities which $\bar{w}_1 = (w_1, w_2, \ldots, w_{\nu+2})$ must solve is

$$[\underline{A}_1 \,\vdots\, (\bar{c}'_1)]\bar{w}_1^T > \bar{0}^T \tag{3.5.4}$$

where

$$\underline{A}'_1 = [\underline{A}_1 \,\vdots\, (\bar{c}_1^1)^T] = \begin{bmatrix} & \vdots & \bar{c}_1^T \\ & \vdots & \bar{c}_1^T \\ \underline{A}_0 & \vdots & \vdots \\ & \vdots & \bar{c}_1^T \\ & \vdots & \mu \\ & \vdots & \vdots \\ & \vdots & \mu \\ \underline{A}_P & \vdots & \\ & \vdots & -\mu \\ & \vdots & \vdots \\ & \vdots & -\mu \end{bmatrix} \begin{array}{l} \text{no failure} \\ \text{failure in } m_1 \\ \\ \text{failure in } m_2 \\ \\ \\ \text{failure in } m_\nu^{(R)} \end{array} \tag{3.5.5}$$

for a type μ failure. The first m^+ components of $\bar{c}_1$ are the same as those of $\bar{c}'_1$ while the last m^- are the negative of those in $\bar{c}'_1$. The vector $\bar{c}_1^1$ is defined by the verticle partition on the right of $[\underline{A}_1 \,\vdots\, (\bar{c}_1^1)^T]$.

In general the weight vector $\bar{w}_\gamma = (w_1, w_2, \ldots, w_{\nu+\gamma+1})$ must solve

$$\underline{A}'_\gamma \overline{w}_\gamma^T > \overline{0}^T \qquad (3.5.6)$$

where $\underline{A}'_\gamma$ is given by Equation 3.5.7. The matrix $\underline{A}'_\gamma$ is the mathematical model for single, type μ failures in any of the $\nu + \gamma$ first layer TLU's shown in Figure 3.5.1. The matrix $\underline{A}_\gamma$ is defined by

$$\underline{A}_\gamma = \begin{bmatrix} \underline{A}_o \\ \hline \underline{A}_P \\ \hline \vdots \\ \hline \underline{A}_P \end{bmatrix} \begin{matrix} \text{failures in } m_1 \cdots m_\nu \\ \text{failures in } m_1^{(R)} \\ \\ \text{failures in } m_\gamma^{(R)} \end{matrix}$$

$$m(\nu+\gamma+1)x(\nu+1)$$

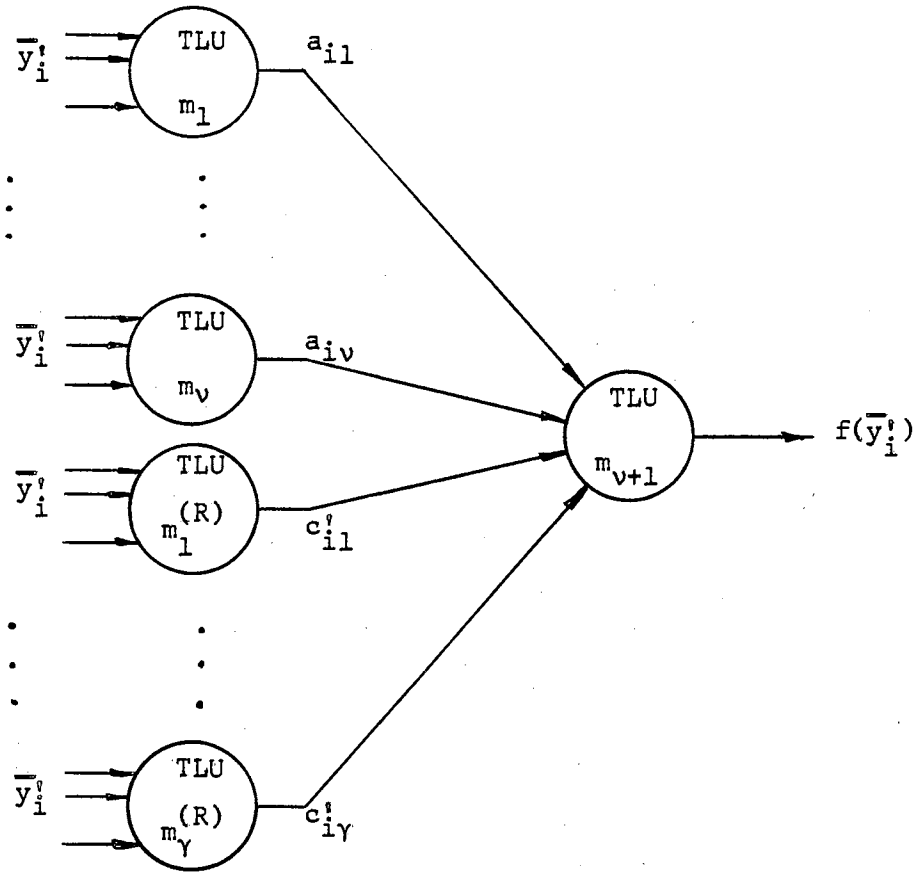


Figure 3.5.1. TLU Network With Redundancy.

$$\underline{A}'_{\gamma} = [\underline{A}_{\gamma} \mid (\bar{c}_1^{\gamma})^T \mid \cdots \mid (\bar{c}_{\gamma}^{\gamma})^T] =$$

$$\begin{array}{|c|c|c|c|c|l|}
\hline
 & \bar{c}_1^T & \bar{c}_2^T & \cdots & \bar{c}_{\gamma}^T & \text{no failures} \\
\hline
 & \bar{c}_1^T & \bar{c}_2^T & \cdots & \bar{c}_{\gamma}^T & \text{failure in } m_1 \\
\hline
 & \cdots & \cdots & \cdots & \cdots & \vdots \\
\underline{A}_0 & & & & & \\
 & \bar{c}_1^T & \bar{c}_2^T & \cdots & \bar{c}_{\gamma}^T & \text{failure in } m_{\nu} \\
\hline
 & \mu & & & & \\
 & \vdots & & & & \\
 & \mu & & & & \\
\underline{A}_p & & \bar{c}_2^T & \cdots & \bar{c}_{\gamma}^T & \text{failure in } m_1^{(R)} \\
 & -\mu & & & & \\
 & \vdots & & & & \\
 & -\mu & & & & \\
\hline
\vdots & \vdots & \vdots & \vdots & \vdots & \\
\hline
 & & & & \mu & \\
\underline{A}_p & \bar{c}_1^T & \bar{c}_2^T & \cdots & \vdots & \text{failure in } m_{\gamma}^{(R)} \\
 & & & & \mu & \\
 & & & & -\mu & \\
 & & & & \vdots & \\
 & & & & -\mu & \\
\hline
\end{array}$$

$\nu+1$ columns $\qquad$ γ columns $\qquad$ $m(\nu+\gamma+1)x(\nu+\gamma+1)$. (3.5.7)

In this model both type -1 and type 1 failures are not permitted to occur simultaneously implying only single failures of one type are corrected; however, this can be done by extending $\underline{A}'_{\gamma}$. An example is presented in Section 4.7 which illustrates this extension of the theory developed here.

The development of a method of selection of the $\bar{c}_k$'s is the major problem solved in Chapter IV. This selection must be made on the basis of the restrictions imposed by the requirement that Equation 3.5.6 must

have a solution and by the requirement that $\bar{c}_k'$ be realizable by a TLU. The first restriction is imposed with the aid of the theory developed in Appendices A and B while the last restriction is imposed on a trial and error basis as shown in the synthesis algorithm.

3.6 Conclusion. The sections of this chapter present material which is needed before the synthesis algorithm of Chapter IV can be understood. The basic notation and mathematical model of two-layer TLU decision-makers is presented after a brief discussion of the reasons for using TLU networks for decision making. This chapter in effect is a link between Chapters II and IV.

CHAPTER IV

DEVELOPMENT OF THE REDUNDANCY SYNTHESIS ALGORITHM

4.1 Introduction. The problem solved in this chapter is formulated in Chapter III. As stated there the problem is to develop a technique for the selection of vectors $\overline{c}_k$ such that there exists a $\overline{w}_\gamma$ satisfying

$$\underline{A}'_\gamma \overline{w}_\gamma^T > \overline{0}^T$$

and such that $\overline{c}'_k$, $k = 1, 2, \ldots, \gamma$, are realizable as the output vectors of γ redundant TLU's, $m_1^{(R)}, m_2^{(R)}, \ldots, m_\gamma^{(R)}$. In Appendix A a technique is developed and a computer program is presented finding a set S_ϕ of extremal vectors $\overline{\phi}_k$, $k = 1, 2, \ldots, s$, which form a system of linear inequalities,

$$\begin{bmatrix} \overline{\phi}_1 \\ \overline{\phi}_2 \\ \vdots \\ \overline{\phi}_s \end{bmatrix} \overline{\theta}^T \gtrless \overline{0}^T ,$$

that a general vector $\overline{\theta}$ must solve in order for a system

$$[\underline{A} \mid \overline{\theta}^T] \overline{w}_1^T > \overline{0}^T$$

to have a solution $\overline{w}_1$ when

$$\underline{A}\overline{w}_o^T > \overline{0}^T$$

has no solution. Extremal vectors are defined in Section A.2. The symbol $\gtrless$ means that either > or < holds, but not both, for every inequality in the system. That is, either

$$\begin{bmatrix} \overline{\phi}_1 \\ \overline{\phi}_2 \\ \vdots \\ \overline{\phi}_s \end{bmatrix} \overline{\theta}^T > \overline{0}^T$$

or

$$\begin{bmatrix} \overline{\phi}_1 \\ \overline{\phi}_2 \\ \vdots \\ \overline{\phi}_s \end{bmatrix} \overline{\theta}^T < \overline{0}^T$$

but not both. The symbols > or < for vectors mean component by component. In this chapter this technique is used to derive a similar set of "inequality constraints" to apply to the vectors in the vertical partitions on the right of the matrix $\underline{A}'_\gamma$. Since these vector partitions contain the vectors $\overline{c}_1, \overline{c}_2, \ldots, \overline{c}_\gamma$, and since the latter must correspond as discussed in Section 4.2 to vectors $\overline{c}'_1, \overline{c}'_2, \ldots, \overline{c}'_\gamma$ which are realizable with TLU's; the partitions must be restricted by "realizability constraints." A partial application of the realizability constraints is performed by a matrix transformation in Section 4.4 resulting in a simplified set of inequality constraints. A suitable $\overline{c}_i$ must satisfy these simplified constraints along with the realizability

of the corresponding $\bar{c}_i'$. In order to find a suitable $\bar{c}_i$ in a given situation, those $\bar{c}_i$'s satisfying the simplified constraints must be successively tested for realizability by using the LS test provided by the technique of Appendix A. A discussion of TLU realizability of a $\bar{c}_i'$ is given in Section 4.2. A digital computer program has been written by the author to perform the search for a $\bar{c}_i'$ with the IBM 7040 computer at Oklahoma State University. This program is presented in Section D.2 and in an altered form in Section D.3.

In the event that more than one redundant TLU must be used ($\gamma > 1$), a problem arises of maintaining single, type μ error correction with imperfect redundant TLU's. This problem is approached by a general synthesis algorithm which is developed by first developing the constraints on $\bar{c}_1$, then on $\bar{c}_2$ given $\bar{c}_1$, and then on $\bar{c}_\gamma$ given $\bar{c}_{\gamma-1}$, $\bar{c}_{\gamma-2}$, ..., $\bar{c}_1$. Using the notation developed in this process, the general synthesis algorithm is presented in Section 4.6. A running example is used to clarify each major step in the development contained in the following sections.

4.2 TLU Realizability of a $\bar{c}_i'$. The selection of a suitable $\bar{c}_i$ procedes by first finding a $\bar{c}_i$ which satisfies a set of inequality constraints and then testing the corresponding $\bar{c}_i'$ to see if it is realizable with a TLU. The definition of $\bar{c}_i'$ is given by Equation 3.5.3 where it is stated that the components c_{1i}', c_{2i}', ..., c_{mi}' of $\bar{c}_i'$ are the outputs of TLU $m_i^{(R)}$ for the inputs $\bar{y}_1'$, $\bar{y}_2'$, ..., $\bar{y}_m'$. Basically the components of $\bar{c}_i'$ serve to assign to each $\bar{y}_k'$, k = 1, 2, ..., m membership in a set

$$S_{c_i}^{+} = \{\bar{y}_k' \mid c_{ki}' = +1\}$$

or

$$S_{c_i}^{-} = \{\overline{y}_k' \mid c_{ki}' = -1\} \quad .$$

In order to test for realizability of $\overline{c}_i'$ the technique of Appendix A can be used in the capacity of a linear separability test. This test is performed by using a matrix $\underline{Y}_{c_i}$ whose rows are the vectors $[\overline{y}_i' \mid 1]$ or $-[\overline{y}_i' \mid 1]$ depending upon whether $\overline{y}_i' \epsilon S_c^{+}$ or $\overline{y}_i' \epsilon S_c^{-}$ respectively. If there are no extremal vectors for the set of solutions to

$$\underline{Y}_{c_i}^{T} \overline{\phi}^{T} = \overline{0}^{T} \quad , \quad \overline{\phi} \neq \overline{0} \quad , \quad \overline{\phi} \geq \overline{0} \quad ,$$

then S_c^{+} and S_c^{-} are LS and $\overline{c}_i'$ is realizable. This test is discussed in general in Appendix A.

4.3 Derivation and Partitioning of the Matrices of Extremal Vectors. One of the fundamental ideas behind the synthesis algorithm is that it is successively assumed that: first, one redundant TLU is sufficient ($\gamma = 1$); then, if $\gamma \neq 1$ that $\gamma = 2$; and then if $\gamma \neq 2$, that $\gamma = 3$; etc. Therefore at a particular point in the procedure and with an assumed value of γ the matrix $\underline{A}_{\gamma}'$ is altered by selecting $\overline{c}_1$, $\overline{c}_2$, ..., $\overline{c}_{\gamma}$ so that, hopefully, there exists a vector $\overline{w}_{\gamma}$ satisfying

$$\underline{A}_{\gamma}' \overline{w}_{\gamma}^{T} > \overline{0}^{T} .$$

In selecting $\overline{c}_1$, $\overline{c}_2$, ..., $\overline{c}_{\gamma}$ it is necessary to use vectors $\overline{\phi}_k^{\gamma}$ which are extremal vectors for the set of solutions to

$$\underline{A}_{\gamma}^{T} (\overline{\phi}^{\gamma})^{T} = \overline{0}^{T}, \quad \overline{\phi}^{\gamma} \neq \overline{0} \quad , \quad \overline{\phi}^{\gamma} \geq \overline{0} \quad .$$

The set of solutions forms a convex polyhedral cone as discussed in

Appendix A; and the vectors $\bar{\phi}_k^{\gamma}$ are said to arise because of positive linear dependencies (PLD's) among the columns of $\underline{A}_{\gamma}^{T}$ corresponding to nonzero elements of the $\bar{\phi}_k^{\gamma}$'s. These extremal vectors form a set S_{ϕ}^{γ} and are the transpose of the columns in a matrix

$$\underline{\Phi}_{\gamma} = [(\bar{\phi}_1^{\gamma})^T, (\bar{\phi}_2^{\gamma})^T, \ldots, (\bar{\phi}_{s_{\gamma}}^{\gamma})^T]_{m(\nu+\gamma+1)xs_{\gamma}}$$

where s_{γ} is the number of elements in the set S_{ϕ}^{γ}. It should be noted that the set S_{ϕ}^{γ} is not necessarily unique. The matrix $\underline{\Phi}_{\gamma}$ is used in Section 4.4 to produce simplified systems of linear inequalities to apply to $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_{\gamma}$.

The elements of S_{ϕ}^{γ} can be computed from $\underline{A}_{\gamma}$ using the technique of Appendix A; however, these elements can be found far more easily by first finding the elements of a reduced set S_{ϕ}^{*} and generating $\underline{\Phi}_{\gamma}$ from the reduced matrix $\underline{\Phi}_{o}^{*}$ by the extremal set extension technique of Appendix B. The set S_{ϕ}^{*} is a set of extremal vectors for the solutions $\bar{\phi}$ to

$$(\underline{A}_{o}^{*})^T \bar{\phi}^T = \bar{0}^T \ , \quad \bar{\phi} \neq \bar{0} \ , \quad \bar{\phi} \geq \bar{0} \ ,$$

and its elements are the transposed columns in $\underline{\Phi}_{o}^{*}$. The matrix $\underline{A}_{o}^{*}$ is formed from

$$\underline{A}_{o} = \left[\begin{array}{c} \underline{A}_P \\ \hline \underline{A}_R \end{array}\right] \begin{array}{l} m \text{ rows} \\ \hline \nu m \text{ rows} \end{array}$$

by first removing the redundant rows in $\underline{A}_P$ to form $\underline{A}_P^{*}$ and then removing the redundant rows remaining in $\underline{A}_o$ from $\underline{A}_R$ to form $\underline{A}_R^{*}$. Thus,

$$\underline{A}_o^* = \begin{bmatrix} \underline{A}_P^* \\ \hline \underline{A}_R^* \end{bmatrix} .$$

The matrices $\underline{A}_i$ defined in Section 3.5 contain no rows which are not in $\underline{A}_o^*$ because $\underline{A}_i$ is constructed from $\underline{A}_o$ by adding ν sets of the m rows from $\underline{A}_P$ to $\underline{A}_o$. This fact permits the application of the extremal set extension technique to generate $\underline{\Phi}_i$.

The partitioning of $\underline{\Phi}_o^*$ and $\underline{\Phi}_i$, i = 0, 1, ..., γ, is aided by the fact that there are no PLD's among the rows of $\underline{A}_P^*$ or $\underline{A}_P$. To show this absence of PLD's consider the matrix $\underline{A}_P$ defined by Equation 3.4.2. The rows of this matrix are obtained from the LS, image space, pattern vectors for the TLU $m_{\nu+1}$ in Figure 3.2.5. These vectors are LS because it is assumed that the TLU network correctly classifies the pattern space vectors provided that no failures occur. As a result of this linear separability, the system

$$\underline{A}_P \overline{w}^T > \overline{0}^T$$

has a solution; thus the system

$$\underline{A}_P^* \overline{w}^T > \overline{0}^T$$

has a solution where $\underline{A}_P^*$ is obtained from $\underline{A}_P$ as described above. Since these solutions exist, Gordon's Theorem (see Appendix A) states that there exist no solutions $\overline{\phi}$ to

$$(\underline{A}_P^*)^T \overline{\phi}^T = \overline{0}^T , \quad \overline{\phi} \neq \overline{0} , \quad \overline{\phi} \geq \overline{0} .$$

Therefore, there are no extremal vectors for the set of solutions and there are no PLD's among the columns of $(\underline{A}_P^*)^T$. In Appendix B it is

shown that, since there are no PLD's among the rows of $\underline{A}_P^*$, there are none among the rows of $\underline{A}_P$ or among the rows of any matrix derived from $\underline{A}_P^*$ by repeating its rows.

Corresponding to every PLD among the rows of $\underline{A}_O^*$ or $\underline{A}_i$, $i = 0, 1, \ldots, \gamma$, there is an extremal vector element in S_ϕ^* or S_ϕ^i, $i = 0, 1, \ldots, \gamma$, respectively. However, there can be no PLD's between rows of $\underline{A}_O^*$ or $\underline{A}_i$, $i = 0, 1, \ldots, \gamma$, if these rows occur only in $\underline{A}_P^*$ or $\underline{A}_P$. On the other hand, there can be PLD's among rows of $\underline{A}_P^*$ or $\underline{A}_P$ and rows that do not occur in $\underline{A}_P^*$ or $\underline{A}_P$. Thus it can be seen that there are two types of extremal vectors for the solutions to

$$(\underline{A}_O^*)^T \overline{\phi}^T = \overline{0}^T \ , \quad \overline{\phi} \neq \overline{0} \ , \quad \overline{\phi} \geq \overline{0}$$

or

$$(\underline{A}_i)^T (\overline{\phi}^i)^T = \overline{0}^T \ , \quad \overline{\phi}^i \neq \overline{0} \ , \quad \overline{\phi}^i \geq \overline{0} \ , \quad i = 0, 1, \ldots, \gamma \ .$$

One type of extremal vector has nonzero components corresponding to the rows creating a PLD exclusively among the rows of $\underline{A}_R^*$ or $\underline{A}_R$. The other type has nonzero components corresponding to the rows creating a PLD among the rows of both $\underline{A}_P^*$ or $\underline{A}_P$ and $\underline{A}_R^*$ or $\underline{A}_R$ respectively. As shown above there can be no extremal vectors whose components correspond to rows from $\underline{A}_P^*$ or $\underline{A}_P$ exclusively. Thus the matrix $\underline{\Phi}_O^*$ can be written in the form

$$\underline{\Phi}_O^* = \left[\begin{array}{c|c} \underline{\Phi}_P^* & \underline{0} \\ \hline \underline{\Phi}_{R_1}^* & \underline{\Phi}_{R_2}^* \end{array}\right] \begin{array}{l} \text{Corresponding to } \underline{A}_P^* \\ \text{Corresponding to } \underline{A}_R^* \end{array}$$

where $\underline{0}$ and $\underline{\Phi}_{R_2}^*$ contain the extremal vectors of the first type above and $\underline{\Phi}_P^*$ and $\underline{\Phi}_{R_1}^*$ contain those of the second type. The columns of $\underline{\Phi}_O^*$ are the

elements in S_{ϕ}^{*}.

As mentioned previously, the matrices $\underline{\Phi}_i$, $i = 0, 1, \ldots, \gamma$, can be obtained from $\underline{\Phi}_o^*$ by the extremal set extension technique of Appendix B. Consider the matrix $\underline{\Phi}_o$ and the manner in which it can be obtained from $\underline{\Phi}_o^*$. According to the technique in Appendix B, if a vector in S_{ϕ}^{*} corresponds to a PLD among some of the rows of $\underline{A}_o^*$ and if one of these rows is repeated in $\underline{A}_o$, then S_{ϕ}^{o} contains two vectors similar to the one in S_{ϕ}^{*}. One of these vectors in S_{ϕ}^{o} contains nonzero elements corresponding to the position of one of the repeated rows and to the positions of the remaining rows of the PLD. The other vector in S_{ϕ}^{o} contains nonzero elements in the same positions as the former vector except that a nonzero element occurs in the position corresponding to the repeated row not considered by the former. By applying this procedure repeatedly, for every combination of repeated rows in $\underline{A}_o$, the matrix $\underline{\Phi}_o$ can be written as

$$\underline{\Phi}_o = \left[\begin{array}{c|c} \underline{\Phi}_P & \underline{0} \\ \hline \underline{\Phi}_{R_1} & \underline{\Phi}_{R_2} \end{array}\right] \begin{array}{l} \text{corresponding to } \underline{A}_P \\ \text{corresponding to } \underline{A}_R \end{array} .$$

Consider $\underline{A}_i$ which is constructed from $\underline{A}_o$ by adding i sets of the m rows of $\underline{A}_P$ to $\underline{A}_o$ as in

$$\underline{A}_i = \left[\begin{array}{c} \underline{A}_P \\ \hline \underline{A}_R \\ \hline \underline{A}_P \\ \vdots \\ \underline{A}_P \end{array}\right] = \left[\begin{array}{c} \underline{A}_o \\ \hline \underline{A}_P \\ \vdots \\ \underline{A}_P \end{array}\right] .$$

The matrix $\underline{\Phi}_i$ can be written from $\underline{\Phi}_o$ by inspection as

$$\underline{\Phi}_i = \left[\begin{array}{c|c|c|c|c|c} \underline{\Phi}_P & \underline{0} & \underline{0} & \underline{0} & \cdots & \underline{0} \\ \hline \underline{\Phi}_{R_1} & \underline{\Phi}_{R_2} & \underline{\Phi}_{R_1} & \underline{\Phi}_{R_1} & \cdots & \underline{\Phi}_{R_1} \\ \hline \underline{0} & \underline{0} & \underline{\Phi}_P & \underline{0} & \cdots & \underline{0} \\ \hline \underline{0} & \underline{0} & \underline{0} & \underline{\Phi}_P & \cdots & \underline{0} \\ \hline \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \hline \underline{0} & \underline{0} & \underline{0} & \underline{0} & \cdots & \underline{\Phi}_P \end{array}\right] \qquad (4.3.1)$$

Example 4.3.1. Consider the nonredundant two-layer TLU network shown in Figure 4.3.1 and the corresponding pattern-to-image-space transformation shown in Figure 4.3.2. The pattern space vectors given by

$$\overline{y}'_1 = (1, 1)$$
$$\overline{y}'_2 = (-1, -1)$$
$$\overline{y}'_3 = (-1, 1)$$
$$\overline{y}'_4 = (1, -1)$$

produce corresponding image space vectors

$$\overline{a}'_1 = (1, 1)$$
$$\overline{a}'_2 = (-1, -1)$$
$$\overline{a}'_3 = (-1, 1)$$
$$\overline{a}'_4 = (-1, 1) \quad .$$

The latter can be used to form the matrix

$$\underline{A}_P = \begin{bmatrix} 1 & 1 & 1 \\ -1 & -1 & 1 \\ 1 & -1 & -1 \\ 1 & -1 & -1 \end{bmatrix}$$

from which the matrices $\underline{\Phi}_o^*$, $\underline{\Phi}_o$, and $\underline{\Phi}_1$ are generated using the techniques of Appendices A and B.

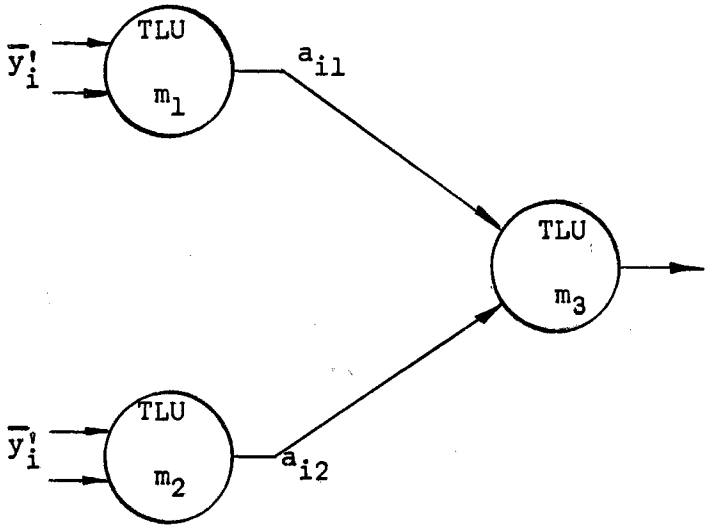


Figure 4.3.1. Nonredundant TLU Network for Example 4.3.1.

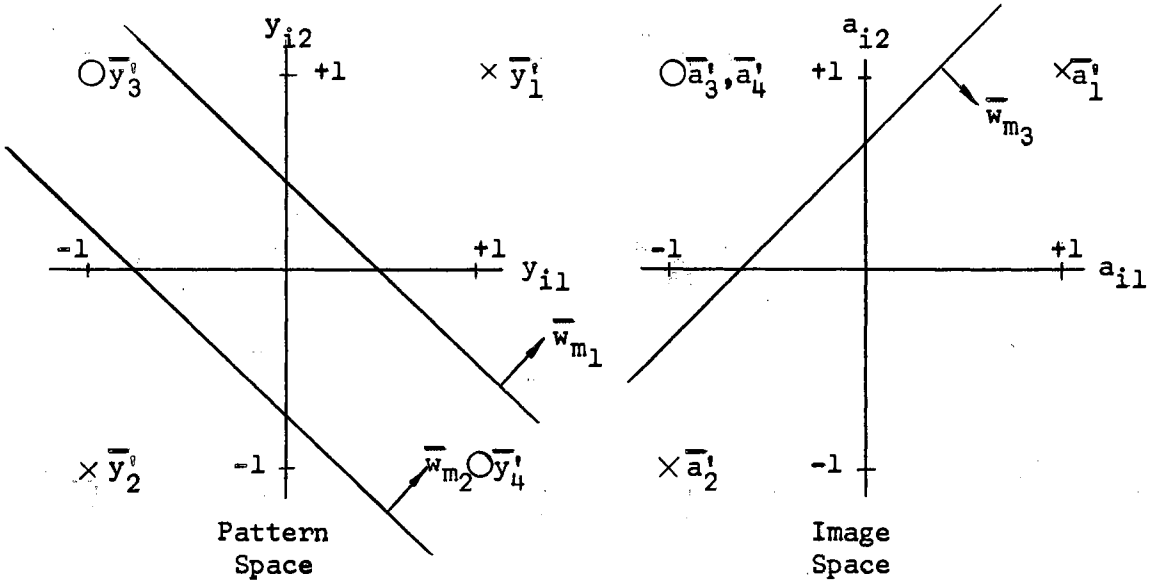


Figure 4.3.2. Pattern-to-Image-Space Transformation for Example 4.3.1.

For a type +1 failure in the first two columns ($\nu=2$ in this example) the matrix $\underline{A}_o$ is written as

$$\underline{A}_o = \left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 2 \\ 1 & -1 & -1 & 3 \\ 1 & -1 & -1 & 4 \\ \hline 1 & 1 & 1 & 5 \\ 1 & -1 & 1 & 6 \\ -1 & -1 & -1 & 7 \\ -1 & -1 & -1 & 8 \\ \hline 1 & 1 & 1 & 9 \\ -1 & 1 & 1 & 10 \\ 1 & -1 & -1 & 11 \\ 1 & -1 & -1 & 12 \end{array}\right].$$

The matrix $\underline{A}_o$ has repeated columns, therefore the labor involved in finding elements of S_ϕ^o can be reduced by finding those of S_ϕ^* from $\underline{A}_o^*$ and generating S_ϕ^o from S_ϕ^*.

$$\underline{A}_o^* = \left[\begin{array}{c} \underline{A}_P^* \\ \hline \underline{A}_R^* \end{array}\right] = \left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 2 \\ 1 & -1 & -1 & 3 \\ \hline 1 & -1 & 1 & 6 \\ -1 & -1 & -1 & 7 \\ -1 & 1 & 1 & 10 \end{array}\right].$$

Using the computer to perform the technique of Appendix A, the set S_ϕ^* contains the columns of $\underline{\Phi}_o^*$ as in

$$\underset{\sim}{\Phi}_o^* = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 6 \\ 7 \\ 10 \end{matrix} .$$

The generation of the matrix $\underset{\sim}{\Phi}_o$ from $\underset{\sim}{\Phi}_o^*$ is facilitated by a "transition chart" which tabulates the location of rows in $\underset{\sim}{A}_o$ which are identical to specified rows of $\underset{\sim}{A}_o^*$. The transition chart is shown in Figure 4.3.3.

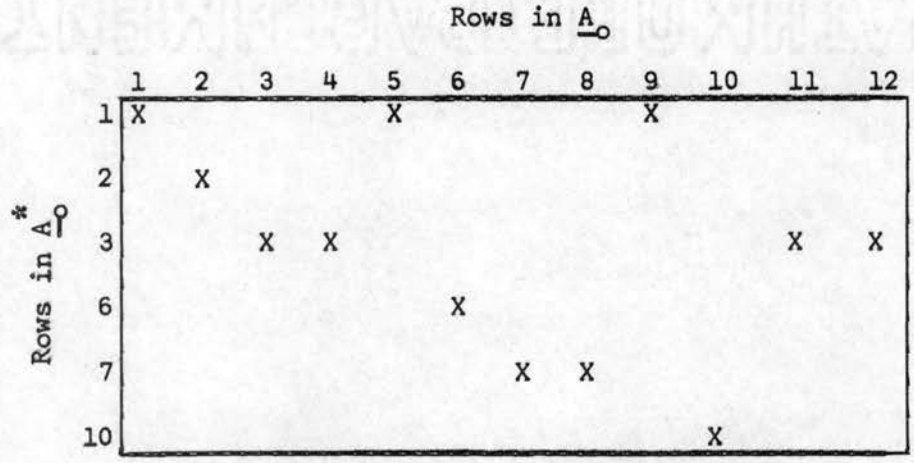


Figure 4.3.3. Transition Chart for $\underset{\sim}{A}_o$

To generate $\underset{\sim}{\Phi}_o$ notice that $\underset{\sim}{\Phi}_o^*$ indicates that there is a PLD between rows 1 and 7 and another between rows 3 and 10 of $\underset{\sim}{A}_o^*$. From the transition chart it can be seen that row 1 is identical to rows 5 and 9; row 7 is identical to row 8; row 3 is identical to rows 4, 11, and 12; and row 10 is identical to no other row. The result is that S_Φ^o contains ten

elements shown in

$$\underline{\Phi}_0 = \left[\begin{array}{cccc|cccccc|l}
1 & 1 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 1 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 2 \\
\circ & \circ & 1 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 3 \\
\circ & \circ & \circ & 1 & \circ & \circ & \circ & \circ & \circ & \circ & 4 \\
\hline
\circ & \circ & \circ & \circ & 1 & 1 & \circ & \circ & \circ & \circ & 5 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 6 \\
1 & \circ & \circ & \circ & 1 & \circ & 1 & \circ & \circ & \circ & 7 \\
\circ & 1 & \circ & \circ & \circ & 1 & \circ & 1 & \circ & \circ & 8 \\
\circ & \circ & \circ & \circ & \circ & \circ & 1 & 1 & \circ & \circ & 9 \\
\circ & \circ & 1 & 1 & \circ & \circ & \circ & \circ & 1 & 1 & 10 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 1 & \circ & 11 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 1 & 12
\end{array}\right]$$

where $\circ$ is used rather than 0 for clarification. The matrix $\underline{\Phi}_1$ is easily generated as shown by Equation 4.3.1.

$$\underline{\Phi}_1 = \left[\begin{array}{cccc|cccccc|cccc|l}
1 & 1 & \circ & \circ & & & & & & & & & & 1 \\
\circ & \circ & \circ & \circ & & & & & & & & & & 2 \\
\circ & \circ & 1 & \circ & & & \underline{0} & & & & & \underline{0} & & 3 \\
\circ & \circ & \circ & 1 & & & & & & & & & & 4 \\
\hline
\circ & \circ & \circ & \circ & 1 & 1 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 5 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 6 \\
1 & \circ & \circ & \circ & 1 & \circ & 1 & \circ & \circ & \circ & 1 & \circ & \circ & \circ & 7 \\
\circ & 1 & \circ & \circ & \circ & 1 & \circ & 1 & \circ & \circ & \circ & 1 & \circ & \circ & 8 \\
\circ & \circ & \circ & \circ & \circ & \circ & 1 & 1 & \circ & \circ & \circ & \circ & \circ & \circ & 9 \\
\circ & \circ & 1 & 1 & \circ & \circ & \circ & \circ & 1 & 1 & \circ & \circ & 1 & 1 & 10 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 1 & \circ & \circ & \circ & \circ & \circ & 11 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 1 & \circ & \circ & \circ & \circ & 12 \\
\hline
& & & & & & & & & & 1 & 1 & \circ & \circ & 13 \\
& & & & & & & & & & \circ & \circ & \circ & \circ & 14 \\
& & \underline{0} & & & & \underline{0} & & & & \circ & \circ & 1 & \circ & 15 \\
& & & & & & & & & & \circ & \circ & \circ & 1 & 16
\end{array}\right]$$

As stated in Section 4.1, if there exists a vector $\overline{w}_1$ such that

$$\underline{A}_1'\overline{w}_1^T = \left[\begin{array}{c|c} \underline{A}_o & \overline{\theta}_1^T \\ \hline \underline{A}_P & \end{array}\right] \overline{w}_1^T > \overline{0}^T \quad ,$$

then it is necessary for $\overline{\theta}_1$ to satisfy

$$\underline{\Phi}_1^T \overline{\theta}_1^T \gtrless \overline{0}^T \qquad (4.3.2)$$

where $\overline{\theta}_1$ contains the components of a redundant TLU output vector $\overline{c}_i$.

4.4 Inequality Constraints on $\overline{c}_1$ for $\gamma = 1$. Assuming that one redundant TLU is sufficient to correct for single, type μ errors in the first layer of a TLU network, the vector $\overline{\theta}_1$ must satisfy Inequality 4.3.2 and certain realizability constraints.

Consider the matrix $\underline{A}_1'$ in Equation 3.5.5 written in more detail as shown in Equation 4.4.1 where $\overline{a}_i^P$ is defined in Section 3.4 and $\overline{a}_i^{(j)}$ is the same as $\overline{a}_i^P$ except that the jth element is replaced by μ if $i \leq m^+$ and by $-\mu$ if $i > m^+$. The last column of $\underline{A}_1'$ forces certain elements of $\overline{\theta}_1$ to be the same since $\overline{\theta}_1 = (c_1,\ldots,c_m,c_1,\ldots,c_m,\ldots,c_1,\ldots,c_m,\mu,\ldots,\mu,-\mu,\ldots,-\mu)$. In effect the realizability constraints are partially applied by this restriction. There is a simple linear transformation from

$$[\overline{c}_1 \mid \mu] = (c_1, c_2, \ldots, c_m, \mu)$$

to $\overline{\theta}_1$; that is,

$$\overline{\theta}_1 = [\overline{c}_1 \mid \mu]\, \underline{T}_{11}$$

where

$$\underset{\sim}{T}_{11} = [\underbrace{\underset{\sim}{T}_p \mid \underset{\sim}{T}_p \mid \cdots \mid \underset{\sim}{T}_p}_{(\nu+1)m \text{ columns}} \mid \underbrace{\underset{\sim}{T}_\mu}_{m \text{ columns}}],$$

$$\underset{\sim}{A}_1^{\nu} = [\underset{\sim}{A}_1 \mid \overline{\theta}_1^T] = \left[\begin{array}{c|c}
\underset{\sim}{a}_1^p & c_1 \\
\underset{\sim}{a}_2^p & c_2 \\
\vdots & \vdots \\
\underset{\sim}{a}_m^p & c_m \\
\hline
\underset{\sim}{a}_1^{(1)} & c_1 \\
\underset{\sim}{a}_2^{(1)} & c_2 \\
\vdots & \vdots \\
\underset{\sim}{a}_m^{(1)} & c_m \\
\hline
\cdots\cdots\cdots & \cdots\cdots\cdots \\
\hline
\underset{\sim}{a}_1^{(\nu)} & c_1 \\
\underset{\sim}{a}_2^{(\nu)} & c_2 \\
\vdots & \vdots \\
\underset{\sim}{a}_m^{(\nu)} & c_m \\
\hline
\underset{\sim}{a}_1^p & \mu \\
\vdots & \vdots \\
\underset{\sim}{a}_{m^+}^p & \mu \\
\underset{\sim}{a}_{(m^+\!+1)}^p & -\mu \\
\vdots & \vdots \\
\underset{\sim}{a}_m^p & -\mu
\end{array}\right] \qquad (4.4.1)$$

$$\underline{T}_P = \begin{bmatrix} 1 & 0 & \cdot & \cdot & \cdot & 0 & 0 \\ 0 & 1 & \cdot & \cdot & \cdot & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot & \cdot & 0 & 1 \\ 0 & 0 & \cdot & \cdot & \cdot & 0 & 0 \end{bmatrix}_{(m+1)xm} ,$$

and

$$\underline{T}_\mu = \begin{array}{c} \overset{m^+ \text{ columns}}{\qquad\qquad} \qquad\qquad \overset{m^- \text{ columns}}{\qquad\qquad} \\ \left[\begin{array}{cccccc|cccccc} 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot & \cdot & 0 & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ 1 & 1 & \cdot & \cdot & \cdot & 1 & -1 & -1 & \cdot & \cdot & \cdot & -1 \end{array}\right]_{(m+1)xm.} \end{array}$$

Referring to Inequality 4.3.2 in Example 4.3.1 the following simplifications can be made:

$$\underline{\Phi}_1^T \underline{\theta}_1^T = \underline{\Phi}_1^T \underline{T}_{11}^T [\bar{c}_1 \mid \mu]^T \gtrless \bar{0}^T$$

or

$$[\bar{c}_1 \mid \mu]\underline{T}_{11} \underline{\Phi}_1 \gtrless \bar{0} \quad .$$

Defining

$$\underline{D}_{11} = \underline{T}_{11} \underline{\Phi}_1$$

the inequality constraints, as modified by the above realizability constraints on $\bar{\theta}_1$, simplify to

$$[\bar{c}_1 \mid \mu]\underline{D}_{11} \gtrless \bar{0} \qquad (4.4.2)$$

where $\mu = 1$ in the example. A given $\bar{c}_1$ satisfying the above inequality

has no assurance of being realizable by a TLU. A computer search must be performed to test those $\bar{c}_1$'s satisfying the above inequality to see if one of them corresponds to a realizable $\bar{c}'_1$. This search is performed by the program of Section D.3.

The terminology introduced here is consistent with the general case of $\gamma > 1$. The last γ columns of $\underline{A}'_\gamma$ are similarly related to the vectors $\bar{c}_2, \bar{c}_3, \ldots, \bar{c}_\gamma$. Let the transpose of these columns be defined as

$$\bar{c}^k_\gamma = (c_1,\ldots,c_m,\ldots,c_1,\ldots,c_m,\mu,\ldots,\mu,-\mu,\ldots,-\mu,c_1,\ldots,c_m,\ldots,c_1,\ldots,c), \quad k = 1, 2, \ldots, \gamma,$$

where the μ's occur in the positions as shown in the $(\nu+1+k)$th column of $\underline{A}'_\gamma$ in Equation 3.5.7. For example, $\bar{\theta}_1$ used above is equal to $\bar{c}^1_1$. In general

$$\bar{c}^k_\gamma = [\bar{c}_\gamma \mid \mu]\underline{T}_{\gamma_k}$$

where

$$\underline{T}_{\gamma_k} = [\underline{T}_P \mid \cdots \mid \underline{T}_P \mid \underline{T}_\mu \mid \underline{T}_P \mid \cdots \mid \underline{T}_P] .$$

The partition containing $\underline{T}_\mu$ corresponds to the location of the μ's in $\bar{c}^k_\gamma$. This notation becomes of considerable value when γ is large because the extension from $\underline{\Phi}^*_o$ to $\underline{\Phi}_i$ reduces to a direct extension of the smaller matrix

$$\underline{D}_{oo} = \underline{T}_{oo}\,\underline{\Phi}_o = [\underline{T}_P \mid \underline{T}_P \mid \cdots \mid \underline{T}_P] \begin{bmatrix} \underline{\Phi}_P & \underline{0} \\ \underline{\Phi}_{R_1} & \underline{\Phi}_{R_2} \end{bmatrix}$$

to $\underline{D}_{ik}$ for $i = 1, 2, \ldots, \gamma$ and $k = 1, 2, \ldots, i$.

In general $\underline{D}_{ik}$ is given by

$$\underline{D}_{ik} = \underline{T}_{ik}\,\underline{\Phi}_i \ ,$$

$$\underline{D}_{ik} = \left[\begin{array}{c|c|c|c|c|c|c|c|c} 1 & & 2 & & 3 & 4 & & k+2 & & i+2 \\ \underline{T}_P & \underline{T}_P & \cdots & \underline{T}_P & \underline{T}_P & \underline{T}_P & \cdots & \underline{T}_\mu & \cdots & \underline{T}_P \end{array}\right].$$

$$\left[\begin{array}{c|c|c|c|c|c|c} \underline{\Phi}_P & \underline{0} & \underline{0} & \underline{0} & \cdots & \underline{0} & 1 \\ \hline \underline{\Phi}_{R_1} & \underline{\Phi}_{R_2} & \underline{\Phi}_{R_1} & \underline{\Phi}_{R_1} & \cdots & \underline{\Phi}_{R_1} & 2 \\ \hline \underline{0} & \underline{0} & \underline{\Phi}_P & \underline{0} & \cdots & \underline{0} & 3 \\ \hline \underline{0} & \underline{0} & \underline{0} & \underline{\Phi}_P & \cdots & \underline{0} & 4 \\ \hline \cdots & \cdots & \cdots & & \cdots & \cdots & \\ \hline \underline{0} & \underline{0} & \underline{0} & \underline{0} & \cdots & \underline{\Phi}_P & i+2 \end{array}\right],$$

or by

$$\underline{D}_{ik} = \left[\begin{array}{c|c|c|c|c|c|c} 1 & 2 & 3 & & k+2 & & i+2 \\ \underline{R}_1 & \underline{R}_2 & \underline{R}_1 & \cdots & \underline{R}_\mu & \cdots & \underline{R}_1 \end{array}\right].$$

The partitions are numbered correspondingly for clarification. The matrix $\underline{R}_1$ is produced by any vertical partition of $\underline{\Phi}_i$ which contains $\underline{\Phi}_{R_1}$ and a $\underline{\Phi}_P$ which is not in the horizontal partition corresponding to $\underline{T}_\mu$. The case thus excluded is written as $\underline{R}_\mu$. The matrix $\underline{R}_2$ is produced by the verticle partition containing $\underline{\Phi}_{R_2}$. For purposes of simplification let

$$\underline{D}'_{ik} = \left[\begin{array}{c|c|c|c|c} \underline{R}_P & \underline{R}_1 & \cdots & \underline{R}_\mu & \cdots & \underline{R}_1 \end{array}\right]$$

where $\underline{R}_P$ is constructed from

$$\left[\begin{array}{c|c} \underline{R}_1 & \underline{R}_2 \end{array}\right]$$

by removing all redundant columns. Also define $\underline{D}''_{11}$ as the matrix constructed from $\underline{D}'_{11}$ by removing all of its redundant columns. $D''_{\gamma\gamma}$ is defined in Section 4.5 for $\gamma = 2$ and in Section 4.6 for $\gamma > 1$.

Example 4.4.1. In Example 4.3.1 the inequality constraints of Inequality 4.3.2 on $\overline{\theta}_1 = \overline{c}_1^{-1}$ for a particular $\underline{A}_o$ are derived. Using the notation of this section the size of the matrices in Inequality 4.3.2 can be reduced. Inequality 4.3.2 is

$$\underline{\Phi}_1^T \overline{\theta}_1^T \gtrless \overline{0}^T$$

and reduces to

$$[\overline{c}_1 \mid 1]\underline{D}_{11} \gtrless \overline{0} \qquad (4.4.3)$$

as in Inequality 4.4.2, where

$$\underline{D}_{11} = \underline{T}_{11}\,\underline{\Phi}_1 = [\underline{R}_1 \mid \underline{R}_2 \mid \underline{R}_\mu]$$

$$\underline{D}_{11} = \left[\begin{array}{cccc|cccccc|cccc} 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \end{array}\right].$$

There are several redundant columns in $\underline{D}_{11}$ above which serve no purpose when $\underline{D}_{11}$ is used in Inequality 4.4.3. Therefore, when searching for a realizable $\overline{c}_1$ which satisfies the inequality, $\underline{D}_{11}$ can be reduced to $\underline{D}''_{11}$ by removing all redundant columns.

$$\underline{D}''_{11} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix} .$$

Then $\bar{c}_1$ must satisfy

$$[\bar{c}_1 \mid 1]\underline{D}''_{11} \gtrless \bar{0} \qquad (4.4.4)$$

and $\bar{c}'_1$ must be realizable with a TLU in order for one redundant TLU to correct for all single, type μ, first layer errors. It turns out that no such $\bar{c}_1$ exists in this case as can be determined by the computer program of Section D.3.

If no suitable $\bar{c}_1$ exists which satisfies Inequality 4.4.4, then the "best" $\bar{c}_1$ must be selected and a search must be made for a $\bar{c}_2$ which satisfies a new set of inequality constraints. The selection of "best" $\bar{c}_1$ and the development of the constraints on $\bar{c}_2$ is the topic of the next section.

4.5 Inequality Constraints on $\bar{c}_2$ given $\bar{c}_1$. Recalling the basic reason for $\bar{c}_1$, it can be seen that a $\bar{c}_1$ is desired such that every $\bar{\phi}^1$, which is a nonnegative, linear combination of extremal vectors $\bar{\phi}^1_k$, $k = 1, 2, \ldots, s_1$, must result in

$$\bar{c}^1_1(\bar{\phi}^1)^T \neq 0 \quad .$$

In other words, $\bar{c}^1_1$ should satisfy

$$\bar{c}_1^1\left[\sum_{k=1}^{s_1}\lambda_k(\bar{\phi}_k^1)^T\right] \neq 0\ ,\quad \lambda_k \geq 0\ ,$$

for all λ_k. This inequality reduces to

$$[\bar{c}_1 \mid \mu]\underline{D}_{11}\,\bar{\lambda}^T \neq 0\ ,\quad \bar{\lambda} \geq \bar{0}\ ,$$

or to

$$[\bar{c}_1 \mid \mu]\underline{D}'_{11}\,\bar{\lambda}^T \neq 0\ ,\quad \bar{\lambda} \geq \bar{0}\ , \tag{4.5.1}$$

where $\bar{\lambda}$ is the vector whose elements are the nonnegative scalors above. Since the elements of $\underline{D}'_{11}$ correspond to the extremal vectors in $\underline{\Phi}_1$, Inequality 4.5.1 is satisfied when

$$\bar{c}_1^1\,\underline{\Phi}_1 = [\bar{c}_1 \mid \mu]\underline{D}'_{11} \gtrless \bar{0}\ . \tag{4.5.2}$$

If there exists no realizable $\bar{c}_1^1$ such that Inequality 4.5.2 is satisfied then a $\bar{c}_1^2$ must be selected which is "best" according to some criterion, and a search must be made to see if there exists a suitable $\bar{c}_2^2$ which completes the task started by $\bar{c}_1^2$.

Assuming that $\gamma = 2$, $\bar{c}_1^2$ and $\bar{c}_2^2$ must be such that for every nonnegative, linear combination of $\bar{\phi}_k^2$'s,

$$\bar{\phi}^2 = \sum_{k=1}^{s_2}\lambda_k\bar{\phi}_k^2\ ,\quad \lambda_k \geq 0,$$

$\bar{c}_1^2$ and $\bar{c}_2^2$ satisfy

$$\begin{bmatrix}\bar{c}_1^2\\ \\ \bar{c}_2^2\end{bmatrix}(\bar{\phi}^2)^T = \begin{bmatrix}\bar{c}_1^2\\ \\ \bar{c}_2^2\end{bmatrix}\left[\sum_{k=1}^{s_2}\lambda_k(\bar{\phi}_k^2)^T\right] \neq \begin{bmatrix}0\\ \\ 0\end{bmatrix}\ ,\ \lambda_k \geq 0, \tag{4.5.3}$$

where $\bar{\phi}_k^2 \varepsilon S_\phi^2$. This inequality reduces to

$$\left[\begin{array}{c|c} [\bar{c}_1 & \mu]\underline{D}_{21} \\ \hline [\bar{c}_2 & \mu]\underline{D}_{22} \end{array}\right] \bar{\lambda}^T \neq \bar{0}^T \ , \ \bar{\lambda} \geq \bar{0} \ ,$$

or to

$$\left[\begin{array}{c|c} [\bar{c}_1 & \mu]\underline{D}'_{21} \\ \hline [\bar{c}_2 & \mu]\underline{D}'_{22} \end{array}\right] \bar{\lambda}^T \neq \bar{0}^T \ , \ \bar{\lambda} \geq \bar{0} \ ,$$

where $\bar{\lambda}$ is conformable. The first inequality in this system of two inequalities is

$$[\bar{c}_1 \mid \mu][\underline{R}_P \mid \underline{R}_\mu \mid \underline{R}_1] \bar{\lambda}^T \neq 0 \ , \ \bar{\lambda} \geq \bar{0} \ , \qquad (4.5.4)$$

and is equivalent to

$$[\bar{c}_1 \mid \mu][\underline{R}_P \mid \underline{R}_\mu] \bar{\lambda}_1^T = [\bar{c}_1 \mid \mu]\underline{D}'_{11} \bar{\lambda}_1^T \neq 0 \ , \ \bar{\lambda}_1 \geq \bar{0} \ ,$$

which is the same as Inequality 4.5.1. Since there exists no $\bar{c}_1$ such that Inequality 4.5.1 is satisfied, there exists none such that Inequality 4.5.4 is satisfied.

In order to reduce the task which $\bar{c}_2$ is expected to perform, $\bar{c}_1$ must be selected judiciously. Obviously the selection of $\bar{c}_1$ has an effect on the system of linear inequalities which $\bar{c}_2$ must satisfy. Before defining the "best" $\bar{c}_1$, some notation must be introduced. Let the matrix

$$\underline{\Lambda}_{21} = \begin{bmatrix} \overline{\lambda}_1 \\ \overline{\lambda}_2 \\ \vdots \\ \overline{\lambda}_{s_{21}} \end{bmatrix}$$

be defined as the matrix whose rows are extremal vectors for the set of solutions $\overline{\lambda}$ to

$$[\overline{c}_1 \mid \mu]\underline{D}'_{21}\,\overline{\lambda}^T = 0 \ , \quad \overline{\lambda} \neq \overline{0}, \quad \overline{\lambda} \geq 0 \ .$$

It is also necessary to point out that the vector $\overline{c}_1$ is trivial if it is

$$\overline{c}_1 = (\underbrace{1, 1, \ldots, 1}_{m^+ \text{ elements}} \mid \underbrace{-1, -1, \ldots, -1}_{m^- \text{ elements}} \mid \mu)$$

or

$$\overline{c}_1 = (-1, -1, \ldots, -1 \mid 1, 1, \ldots, 1 \mid \mu)$$

corresponding to

$$\overline{c}'_1 = (1, 1, \ldots, 1 \mid \mu)$$

or

$$\overline{c}'_1 = (-1, -1, \ldots, -1 \mid \mu) \ .$$

The addition of a redundant TLU with this output vector amounts to nothing more than adding another constant input to the second layer TLU $m_{\nu+1}$.

Definition 2.5.1. The best $\overline{c}_1$ is that nontrivial $\overline{c}_1$, corresponding

to a realizable $\bar{c}_1'$, for which the matrix $\underline{\Lambda}_{21}$ results in the least number of unique columns in the matrix

$$\underline{D}_{22}' \underline{\Lambda}_{21}^T .$$

Notice that a column which is equal to a positive multiple of another column is not unique because it applies the same inequality constraint as the other column. The computer program of Section D.2 computes $\bar{c}_1$ from input data consisting of the input patterns $\bar{y}_1'$, $\bar{y}_2'$, ..., $\bar{y}_m'$, the value of μ, and the matrices $\underline{D}_{21}'$ and $\underline{D}_{22}'$.

Given a best $\bar{c}_1$, the vector $\bar{c}_2$ must satisfy

$$[\bar{c}_2 \mid \mu]\underline{D}_{22}' \underline{\Lambda}_{21}^T \gtrless \bar{0}$$

where the unique columns of the matrix $\underline{D}_{22}' \underline{\Lambda}_{21}^T$ are computed by the program of Section D.2 while searching for the best $\bar{c}_1$. Call the matrix $\underline{D}_{22}''$ the matrix whose columns are these unique columns, then $\bar{c}_2$ must satisfy

$$[\bar{c}_2 \mid \mu]\underline{D}_{22}'' \gtrless \bar{0} \quad .$$

The computer program in Section D.3 can be used to search for $\bar{c}_2$ and determine whether or not one exists. If one does exist, then $\gamma = 2$ as assumed.

Example 4.5.1. Continuing the running example in Examples 4.3.1 and 4.4.1 the problem at this point is to find the best $\bar{c}_1$ and a $\bar{c}_2$. Since there exists no $\bar{c}_1$ such that Inequality 4.4.3 is satisfied, a realizable $\bar{c}_1$ is selected such that the matrix $\underline{D}_{22}' \underline{\Lambda}_{21}^T$ has the least number of unique columns. Using

$$[\bar{c}_1 \mid 1]\underline{D}'_{21}\,\bar{\lambda}^T = 0\ ,\quad \bar{\lambda} \neq \bar{0}\ ,\quad \bar{\lambda} \geq \bar{0}$$

or

$$[\bar{c}_1 \mid 1]\left[\begin{array}{cccc|cccc|cccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \end{array}\right]\bar{\lambda}^T = \bar{0} \qquad (4.5.5)$$

$$\bar{\lambda} \neq \bar{0}\ ,\ \bar{\lambda} \geq \bar{0}\ ,$$

the computer search routine of Section D.2 calculates

$$\bar{c}_1 = (1,\ -1,\ -1,\ -1)$$

for the realizable $\bar{c}'_1$ which is best according to Definition 4.5.1.

For the given value of $\bar{c}_1$, Equation 4.5.5 results in

$$\underline{\Lambda}_{21} = \left[\begin{array}{c} \bar{\lambda}_1 \\ \bar{\lambda}_2 \\ \bar{\lambda}_3 \\ \bar{\lambda}_4 \\ \bar{\lambda}_5 \\ \bar{\lambda}_6 \end{array}\right] = \left[\begin{array}{cccc|cccc|cccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right]$$

through the use of the technique of Appendix A. If γ is actually equal to two, then $\bar{c}_2$ must satisfy

$$[\bar{c}_2 \mid 1]\underline{D}'_{22}\,\underline{\Lambda}_{21}^T \gtrless \bar{0} \qquad (4.5.6)$$

or

$$[\bar{c}_2 \mid 1]\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \gtrless \bar{0}$$

Replacing $\underline{D}''_{22}\,\underline{\Lambda}_{21}^{T}$ by $\underline{D}''_{22}$, the computer search routine of Section D.3 searches for a realizable $\bar{c}_2$ such that

$$[\bar{c}_2 \mid 1]\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \gtrless \bar{0} \quad . \qquad (4.5.7)$$

This search results in

$$\bar{c}_2 = (1, -1, 1, 1)$$

which satisfies Inequality 4.5.7. Therefore $\gamma = 2$ and $\bar{c}_1$ and $\bar{c}_2$ are as given above for single, type 1 errors in the first layer of a TLU network which produces $\underline{A}_P$ given in Example 4.2.1. Notice that the actual output vectors for the redundant TLU's $m_1^{(R)}$ and $m_2^{(R)}$ are given by

$$\bar{c}'_1 = (1, -1, 1, 1)$$

and

$$\bar{c}'_2 = (1, -1, -1, -1) \quad .$$

The result of this procedure is that the matrix $\underline{A}'_2$ written as in

Equation 4.5.8, forms a consistent system of linear inequalities

$$\underline{A}_2^{\prime}\overline{w}_2^T > \overline{0}^T .$$

This consistency is verified with the technique of Appendix A by showing that there are no extremal vectors for the set of solutions to

$$(\underline{A}_2^{\prime})^T \overline{\phi}^T = \overline{0} , \quad \overline{\phi} \neq \overline{0} , \quad \overline{\phi} \geq \overline{0} ,$$

where $\overline{c}_1$ and $\overline{c}_2$ are given. The resulting redundant TLU network is shown in Figure 4.5.1.

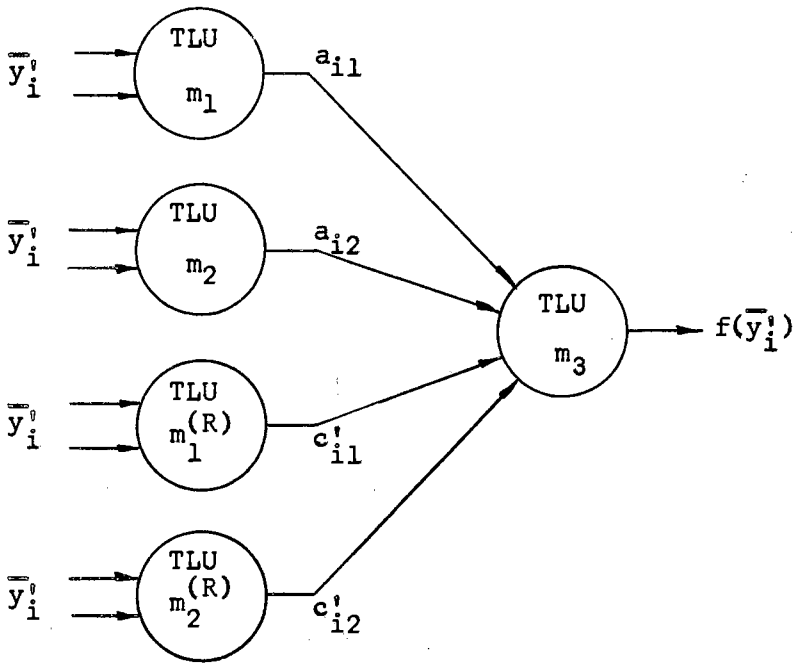


Figure 4.5.1. Redundant TLU Network for Example 4.5.1

If no realizable $\bar{c}_2$ exists such that Equation 4.5.7 has no solutions, then γ must be greater than two. The procedures of the previous two sections can be generalized to investigate all values of γ in a systematic manner.

$$\underline{A}'_2 = [\underline{A}_2 \mid (\bar{c}_1^2)^T \mid (\bar{c}_2^2)^T] = \left[\begin{array}{ccccc} 1 & 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & -1 & -1 \\ 1 & -1 & -1 & -1 & 1 \\ 1 & -1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 1 \\ -1 & -1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 & 1 \\ 1 & -1 & -1 & -1 & 1 \\ 1 & -1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & -1 & 1 \\ 1 & -1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & -1 & -1 & -1 \end{array}\right] \qquad (4.5.8)$$

4.6 Inequality Constraints on $\bar{c}_\gamma$ given $\bar{c}_{\gamma-1}, \bar{c}_{\gamma-2}, \ldots, \bar{c}_1$. For $\gamma > 2$ the procedure of Section 4.5 becomes more complicated. The

notation used there is extended to the general case here and additional notation is introduced. Keep in mind that the value of γ is an assumed value; and, if the algorithm does not find γ vectors $\bar{c}_i$ which correct single type μ failures, then the algorithm must be repeated with a larger value of γ.

In general the matrix $\underline{\Lambda}_{\gamma i}$ has as its $s_{\gamma i}$ rows the extremal vectors $\bar{\lambda}_k$ for the solutions to

$$\left[\begin{array}{c|c} [\bar{c}_1 & \mu]\underline{D}'_{\gamma 1} \\ \hline [\bar{c}_2 & \mu]\underline{D}'_{\gamma 2} \\ \hline \vdots & \\ \hline [\bar{c}_i & \mu]\underline{D}'_{\gamma i} \end{array}\right] \bar{\lambda}^T = \left[\begin{array}{c} 0 \\ 0 \\ \vdots \\ 0 \end{array}\right] \qquad \begin{array}{l} \bar{\lambda} \neq \bar{0},\ \bar{\lambda} \geq \bar{0}, \\ i = 1, 2, \ldots, \gamma-1. \end{array}$$

Using the matrix $\underline{\Lambda}_{\gamma i}$, the problem here is to extend Definition 4.5.1 to aid in the selection of $\bar{c}_i$, $2 \leq i \leq \gamma-1$, such that the task remaining for $\bar{c}_{i+1}$ is reduced. If it happened to be possible for the algorithm to terminate on $\bar{c}_{i+1}$, then $\bar{c}_{i+1}$ would have to satisfy

$$[\bar{c}_{i+1} \mid \mu]\underline{D}'_{\gamma(i+1)}\underline{\Lambda}^T_{\gamma(i-1)}\bar{\omega}^T \begin{matrix} > \\ < \end{matrix} 0$$

for every $\bar{\omega}$ which is a solution to

$$[\bar{c}_i \mid \mu]\underline{D}'_{\gamma i}\underline{\Lambda}^T_{\gamma(i-1)}\bar{\omega}^T = 0\ , \quad \bar{\omega} \neq 0,\ \ \bar{\omega} \geq \bar{0}\ .$$

In order to explain this, consider every positive linear combination

$$\bar{\lambda}^T = \underline{\Lambda}^T_{\gamma(i-1)}\bar{\omega}^T$$

of extremal vectors for the solutions $\bar{\lambda}$ to

$$\left[\begin{array}{c|c} [\bar{c}_1 & \mu]\underline{D}'_{\gamma 1} \\ \hline [\bar{c}_2 & \mu]\underline{D}'_{\gamma 2} \\ \hline \vdots & \\ \hline [\bar{c}_{i-1} & \mu]\underline{D}'_{\gamma(i-1)} \end{array}\right] \bar{\lambda}^T = 0, \ \bar{\lambda} \neq \bar{0}, \ \bar{\lambda} \geq \bar{0},$$

for which the best $\bar{c}_i$ permits the equation

$$[\bar{c}_i \mid \mu]\underline{D}'_{\gamma i}\bar{\lambda}^T = [\bar{c}_i \mid \mu]\underline{D}'_{\gamma i}\underline{\Lambda}^T_{\gamma(\gamma-1)}\bar{\omega}^T = 0, \ \bar{\omega} \neq \bar{0}, \ \bar{\omega} \geq \bar{0}, \quad (4.6.1)$$

to be true. Then $\bar{c}_{i+1}$ must satisfy the inequality

$$[\bar{c}_{i+1} \mid \mu]\underline{D}'_{\gamma(i+1)}\underline{\Lambda}^T_{\gamma(i-1)}\bar{\omega}^T \gtrless 0$$

for every $\bar{\omega}$ satisfying Equation 4.6.1. The $\bar{\omega}$'s satisfying Equation 4.6.1 form a convex polyhedral cone; therefore, each can be expressed as a positive, linear combination of extremal vectors $\bar{\omega}_1, \bar{\omega}_2, \ldots, \bar{\omega}_{s_{\gamma i}}$. Furthermore, if the matrix $\underline{\Omega}_{\gamma i}$ contains as its rows these extremal vectors, then it is sufficient for $\bar{c}_{i+1}$ to satisfy

$$[\bar{c}_{i+1} \mid \mu]\underline{D}'_{\gamma(i+1)}\underline{\Lambda}^T_{\gamma(i-1)}\underline{\Omega}^T_{\gamma i} \gtrless 0 \ . \quad (4.6.2)$$

It is very significant that the matrix

$$[\bar{c}_i \mid \mu]\underline{D}'_{\gamma i}\underline{\Lambda}^T_{\gamma i}$$

contains only one row. This permits extremal vectors for the solutions $\bar{\omega}$ to Equation 4.6.1 to be found very simply as shown in Appendix C. This is incorporated in the program of Section D.2, which computes $\bar{c}_i$ based on the definitions below.

Obviously, if $\gamma > i$, then the algorithm does not terminate with $\bar{c}_i$. Therefore some criterion must be established such that the task performed

by $\bar{c}_i$, $2 \leq i \leq \gamma-1$, reduces the task expected of $\bar{c}_{i+1}$.

Definition 4.6.1. The best $\bar{c}_i$, $2 \leq i \leq \gamma-1$, is a realizable, nontrivial $\bar{c}_i$ for which the matrix

$$\underline{D}'_{\gamma(i+1)}\underline{\Lambda}^T_{\gamma(i-1)}\underline{\Omega}^T_{\gamma i}$$

has the least number of unique columns. Similarly Definition 4.5.1 can be extended for $\gamma > 2$ to the following.

Definition 4.6.2. The best $\bar{c}_1$ is a realizable, nontrivial $\bar{c}_1$ for which the matrix

$$\underline{D}'_{\gamma 2}\underline{\Lambda}^T_{\gamma 1}$$

has the least number of unique columns.

Notice that in Definitions 4.5.1, 4.6.1, and 4.6.2, no claim of optimality is made. These definitions are made on an intuitive basis. A geometrical description of these definitions and the rest of the algorithm is given in Section 4.8.

Based on the above definitions and the theory of Appendices A and C, the computer program of Section D.2 computes the best $\bar{c}_i$. For $2 \leq i \leq \gamma-1$ the input data for this program consists of the input pattern vectors $\bar{y}'_1, \bar{y}'_2, \ldots, \bar{y}'_m$, the value of μ, and the matrices $\underline{D}'_{\gamma i}\underline{\Lambda}^T_{\gamma(i-1)}$ and $\underline{D}'_{\gamma(i+1)}\underline{\Lambda}^T_{\gamma(i-1)}$.

Given $\bar{c}_{\gamma-1}, \bar{c}_{\gamma-2}, \ldots, \bar{c}_1$; $\bar{c}_\gamma$ must satisfy

$$[\bar{c}_\gamma \mid \mu]\underline{D}'_{\gamma\gamma}\underline{\Lambda}^T_{\gamma(\gamma-1)} \gtrless \bar{0} \qquad (4.6.3)$$

and $\bar{c}'_\gamma$ must be realizable. The search for $\bar{c}_\gamma$ can be simplified by defining $\underline{D}''_{\gamma\gamma}$ to be the matrix formed by removing the redundant columns

from $\underline{D}'_{\gamma\gamma}\underline{\Lambda}^{T}_{\gamma(\gamma-1)}$. As in the case for $\gamma = 2$ in Section 4.5, the computer program of Section D.3 can be used to search for a $\bar{c}_\gamma$ which satisfies Inequality 4.6.3. If there exists such a $\bar{c}_2$, then the assumed value of γ is correct; if not, then the algorithm must be repeated with a larger value of γ.

The flow-chart in Figure 4.6.1 illustrates the algorithm and its use is illustrated by Example 4.7.1.

4.7 Extension of the Theory to Consider Both Failure Types Simultaneously. The theory developed here and in Chapter II is flexible to the extent that the consideration of the occurrence of both type -1 and type +1 failures is permitted. This extension can be performed by adding extra rows to $\underline{A}'_i$ and by making the corresponding alterations in $\underline{\Phi}^{*}_{o}$, $\underline{\Phi}_{o}$, $\underline{D}'_{ik}$, and $\bar{c}_i$. There is no change in the application of the algorithm illustrated in Figure 4.6.1 after the above alterations are made.

Notice that in Equation 3.5.7 the partition containing $\underline{A}_\gamma$ is

$$\underline{A}_\gamma = \begin{bmatrix} \underline{A}_P & \text{m rows} \\ \underline{A}_R & \nu\text{m rows} \\ \underline{A}_P & \text{m rows} \\ \vdots & \vdots \quad \gamma\text{m rows} \\ \underline{A}_P & \text{m rows} \end{bmatrix}.$$

The most significant alteration in $\underline{A}_\gamma$ is in the partition $\underline{A}_R$. $\underline{A}_R$ consists of ν sets of m rows as shown in Equation 3.5.2 and the alteration in $\underline{A}_R$ amounts to adding ν additional sets of m rows as in

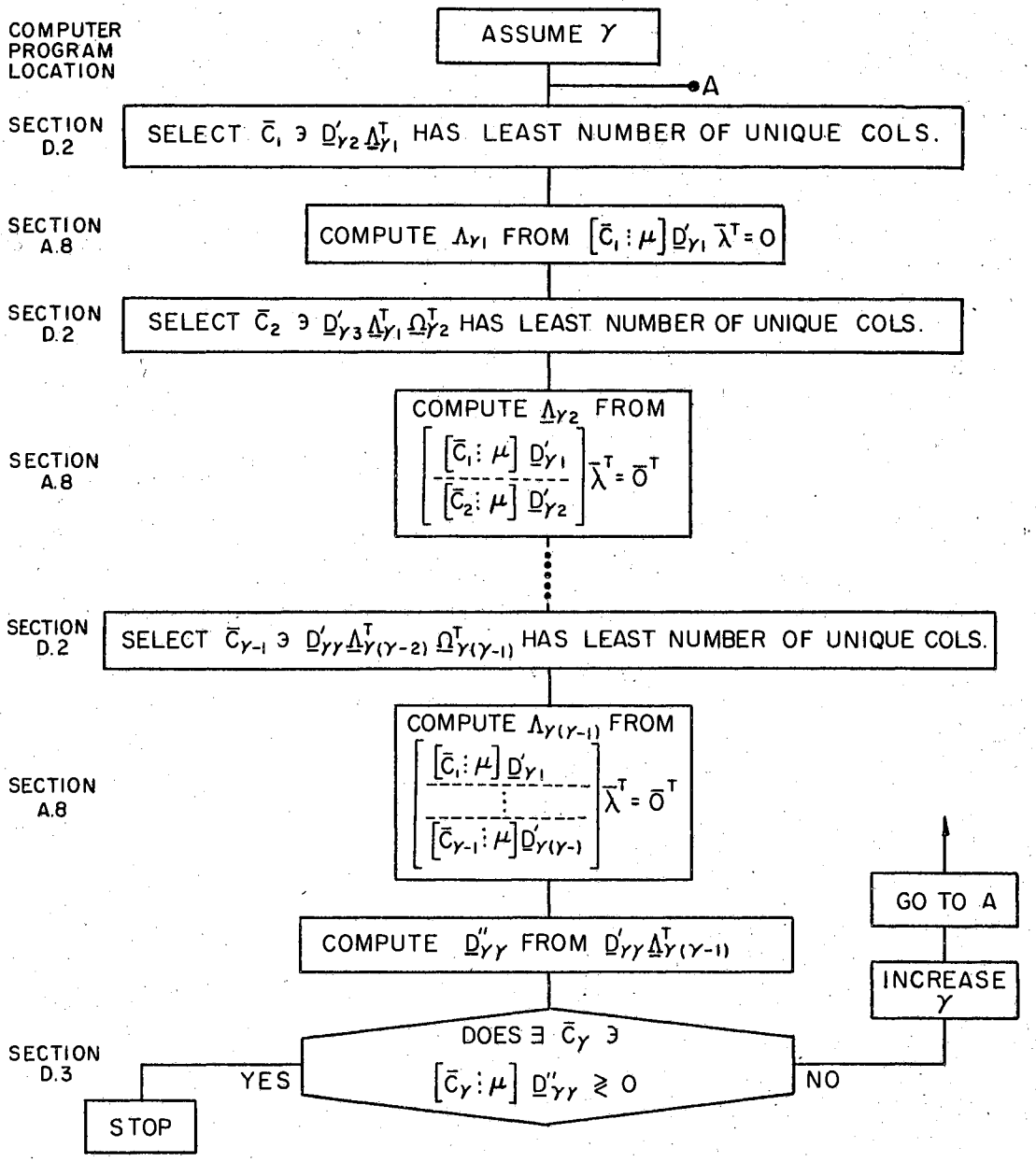


Figure 4.6.1. Flow-Chart for Synthesis Algorithm

$$\underline{A}_R^a = \begin{bmatrix} \underline{A}_R \\ \hline \underline{\bar{A}}_R \end{bmatrix}$$

where $\underline{\bar{A}}_R$ is the same as $\underline{A}_R$ except that μ is replaced by $-\mu$. The remaining alteration in $\underline{A}_\gamma$ is the addition of γ partitions $\underline{A}_P$ as in

$$\underline{A}_\gamma^a = \begin{bmatrix} \underline{A}_P \\ \hline \underline{A}_R^a \\ \hline \underline{A}_P \\ \hline \vdots \\ \hline \underline{A}_P \\ \hline \underline{A}_P \\ \hline \vdots \\ \hline \underline{A}_P \end{bmatrix} \begin{matrix} m \text{ rows} \\ 2\nu m \text{ rows} \\ \\ \gamma m \text{ rows} \\ \\ \\ \gamma m \text{ rows} \\ \\ \end{matrix} .$$

In order to consider both failure types in the redundant TLU's, the altered form of the matrix $\underline{A}_\gamma'$ must be written as shown in Equation 4.7.1. The partitioning shown means that the matrix $\underline{\Phi}_\gamma^a$, which is the altered form of $\underline{\Phi}_\gamma$, has the same form as $\underline{\Phi}_\gamma$, but that the matrices $\underline{\Phi}_P$ in the diagonal partitions (except the upper left partition) occur 2γ times rather than γ. Also the altered partitions $\underline{\Phi}_{R_1}^a$ and $\underline{\Phi}_{R_2}^a$ contain twice as many rows and, possibly, more columns. The matrices $\underline{\Phi}_i^a$ for $i > 0$ are of no importance in the mechanics of the algorithm; however, the resulting altered matrices $\underline{D}_{ik}^a$ are of importance.

$\underline{D}_{\gamma k}^a$ can be obtained from $\underline{\Phi}_\gamma^a$ by the transformation matrix

$$\underline{T}_{\gamma k}^a = [\underbrace{\underline{T}_P}_{m \text{ cols}} \mid \underline{T}_P \mid \underbrace{\cdots}_{2\nu m \text{ cols}} \mid \underline{T}_P \mid \underline{T}_P \mid \cdots \mid \underbrace{\underline{T}_P}_{2\nu m \text{ cols}} \mid \underline{T}_\mu \mid -\underline{T}_\mu \mid \underline{T}_P \mid \cdots \mid \underline{T}_P]$$

$$
A_{\gamma}^{a'} = \left[\begin{array}{c|c|c|c|c}
\underline{A}_P & \underline{c}_1^T & \underline{c}_2^T & \cdots & \underline{c}_\gamma^T \\
\hline
 & \cdots & \cdots & \cdots & \cdots \\
\hline
\underline{A}_R^a & \underline{c}_1^T & \underline{c}_2^T & \cdots & \underline{c}_\gamma^T \\
\hline
 & \cdots & \cdots & \cdots & \cdots \\
\hline
 & \underline{c}_1^T & \underline{c}_2^T & \cdots & \underline{c}_\gamma^T \\
\hline
 & \mu & & & \\
 & \vdots & & & \\
 & \mu & & & \\
\underline{A}_P & -\mu & \underline{c}_2^T & \cdots & \underline{c}_\gamma^T \\
 & \vdots & & & \\
 & -\mu & & & \\
\hline
 & -\mu & & & \\
 & \vdots & & & \\
 & -\mu & & & \\
\underline{A}_P & \mu & \underline{c}_2^T & \cdots & \underline{c}_\gamma^T \\
 & \vdots & & & \\
 & \mu & & & \\
\hline
\vdots & \vdots & \vdots & \vdots & \vdots \\
\hline
 & & & & \mu \\
 & & & & \vdots \\
 & & & & \mu \\
\underline{A}_P & \underline{c}_1^T & \underline{c}_2^T & \cdots & -\mu \\
 & & & & \vdots \\
 & & & & -\mu \\
\hline
 & & & & -\mu \\
 & & & & \vdots \\
\underline{A}_P & \underline{c}_1^T & \underline{c}_2^T & \cdots & -\mu \\
 & & & & \mu \\
 & & & & \vdots \\
 & & & & \mu
\end{array}\right]
\begin{array}{l}
m \text{ rows} \\
\\
2\nu m \text{ rows} \\
\\
m \text{ rows} \\
\\
m \text{ rows} \\
\\
m \text{ rows} \\
\\
(2\nu+2\gamma+1)\times(\nu+\gamma+1)
\end{array}
\qquad (4.7.1)
$$

where the partition

$$[\underline{T}_{\mu} \mid -\underline{T}_{\mu}]$$

occurs in the columns corresponding to the elements μ in the $(\nu+1+1)$st column of $\underline{A}_{\gamma}^{a'}$.

The only other alteration is in $[\bar{c}_i \mid \mu]$. Since the matrix $\underline{T}_{\gamma k}^{a}$ contains both $\underline{T}_{\mu}$ and $-\underline{T}_{\mu}$, μ must be set equal to one; that is

$$[\bar{c}_i \mid 1]$$

must be used exclusively in the algorithm.

The altered matrix $\underline{D}_{\gamma k}^{a}$ can be simplified to $\underline{D}_{\gamma k}^{a'}$ by removing the redundant columns in the first $(2\nu+1)m$ columns. The result is

$$\underline{D}_{\gamma k}^{a'} = [\underline{R}_{P}^{a} \mid \underline{R}_{1}^{a} \mid \cdots \mid \underline{R}_{\mu}^{a} \mid \cdots \mid \underline{R}_{1}^{a}] \quad .$$

The use of this extension and of the general algorithm in Figure 4.6.1 is illustrated by the following examples.

Example 4.7.1. This example is an extension of the running example appearing previously in Examples 4.3.1, 4.4.1, and 4.5.1. There it is assumed that only type +1 failures can occur. This assumption is consistent with all of the theory with the exception of that developed in this section. Here it is assumed that either single, type -1 failures or single, type +1 failures can occur but not both simultaneously. Since some of the matrices in this case become too large to conveniently present, the detail of the running example is not presented here.

The matrix $\underline{A}_{o}$ in Example 4.3.1 is extended to $\underline{A}_{o}^{a}$ as shown in Equation 4.7.2. By removing redundant rows, $\underline{A}_{o}^{a}$ simplifies to $\underline{A}_{o}^{a*}$ in

Equation 4.7.3. Using the computer program of Appendix A a set of extremal vectors for the solutions $\bar{\phi}$ to

$$(\underline{A}_o^{a*})^T \bar{\phi}^T = \bar{0}^T \ , \quad \bar{\phi} \neq \bar{0} \ , \quad \bar{\phi} \geq \bar{0} \ ,$$

turn out to be the vectors shown in the columns of $\underline{\Phi}_o^{a*}$ in Equation 4.7.4.

$$\underline{A}_o^a = \begin{bmatrix} \underline{A}_P \\ \hline \underline{A}_R^a \end{bmatrix} = \left[\begin{array}{ccc|l} 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 2 \\ 1 & -1 & -1 & 3 \\ 1 & -1 & -1 & 4 \\ \hline 1 & 1 & 1 & 5 \\ 1 & -1 & 1 & 6 \\ -1 & -1 & -1 & 7 \\ -1 & -1 & -1 & 8 \\ \hline 1 & 1 & 1 & 9 \\ -1 & 1 & 1 & 10 \\ 1 & -1 & -1 & 11 \\ 1 & -1 & -1 & 12 \\ \hline -1 & 1 & 1 & 13 \\ -1 & -1 & 1 & 14 \\ 1 & -1 & -1 & 15 \\ 1 & -1 & -1 & 16 \\ \hline 1 & -1 & 1 & 17 \\ -1 & -1 & 1 & 18 \\ 1 & 1 & -1 & 19 \\ 1 & 1 & -1 & 20 \end{array}\right] \quad \begin{array}{l} \\ \\ \text{no failures} \\ \\ \\ \\ +1 \text{ failure in } m_2 \\ \\ \\ \\ +1 \text{ failure in } m_2 \\ \\ \\ \\ -1 \text{ failure in } m_1 \\ \\ \\ \\ -1 \text{ failure in } m_2 \\ \\ \\ \end{array} \qquad (4.7.2)$$

$$\underline{A}_o^{a*} = \begin{bmatrix} A_P^* \\ \hline A_R^{a*} \end{bmatrix} = \left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 2 \\ 1 & -1 & -1 & 3 \\ \hline 1 & -1 & 1 & 6 \\ -1 & -1 & -1 & 7 \\ -1 & 1 & 1 & 10 \\ 1 & 1 & -1 & 19 \end{array}\right] \qquad (4.7.3)$$

$$\underline{\Phi}_o^{a*} \left[\begin{array}{ccc|c|c} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 3 \\ \hline 0 & 0 & 0 & 1 & 6 \\ 1 & 0 & 0 & 1 & 7 \\ 0 & 0 & 1 & 1 & 10 \\ 0 & 1 & 0 & 1 & 19 \end{array}\right] \qquad (4.7.4)$$

Although it is too large to present here the matrix $\underline{\Phi}_o^a$ can be written from $\underline{\Phi}_o^{a*}$ using a transition chart and the extremal set extension technique of Appendix B. Then from $\underline{\Phi}_o^a$ the matrix $\underline{D}_{11}^{a'}$ in Equation 4.7.5 can be written using the transformation matrix $\underline{T}_{11}^a$. For values of $\gamma > 1$, $\underline{D}_{\gamma i}^{a'}$ can be constructed from $\underline{D}_{11}^{a'}$ by simply adding $2(\gamma-1)$ partitions whose columns are shown in $\underline{R}_1^a$ given by Equation 4.7.6 and by shifting the partition $\underline{R}_\mu$ to the proper location.

From this point on, the problem is continued by applying the algorithm of Figure 4.6.1. After successively performing the steps for

$\gamma = 1, 2,$ and 3 the procedure terminates on $\gamma = 4$. This iteration begins by constructing

$$\underline{D}_{41}^{a'} = [\underline{R}_P^a \mid \underline{R}_\mu^a \mid \underline{R}_1^a \mid \underline{R}_1^a \mid \underline{R}_1^a \mid \underline{R}_1^a \mid \underline{R}_1^a \mid \underline{R}_1^a]$$

and

$$\underline{D}_{42}^{a'} = [\underline{R}_P^a \mid \underline{R}_1^a \mid \underline{R}_1^a \mid \underline{R}_\mu^a \mid \underline{R}_1^a \mid \underline{R}_1^a \mid \underline{R}_1^a \mid \underline{R}_1^a] .$$

This data along with the pattern vectors $\bar{y}_1'$, $\bar{y}_2'$, $\bar{y}_3'$, $\bar{y}_4'$ is used as the input to the program in Section D.2 to compute

$$\bar{c}_1 = (1, -1, -1, -1) .$$

$$\underline{D}_{11}^{a'} = [\underline{R}_P^a \mid \underline{R}_\mu^a]$$

$$\underline{D}_{11}^{a'} = \left[\begin{array}{cccccccccccccc|cc}
1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 2 & 2 & 2 & & \\
0 & 0 & 1 & 1 & 2 & 2 & 2 & 1 & 2 & 1 & 1 & 0 & 0 & 0 & & \\
1 & 0 & 1 & 0 & 2 & 1 & 1 & 1 & 0 & 0 & 2 & 2 & 1 & 0 & & \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 2 & 2 & 0 & 0 & 1 & 2 & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1
\end{array}\right] \qquad (4.7.5)$$

$$\underline{R}_1^a = \left[\begin{array}{cccccccc}
1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right] \qquad (4.7.6)$$

Using this value of $\bar{c}_1$ the matrix $\underline{\Lambda}_{21}$ is constructed from the set of extremal vectors for the solutions $\bar{\lambda}$ to

$$[\bar{c}_1 \mid 1]\underline{D}_{41}^{a'}\,\bar{\lambda}^T = 0 \ , \quad \bar{\lambda} \neq \bar{0} \ , \quad \bar{\lambda} \geq \bar{0} \ .$$

Since the matrix $[\bar{c}_1 \mid 1]\underline{D}_{41}^{a'}$ has only one row, $\underline{\Lambda}_{41}$ can be found by inspection using the theory of Appendix B rather than using the program in Appendix A.

In order to compute $\bar{c}_2$ the matrices

$$\underline{D}_{42}^{a'}\,\underline{\Lambda}_{41}^{T}$$

and

$$\underline{D}_{43}^{a'}\,\underline{\Lambda}_{41}^{T}$$

are constructed and entered as data in the program of Section D.2. The result is

$$\bar{c}_2 = (1, -1, 1, 1) \ .$$

According to the algorithm the next step is to construct $\underline{\Lambda}_{42}$ from the extremal vectors for the solutions $\bar{\lambda}$ to

$$\left[\begin{array}{c|c} [\bar{c}_1 \mid 1]\underline{D}_{41}^{a'} \\ \hline [\bar{c}_2 \mid 1]\underline{D}_{42}^{a'} \end{array}\right] \bar{\lambda}^T = \bar{0} \ , \quad \bar{\lambda} \neq \bar{0} \ , \quad \bar{\lambda} \geq \bar{0} \ .$$

These extremal vectors are found by removing all redundant columns from

$$\left[\begin{array}{c|c} [\bar{c}_1 \mid 1]\underline{D}_{41}^{a'} \\ \hline [\bar{c}_2 \mid 1]\underline{D}_{42}^{a'} \end{array}\right] ,$$

finding the extremal vectors for the simplified system with the program of Appendix A, and generating the rows of $\underline{\Lambda}_{42}$ by the extremal set extension technique of Appendix B.

The vector

$$\bar{c}_3 = (1, -1, -1, -1)$$

is found with the program of Section D.2 from $\underline{D}_{43}^{a'}$ and $\underline{D}_{44}^{a'}$. The output of this computer run includes the set of constraints that $\bar{c}_4$ must satisfy. The resulting system of inequalities is

$$[\bar{c}_4 \mid 1] \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \gtrless \bar{0}^T$$

and is obviously satisfied by several vectors. If the system of inequalities is more complex than in this example the program of Section D.3 could be used to find

$$\bar{c}_4 = (-1, 1, -1, -1) \quad .$$

In summary, both type +1 and type -1 failures can be corrected in the problem presented in Example 4.3.1 by adding four redundant TLU's with output vectors

$$\bar{c}_1' = (1, -1, 1, 1)$$
$$\bar{c}_2' = (1, -1, -1, -1)$$
$$\bar{c}_3' = (1, -1, 1, 1)$$
$$\bar{c}_4' = (-1, 1, 1, 1) \quad .$$

In this example the correction of single type -1 or type +1 failures requires the addition of four redundant TLU's while the nonredundant network shown in Figure 4.3.1 contains only two first layer TLU's. The following example illustrates that the amount of redundancy required in Example 4.7.1 is not always necessary. It shows that a certain amount of overdesign in the nonredundant TLU network can result in a considerable savings in the amount of redundancy required to achieve single error correction of both type +1 and -1 failures.

Example 4.7.2. Consider the two-layer TLU network shown in Figure 4.7.1 and the corresponding pattern-to-image-space transformation shown in Figures 4.7.2 and 4.7.3.

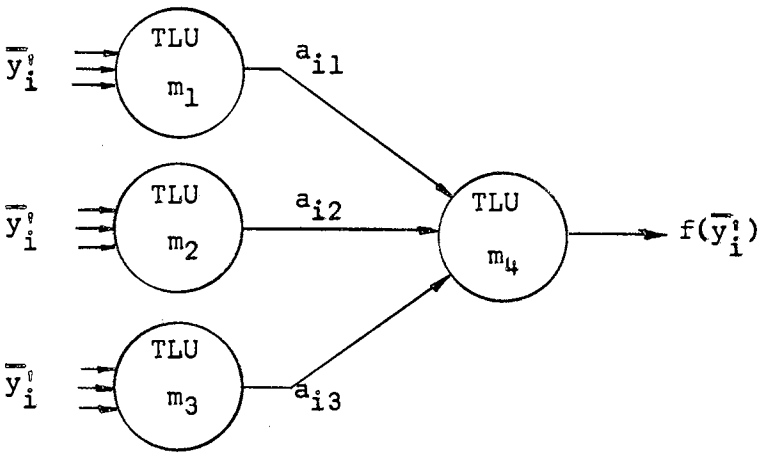


Figure 4.7.1. Nonredundant TLU Network for Example 4.7.2

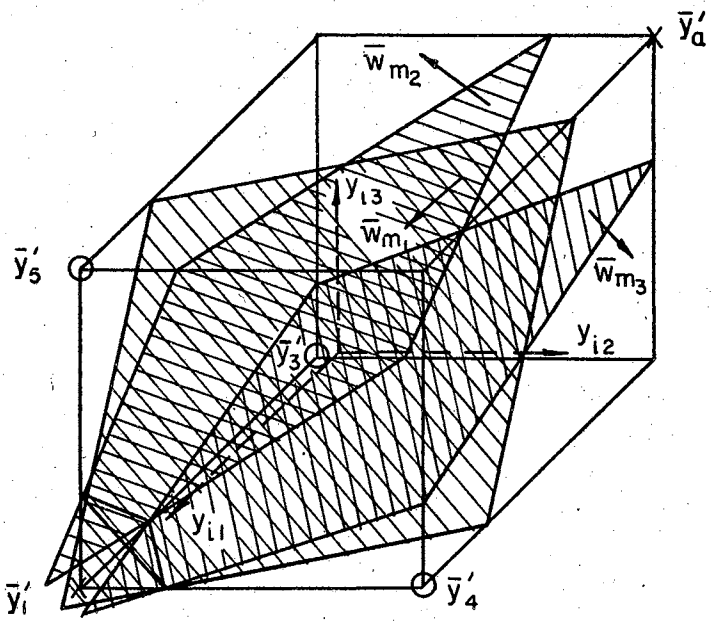


Figure 4.7.2. Pattern Space for Example 4.7.2

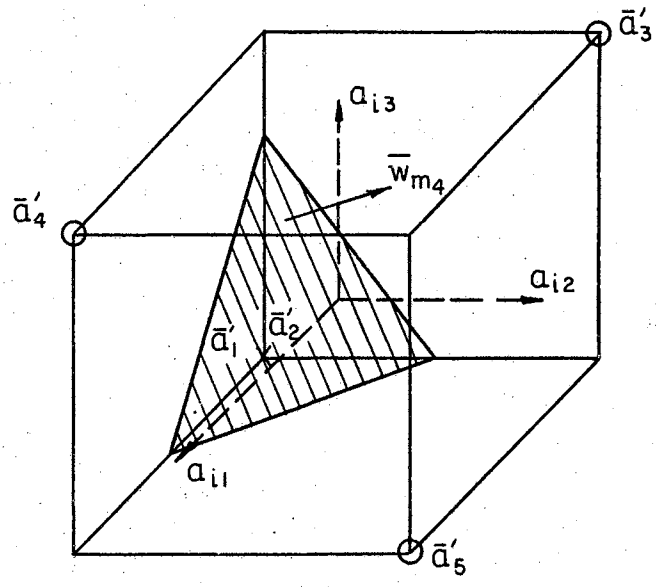


Figure 4.7.3. Image Space for Example 4.7.2

From the image space patterns the matrix $\underline{A}_P$ can be written as

$$\underline{A}_P = \begin{bmatrix} -1 & -1 & -1 & 1 \\ -1 & -1 & -1 & 1 \\ 1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 \\ -1 & -1 & 1 & -1 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix}$$

The matrix $\underline{A}_O^{a^*}$ as obtained from $\underline{A}_O^{a}$ is shown in Equation 4.7.7. The numbering of the rows of $\underline{A}_O^{a^*}$ corresponds to their location in $\underline{A}_O^{a}$ which is not shown because of its size.

$$\underline{A}_O^{a^*} = \begin{bmatrix} -1 & -1 & -1 & 1 \\ 1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 \\ -1 & -1 & 1 & -1 \\ \hline 1 & -1 & -1 & 1 \\ -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 \end{bmatrix} \begin{matrix} 1 \\ 3 \\ 4 \\ 5 \\ \hline 6 \\ 8 \\ 11 \\ 16 \\ 24 \\ 25 \\ 30 \end{matrix} \begin{matrix} \\ \\ \text{from } \underline{A}_P \\ \\ \hline \\ \\ \\ \text{from } \underline{A}_R^a \\ \\ \\ \\ \end{matrix} \qquad (4.7.7)$$

Using the program of Appendix A, the resulting matrix $\underline{\Phi}_O^{a^*}$ is as shown in Equation 4.7.8. It is significant that $\underline{\Phi}_O^{a^*}$ indicates that there are no PLD's involving any of the rows 1, 3, 4, and 5 from $\underline{A}_P$. This is a

$$\underline{\Phi}_0^{a^*} = \left[\begin{array}{ccc|c} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 5 \\ \hline 0 & 0 & 1 & 6 \\ 0 & 0 & 0 & 8 \\ 0 & 1 & 0 & 11 \\ 1 & 0 & 0 & 16 \\ 1 & 0 & 0 & 24 \\ 0 & 1 & 0 & 25 \\ 0 & 0 & 1 & 30 \end{array}\right] \qquad (4.7.8)$$

direct result of the fact that the image space patterns $\bar{a}_1'$ and $\bar{a}_2'$ are a Hamming distance of two from the patterns $\bar{a}_3'$, $\bar{a}_4'$, and $\bar{a}_5'$ in Figure 4.7.3. The importance of this is that the submatrices $\underline{\Phi}_P^a$ in $\underline{\Phi}_\gamma^a$ are identically zero. As a result, the matrices $\underline{D}_{\gamma i}^{a'}$, $i = 1, 2, \ldots, \gamma$, do not change with either γ or i. The matrices $\underline{D}_{\gamma i}^a$, $i = 1, 2, \ldots, \gamma$, are given by

$$\underline{D}_{\gamma i}^a = [\underline{R}_1^a \mid \underline{R}_2^a \mid \underline{R}_1^a \mid \cdots \mid \underline{R}_\mu^a \mid \cdots \mid \underline{R}_1^a]$$

$$\underline{D}_{\gamma i}^a = \left[\begin{array}{c|cccccccccccc|c} & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & \\ & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & \\ & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & \\ \underline{0} & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \underline{0} \\ & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \end{array}\right]$$

$\underline{R}_1^a$ $\qquad$ $\underline{R}_2^a$ $\qquad$ all other partitions

for all γ and i. This matrix reduces to

$$\underline{D}_{\gamma i}^{a'} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} .$$

For this example $\gamma = 2$ and the vector $\bar{c}_1$ is computed by using the pattern space vectors and the matrices $\underline{D}_{21}^{a'}$ and $\underline{D}_{22}^{a'}$ as input to the program in Section D.2. The result is

$$\bar{c}_1 = (-1, -1, 1, -1, -1)$$

or

$$\bar{c}_1' = (-1, -1, -1, 1, 1)$$

and the unique constraints on $\bar{c}_2$ result in the inequalities

$$[\bar{c}_2 \mid 1] \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \gtrless \bar{0}$$

which must be satisfied by $\bar{c}_2$. Obviously there are several values of $[\bar{c}_2 \mid 1]$ which satisfy this system of inequalities. The computer program of Section D.3 could be used to find $\bar{c}_2$, but it can be found by inspection in this simple case. One realizable choice is

$$\bar{c}_2 = (-1, -1, -1, -1, 1)$$

for which

$$\bar{c}_2' = (-1, -1, 1, 1, -1)$$

means that TLU m_3 is repeated. $\bar{c}_1'$ means that TLU m_1 is repeated.

The result of this example is that, for three first-layer TLU's in the nonredundant network, two redundant TLU's are sufficient to provide single error correction of both type +1 and type -1 failures. It is significant that the pattern space is incompletely specified; that is, of the $2^3 = 8$ possible patterns only five are considered. This is, in a sense, redundancy.

4.8 A Geometrical Description of the Synthesis Algorithm. The redundancy synthesis algorithm presented here has not been shown to converge by a formal proof. However, examples have shown that it does converge for the specific problems tried. In order to give the reader a better understanding of the mechanics of the algorithm, a geometrical discussion is given here.

Assume that for some γ the algorithm is at the point of selecting the best $\bar{c}_i$. The previous vectors $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_{i-1}$ result in solutions $\bar{\lambda}$ to the system

$$\begin{bmatrix} [\bar{c}_1 \mid \mu]\underline{D}'_{\gamma 1} \\ \hline [\bar{c}_2 \mid \mu]\underline{D}'_{\gamma 2} \\ \hline \vdots \\ \hline [\bar{c}_{i-1} \mid \mu]\underline{D}'_{\gamma(i-1)} \end{bmatrix} \bar{\lambda}^T = \bar{0}^T , \quad \bar{\lambda} \neq \bar{0} , \quad \bar{\lambda} \geq \bar{0} , \qquad (4.8.1)$$

which means that the desired error correction has not been achieved.

The solutions $\bar{\lambda}$ to Equation 4.8.1 can be visualized for the case of $\gamma = 3$ in the space shown in Figure 4.8.1. The two hyperplanes shown are generated by the solutions $\bar{\lambda}$ to

$$[\bar{c}_1 \mid \mu]\underline{D}'_{31}\,\bar{\lambda}^T = 0 \qquad (4.8.2)$$

and

$$[\bar{c}_2 \mid \mu]\underline{D}'_{32}\,\bar{\lambda}^T = 0 \quad . \qquad (4.8.3)$$

Notice that the restriction of $\bar{\lambda} \geq 0$ and $\bar{\lambda} \neq \bar{0}$ restrict the solutions to the positive orthant and away from the origin. The solutions to Equations 4.8.2 and 4.8.3 are contained in the intersection of the two hyperplanes and in the portion of the space bounded by $\bar{\lambda} \geq 0$ and $\bar{\lambda} \neq \bar{0}$. The resulting convex polyhedral cone (a half-line) is spanned by the single extremal vector shown in the figure.

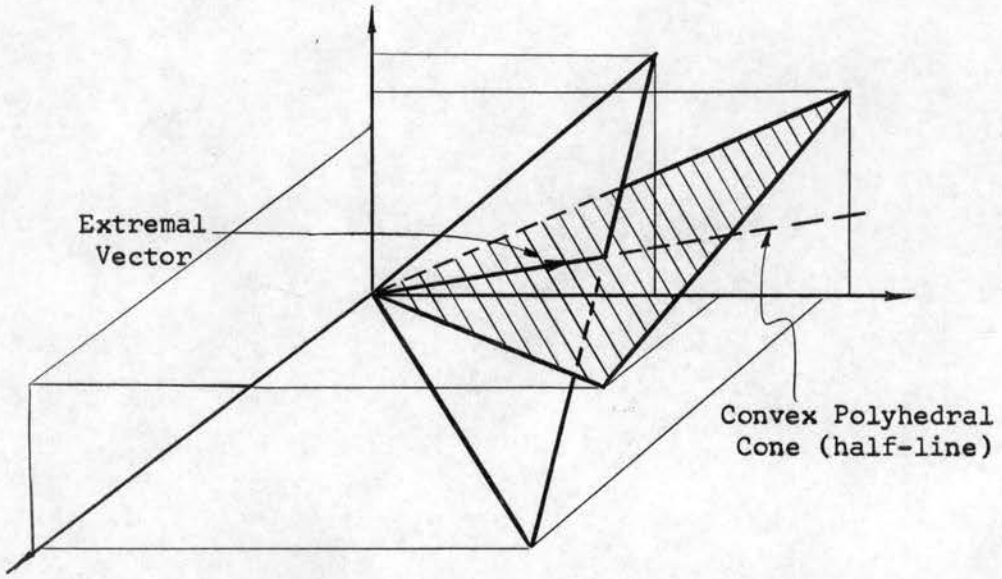


Figure 4.8.1. Intersection of Hyperplanes in E_3

In general the extremal vectors for the cone thus formed are rows in the matrix

$$\underline{\Lambda}_{\gamma(i-1)} = \begin{bmatrix} \\ \overline{\lambda}_1 \\ \overline{\lambda}_2 \\ \vdots \\ \overline{\lambda}_{s_{\gamma(i-1)}} \end{bmatrix} .$$

The desired result of the algorithm is to find a $\overline{c}_\gamma$ such that the hyperplane defined by the vector

$$[\overline{c}_\gamma \mid \mu]\underline{D}'_{\gamma\gamma}$$

does not intersect the convex polyhedral cone described by the extremal vectors in $\underline{\Lambda}_{\gamma(\gamma-1)}$. However, for $\overline{c}_i$, $i < \gamma$, such a hyperplane does not exist. Therefore, how should $\overline{c}_i$ be selected?

The criterion proposed here for selecting $\overline{c}_i$, $i < \gamma$, is based on Definition 4.6.1. This is an intuitive approach and its optimality remains to be shown. In order to understand this criterion, consider the hyperplane in $\overline{\lambda}$-space defined by the vector

$$[\overline{c}_i \mid \mu]\underline{D}'_{\gamma i} .$$

Since it is assumed that $i < \gamma$, this hyperplane intersects the cone described by Equation 4.8.1. As a result, certain positive linear combinations of the extremal vectors in $\underline{\Lambda}_{\gamma(i-1)}$ result in

$$[\overline{c}_i \mid \mu]\underline{D}'_{\gamma i}\,\overline{\lambda}^T = 0 \qquad (4.8.4)$$

where

$$\bar{\lambda}^T = \underline{\Lambda}^T_{\gamma(i-1)} \bar{\omega}^T \quad , \quad \bar{\omega} \neq \bar{0} \quad , \quad \bar{\omega} \geq \bar{0} \quad .$$

Obviously the set of $\bar{\omega}$'s satisfying

$$[\bar{c}_i \mid \mu] \underline{D}'_{\gamma i} \underline{\Lambda}^T_{\gamma(i-1)} \bar{\omega}^T = 0 \quad , \quad \bar{\omega} \neq \bar{0} \quad , \quad \bar{\omega} \geq \bar{0} \quad , \qquad (4.8.5)$$

can be spanned by a finite set of extremal vectors $\bar{\omega}_k$. Corresponding to each such extremal vector a half-space (not necessarily unique) in the space of $[\bar{c}_{i+1} \mid \mu]$ vectors is defined by the inequality

$$[\bar{c}_{i+1} \mid \mu] \underline{D}'_{\gamma(i+1)} \underline{\Lambda}^T_{\gamma(i-1)} \bar{\omega}^T_k \gtrless 0 \quad . \qquad (4.8.6)$$

Actually two half-spaces are defined by this inequality due to the symbol $\gtrless$. The important point here is that, if the positive linear combinations $\underline{\Lambda}^T_{\gamma(i-1)} \bar{\omega}^T$ of the vectors in $\underline{\Lambda}^T_{\gamma(i-1)}$ which are solutions to Equation 4.8.4 are not to be solutions $\bar{\lambda}$ to

$$[\bar{c}_{i+1} \mid \mu] \underline{D}'_{\gamma(i+1)} \bar{\lambda}^T = [\bar{c}_{i+1} \mid \mu] \underline{D}'_{\gamma(i+1)} \underline{\Lambda}^T_{\gamma(i-1)} \bar{\omega}^T = 0 \quad ,$$

then the vector $[\bar{c}_{i+1} \mid \mu]$ must be in the intersection of all the half-spaces defined by the vectors

$$\underline{D}'_{\gamma(i+1)} \underline{\Lambda}^T_{\gamma(i-1)} \bar{\omega}^T_k$$

as in Inequality 4.8.6. If $i + 1 < \gamma$, then no such $\bar{c}_{i+1}$ exists; however, the concept of the intersection of these half-spaces forms the basis for Definition 4.6.1.

The matrix $\underline{\Omega}_{\gamma i}$ is the matrix whose rows are the extremal vectors $\bar{\omega}_k$ for the set of solutions $\bar{\omega}$ to Equation 4.8.5. The system of inequalities

$$[\bar{c}_{i+1} \mid \mu] \underline{D}'_{\gamma(i+1)} \underline{\Lambda}^T_{\gamma(i-1)} \underline{\Omega}^T_{\gamma i} \gtrless \bar{0}$$

defines the empty or nonempty intersection of half-spaces which restricts the vectors $[\overline{c}_{i+1} \mid \mu]$. The fundamental idea behind Definition 4.6.1 is that this intersection should be of as few as possible unique half-spaces. Thus, the best solution region is bounded by as few hyperplanes as possible and the choice of $\overline{c}_i$ can effect this number. The result is that $\overline{c}_i$ is chosen such that the matrix

$$\overline{D}'_{\gamma(i+1)}\overline{\Lambda}^{T}_{\gamma(i-1)}\Omega^{T}_{\gamma i}$$

has as few as possible unique columns and such that $\overline{c}'_i$ is realizable. This definition of the best $\overline{c}_i$ is best in the sense that the number of unique inequality constraints on $\overline{c}_{i+1}$ is minimized.

It should be pointed out that some other set of constraints and the resulting intersection of half-spaces might be better when the realizability of $\overline{c}'_{i+1}$ is considered. The geometrical constraints imposed by realizability is still an open question. It appears that a better criterion could be found by considering the location of the intersection of the half-spaces relative to the location of the $\overline{c}_i$'s corresponding to realizable $(\overline{c}'_i)$'s.

4.9 Conclusion. This chapter consists of a step-by-step development of an algorithm for selecting the output vectors of redundant TLU's in the first layer of a two-layer network. In this algorithm an attempt has been made to utilize the fundamental mathematical ideas presented in Appendices A and B. As a result of this use of mathematics the criterion for the selection of TLU's is based on an intuitive, geometrical view of the problem rather than a blind application of redundancy. The question of whether this approach actually pays off in terms of less

redundancy than for approaches such as simple triplication is approached by examples and is by no means rigorous. It is felt that, regardless of the practical value of this approach, the presentation of the task to be performed in introducing redundancy by the removal of PLD's is of value. It can be seen that the task is not as simple as in the problem of network synthesis by removal of PLD's as presented by Hopcroft and Mattson (13). The fact that the redundant TLU's are allowed to fail is the source of the difficulty.

Contained in this chapter is the development of the constraints on a TLU output vector $\bar{c}_i$ given $\bar{c}_{i-1}, \bar{c}_{i-2}, \ldots, \bar{c}_1$. These constraints are the starting place for any iterative technique for selection of the $\bar{c}_i$'s. Obviously, one approach is to try all combinations of a set of vectors $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_\gamma$, but the limitations here are obvious. Therefore, an iterative technique is needed, which implies that a criterion is needed for selection of $\bar{c}_i$ given that it cannot satisfy the constraints generated by $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_{i-1}$. The resulting criterion which is proposed is presented in Sections 4.5 and 4.6 and discussed in Section 4.8.

The entire development of the algorithm is centered around correcting only one type of single, first-layer error. The reason for this is to allow a simple presentation while leaving open the possibility of considering failures for which a failed TLU has its output held at any value μ. Obviously, in order to compare the redundancy, thus induced, to conventional redundancy, the occurrence of both +1 and -1 failures must be considered. The theory is flexible to the extent that this extension is permitted without excessive complication. This extension of the theory is the topic for Section 4.7.

After synthesizing some redundant networks, it appears that there

is a relationship between the minimality of the nonredundant network realization and the number of redundant TLU's. In other words, it appears that a minimal, nonredundant realization requires more redundant TLU's per TLU in the nonredundant network than for a subminimal realization. The lack of theory concerning minimal realizations prevents a more precise statement of the results in this situation. The examples shown here indicate that this approach is little or no better than triplication for minimal realizations as in Example 4.7.1. For subminimal realizations the situation is different.

Example 4.7.2, for which the nonredundant realization is subminimal, illustrates a different situation. In this example three significant factors permit the use of less redundancy per nonredundant network TLU than in Example 4.7.1. First, the pattern space is incompletely specified; second, three first layer TLU's were used rather than the minimum of two; and third, the pattern space hyperplanes are located so that the image space patterns in opposite classifications are a Hamming distance of two away from each other.

Regardless of the minimality of a nonredundant realization, the theory presented here should help to determine the feasibility of introducing redundancy in TLU networks from a purely mathematical point of view. It is quite possible that some major simplifications can be made in the algorithm presented here. The algorithm, as it is, is quite cumbersome and only small examples can be considered. Its value may fall into one of two catagories. It may be the starting point for a more efficient algorithm or it may indicate that a completely different approach is more feasible.

CHAPTER V

SUMMARY AND CONCLUSIONS

5.1 Summary. The problem considered here can be broken into two general areas. The first area involves a comparison of two types of decision-makers and the second area involves the development of an algorithm to make reliability improvements in a particular implementation of one of the decision-maker types.

The investigation of the first problem area is based on two binary, decision-making system models which possess the desired characteristics of each decison-maker type. One model represents a decision-maker in which unreliable, binary information is fed into a set of n inputs, each of which leads directly to at least one switching element. These switching elements are modeled with binary channels. The interconnection of these input channels is then represented by another binary channel. The other decision-maker model is essentially the same except that the inputs do not lead to individual switching elements and the interconnection channel is assumed to be more complicated.

The two decision-maker types are compared on the basis of risk functions. These risk functions are evaluated as functions of the input switching element channel parameters, the number of input channels, the interconnection channel parameters, and the type of decision problem.

This comparison indicates the validity of the problem of redundancy in threshold logic decision-makers. In particular, if threshold logic

is used to implement one of the two decision-maker types, then the comparison indicates that networks of threshold logic units (TLU's) must be used rather than single TLU's.

Beginning in Chapter III a development is presented for introducing redundancy in two-layer networks of TLU's. The basis of this development is a mathematical model of the network. This model is a system of linear inequalities and takes into consideration the possibility of failures of TLU's in the first layer of the network. The development proceedes at first on the assumption that a failure consists of a TLU output being hung at one particular value μ. Thus the redundancy must correct for single, "type μ" errors. In Section 4.7 the theory is extended to consider both type μ and $-\mu$ failures.

The manner of selecting the redundant TLU's consists of finding a vector $\overline{c}_i$ which satsifies a set of linear inequality constraints and which determines the design of the TLU. The constraints on this vector are found from an application of the theory of linear inequalities. It is necessary in most situations to add more than one redundant TLU; therefore, an iterative technique is needed for computing a set of γ vectors $\overline{c}_1, \overline{c}_2, \ldots, \overline{c}_\gamma$ using the constraints mentioned above.

Chapter IV contains the development of the iterative technique and illustrates, by examples, the resulting redundancy synthesis algorithm. These examples also illustrate to a limited degree the feasibility and utility of the algorithm. Due to the time required in applying the algorithm, three computer programs were written to aid in the computations. These programs are presented in Appendices A and D.

5.2 Conclusions. The most important conclusion which can be drawn

from the comparison appearing in Chapter II involves the particular task to which decision-makers with distributed redundancy are applicable. If the interconnection channel of such a decision-maker can be made reasonably reliable, then it is better (in terms of risk for decision problems like the full pattern set case) than a decision-maker with switching elements on every input. This implies that any decision-maker implementation, which possesses the characteristics attributed to distributed redundancy, is most applicable to decision problems in which all combinations of input patterns are reasonably likely to occur. This condition becomes stronger as the number of input channels increases.

On the other hand, the comparison shows that, if only a few of the possible input patterns are likely to occur, then conventional networks of switching elements are better in terms of risk. For this particular problem type it is possible to use a single TLU (which possesses certain distributed redundancy characteristics) to perform the decision making; however, its interconnection of inputs must have unreasonably high reliability in order to justify its use.

The type of decision problem, for which distributed redundancy is applicable, is such that a single TLU cannot perform all of the decision tasks which can arise. Therefore, networks of TLU's must be considered and ways found for improving their reliability.

The technique suggested here for improving reliability in two-layer TLU networks is centered on correction of errors occurrring in the first layer. There is no need to be concerned with compensating for second layer failures because the resulting reliability can be no greater than that of the device which compensates for second layer failures.

Perhaps the most fundamental trait of the error-correction scheme

proposed here is that it adds redundancy in such a way that it insures the existence of an error-correcting second layer. Other approaches (such as Coates and Bargainer (5)) correct for errors through extending an existing second layer design by repeating some of the existing non-redundant, first layer TLU's.

The development of the synthesis algorithm demonstrates the possibility of iteratively selecting redundant TLU's. The vector $\overline{c}_i$, characterizing the ith redundant TLU, can be computed from a knowledge of the previous vectors $\overline{c}_1, \overline{c}_2, \ldots, \overline{c}_{i-1}$ and the inequality constraints which apply to $\overline{c}_1$. Thus, the inequality constraints on $\overline{c}_i$ do not have to be computed as if it were the first redundant TLU added. As a result of this iterative approach, a more direct tie between the $\overline{c}_i$'s exists and it is easier to visualize the effects of one on the other.

The redundancy synthesis algorithm has some rather severe limitations with the most significant being the size of the matrices of constraint vectors used to evaluate the $\overline{c}_i$'s. The size of these matrices prevent consideration of examples very much larger than those shown in Chapter IV.

The examples in Section 4.7 point out some interesting characteristics although no definite conclusions can be made from them. In Example 4.7.1 the nonredundant realization has a minimum number of first layer TLU's. The result is that correction of single, type +1 or -1 errors can be corrected only by adding four redundant TLU's to the two nonredundant first layer TLU's. In Example 4.7.2 not all of the pattern vectors are allowed to occur and the nonredundant realization is subminimal. The desired error-correction is achieved by adding only two redundant TLU's to the three nonredundant, first layer TLU's. These results

indicate that a certain amount of over design in Example 4.7.2 has permitted the use of less redundancy than is required in the minimal, nonredundant realization of Example 4.7.1.

In summary, the area of applicability for distributed redundancy is with decision problems for which it is likely that any set of the possible binary inputs to the decision-maker can occur. For decision problems that cause it to be unlikely that the decision-maker will receive any patterns except a very small portion of those possible, conventional networks of switching elements are best. If it is feasible to use TLU network decision-makers, it appears that a strictly mathematical redundancy synthesis approach is most likely to pay off if a certain amount of over design is used in the nonredundant realization. If the redundancy synthesis algorithm developed here is used, the decision-maker must be small in terms of the number of patterns into the first layer and the number of first layer TLU's.

5.3 Recommendations for Further Study. The following is a list of suggestions of topics for further study.

1. A better model is needed for the interconnection of decision-maker inputs. For example, the parameters of the model should be functions of the inputs to the decision-maker.

2. It is very likely that a considerable amount of simplification can be done on the synthesis algorithm developed here. The matrices $\underline{D}'_{\gamma i}$ contain many redundant columns in most examples and all of these columns may not be needed.

3. Referring to Section 4.8, a need exists for finding a criterion for finding $\bar{c}_i$'s such that consideration is given to the relationship

between the realizable vectors $\bar{c}_i$ and the intersection of half-spaces which bound the solution regions for the $\bar{c}_i$'s.

4. The effect of variations in TLU parameters could be considered in the selection criterion for the $\bar{c}_i$'s.

5. A study is needed to determine the feasibility (in terms of increasing reliability) of multiple error correction for TLU networks in particular.

6. The synthesis of a redundant TLU network could be approached from strictly a Hamming distance point of view.

7. The synthesis of a redundant TLU network could be considered in the synthesis of the nonredundant network instead of "adding on" redundant TLU's.

BIBLIOGRAPHY

1. Brain, A. E. "The Simulation of Neural Elements by Electrical Networks Based on Multi-Aperature Magnetic Cores." Proceedings of the IRE. Vol. 49. (1961) 49-52.

2. Browne, E. T. Introduction to the Theory of Determinants and Matrices. Chapel Hill: The University of North Carolina Press, 1958.

3. Chow, C. K. "Boolean Functions Realizable With Single Threshold Devices." Proceedings of the IRE. (Correspondence). Vol. 49. (1961) 370-371.

4. Coates, C. L. and P. M. Lewis. "Linearly Separable Switching Functions." J. Franklin Inst. Vol. 272. (1961) 366-410.

5. Coates, C. L. and J. D. Bargainer. "Error Correcting Threshold Gate Networks." Proceedings of the Southwestern IEEE Conference. Dallas, Texas, 1966.

6. Dertouzos, M. L. "An Approach to Single-Threshold-Element Synthesis." IEEE Transactions on Electronic Computers. Vol. EC-13. (1964) 519-528.

7. Gableman, I. J. "A Note on the Realization of Boolean Functions Using a Single Threshold Element." Proceedings of the IRE. (Correspondence). Vol. 50. (1962) 225-226.

8. Goldman, A. J. and A. W. Tucker. "Polyhedral Convex Cones." Annals of Mathematics Study No. 38. Eds. H. W. Kuhn and A. W. Tucker. Princeton: Princeton University Press, 1956.

9. Good, R. A. "Systems of Linear Relations." Soc. Ind. Appl. Math. Review. Vol. 1. (1959) 1-31.

10. Gorden, P. "Uber die Auflosungen Linearer Gleichungen mit Reelen Coefficienten." Mathematische Annalen 6. (1873) 23-28.

11. Ho, Y. C. and R. L. Kashyap. "An Algorithm for Linear Inequalities and Its Applications." IEEE Transactions on Electronic Computers. Vol. EC-14. (1965) 683-688.

12. Highleyman, W. H. "A Note on Linear Separation." IRE Transactions on Electronic Computers. (Correspondence). Vol. EC-10. (1961) 777-778.

13. Hopcroft, J. E. and R. L. Mattson. "Synthesis of Minimal Threshold Logic Networks." IEEE Transactions on Electronic Computers. Vol. EC-14. (1965) 552-560.

14. Jenson, P. A. "The Reliability of Redundant Multiple-Line Networks." IEEE Transactions on Reliability. Vol. R-13. (1964) 23-33.

15. Kaszerman, P. "A Geometric Test-Synthesis Procedure for a Threshold Device." Information and Control. Vol. 6. (1963) 381-398.

16. Knox-Seith, J. K. "Improving the Reliability of Digital Systems by Redundancy and Restoring Organs." TR NO 4816-2. Stanford, California: Stanford Electronics Laboratory, Stanford University, 1964.

17. Nilsson, N. J. Learning Machines. New York: McGraw-Hill, 1965.

18. Papoulis, A. Probability, Random Variables, and Stochastic Processes. New York: McGraw-Hill, 1965.

19. Paull, M. C. and E. J. McCluskey, Jr. Proceedings of the IRE. (Correspondence). Vol. 48. (1960) 1335-1337.

20. Pierce, W. H. "Improving the Reliability of Digital Systems by Redundancy and Adaption." TR NO 1552-3. Stanford, California: Stanford Electronics Laboratory, Stanford University, 1961.

21. Pierce, W. H. "Adaptive Decision Elements to Improve the Reliability of Redundant Systems." IRE Intern. Conv. Rec., Vol. 10, Pt. 4. (1962) 124-131.

22. Pierce, W. H. "Interwoven Redundant Logic." J. Franklin Inst. Vol. 277. (1964) 55-85.

23. Stokes, R. W. "A Geometric Theory of the Solution to Linear Inequalities." Trans. Amer. Math. Soc. Vol. 33. (1931) 782-805.

24. Weintraub, S. Tables of the Cumulative Binomial Probability Distribution for Small Values of p. New York: Free Press of Glencoe, 1963.

25. Wilcox, R. H. and W. C. Mann, Eds. Redundancy Techniques for Computing Systems. Washington, D. C.: Spartan, 1962.

APPENDIX A

DEVELOPMENT OF THE APPLIED THEORY OF LINEAR INEQUALITIES

A.1 Introduction. The intent here is to develop a technique for computing a set S_ϕ of vectors which can be used to form a system of linear inequality constraints that a vector $\bar{\theta}$ must satisfy in order for the system

$$[\underline{A} \mid \bar{\theta}^T]\bar{w}_1^T > \bar{0}^T \qquad (A.1.1)$$

to have a solution $\bar{w}_1$ when the system

$$\underline{A}\bar{w}_o^T > \bar{0}^T \qquad (A.1.2)$$

does not necessarily have a solution. The matrix $\underline{A}$ consists of an m x n array of real numbers where m > n. The resulting technique can also be used to test for the existence of a solution $\bar{w}$ to a system of the form

$$\underline{A}\bar{w}^T > \bar{0}^T \quad . \qquad (A.1.3)$$

A.2 Gordon's Theorem. The basis for the synthesis algorithm in Chapter IV and the technique developed here is a theorem due to Gordon (10) in 1873. This theorem is discussed and proved by Good (9).

Theorem A.2.1. There exists a solution $\bar{w}$ to

$$\underline{A}\bar{w}^T > \bar{0}^T \qquad (A.2.1)$$

if and only if there exists no solution $\overline{\phi}$ to

$$A^T \overline{\phi}^T = \overline{0}^T \quad , \quad \overline{\phi} \neq \overline{0} \quad , \quad \overline{\phi} \geq \overline{0} \quad . \qquad \text{(A.2.2)}$$

The interpretation of this theorem is that there can be no positive linear dependencies (PLD's) among the rows of $\underline{A}$. The "removal" of these dependencies is the desired result of the technique presented here. Hopcroft and Mattson (13) have used the same technique in the synthesis of TLU networks.

In order to remove the PLD's a column $\overline{\theta}^T$ is added to $\underline{A}$ such that, for every $\overline{\phi}$ satisfying Equation A.2.2, $\overline{\theta}$ satisfies

$$\overline{\theta}\ \overline{\phi}^T \gtrless \overline{0}^T \quad . \qquad \text{(A.2.3)}$$

The symbol $\gtrless$ means that either > or < holds but not both. It is possible that there exists no $\overline{\theta}$ such that Inequality A.2.3 is satisfied. If this situation occurs, more than one column can be added to $\underline{A}$; however, the selection of the columns is complicated. The redundancy synthesis algorithm of Chapter IV considers the case while the development here is restricted to the derivation of constraints on a single additional column.

A significant point in the removal of PLD's is that the set

$$S_A = \{\overline{\phi} \mid \underline{A}^T \overline{\phi}^T = \overline{0} \ , \ \overline{\phi} \neq \overline{0} \ , \ \overline{\phi} \geq \overline{0}\}$$

forms a convex polyhedral cone with the vector $\overline{0}$ removed. According to a theorem by Minkowski presented on page 30 of Goldman and Tucker (8) every member of a convex polyhedral cone can be expressed as a nonnegative, linear combination of extremal vectors for the cone. Minkowski's theorem is stated formally in Section A.3.

Definition A.2.1. An extremal vector of a convex polyhedral cone is a vector $\bar{\phi}_k$ of the cone which cannot be written as a positive, linear combination of two other vectors in the cone unless the two vectors are constant multiples of $\bar{\phi}_k$.

In the situation at hand the cone, containing the elements of S_A, can be spanned by the elements of a set $S_\phi = \{\bar{\phi}_1, \bar{\phi}_2, \ldots, \bar{\phi}_s\}$, that is, every $\bar{\phi}$ in S_A can be expressed as

$$\bar{\phi} = \sum_{k=1}^{s} \zeta_k \bar{\phi}_k \ , \ \zeta_k \geq 0 \ ,$$

where $\bar{\phi}_k$ is an element of S_ϕ, s is the number of elements in S_ϕ, and ζ_k is a real number. Therefore the system of linear inequalities formed by the elements of S_ϕ is the system

$$\begin{bmatrix} \bar{\phi}_1 \\ \bar{\phi}_2 \\ \vdots \\ \bar{\phi}_s \end{bmatrix} \bar{\theta}^T \gtrless \bar{0}^T \ .$$

The reason for the restriction imposed by $\gtrless$ can be explained as follows. If there exist two $\bar{\phi}_k$'s, say $\bar{\phi}_{k_1}$ and $\bar{\phi}_{k_2}$, such that $\bar{\theta}\,\bar{\phi}_{k_1}^T < 0$ and $\bar{\theta}\,\bar{\phi}_{k_2}^T > 0$, then there exists some $\bar{\phi}' = \zeta_{k_1}\bar{\phi}_{k_1} + \zeta_{k_2}\bar{\phi}_{k_2}$ such that $\bar{\theta}(\bar{\phi}')^T = 0$.

The computation of a set S_ϕ procedes by first finding a basis

$$S_x = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_d\}$$

of solutions to

$$\underline{A}^T \bar{x}^T = \bar{0}^T \qquad \text{(A.2.4)}$$

where d = m - rank ($\underline{A}$). Since every solution $\bar{x}$ can be expressed as a linear combination

$$\bar{x} = \sum_{i=1}^{d} b_i \bar{x}_i \ , \quad \bar{x}_i \epsilon S_x \ , \tag{A.2.5}$$

the restriction of $\bar{x} \geq \bar{0}$ places restrictions on the real numbers b_i, $i = 1, 2, \ldots, d$. The $\bar{b}$'s, $\bar{b} = (b_1, b_2, \ldots, b_d)$, for which

$$[\bar{x}_1^T, \bar{x}_2^T, \ldots, \bar{x}_d^T]\bar{b}^T = \underline{X}\bar{b}^T \geq \bar{0}^T$$

is satisfied, form a convex polyhedral cone as defined on page 19 of Goldman and Tucker (8). Using an extremal set

$$S_b = \{\bar{b}_1, \bar{b}_2, \ldots, \bar{b}_s\} \ ,$$

of the vectors in this cone, the corresponding set S_ϕ can be determined. The integer s is the number of extremal vectors in the set.

A basis S_x can be found easily by straightforward techniques of matrix theory as shown on page 59 of Browne (2) or by a sweepout procedure. The problem of finding an extremal set S_b is not quite as simple but it can be done using the theory of convex polyhedral cones. The fact that a knowledge of S_x and S_b permits the determination of S_ϕ is shown in the following section. It should be pointed out that a set S_b or S_ϕ is not unique.

A.3 Verification that S_x and S_b Lead to S_A. As previously defined the set S_A is

$$S_A = \{\bar{\phi} \mid \underline{A}^T \bar{\phi}^T = \bar{0} \ , \ \bar{\phi} \neq \bar{0} \ , \ \bar{\phi} \geq \bar{0}\} \tag{A.3.1}$$

and is the desired set of nonnegative, nontrivial solutions to Equation

A.2.2 for which Inequality A.2.3 must be satisfied in order for a column $\bar{\theta}^T$ to perform the desired task. A more useful specification of the elements of S_A is a specification in terms of a nonnegative, linear combination of extremal vectors for the cone. The intent in this section is to show that the specification of a set

$$S_A' = \{\bar{\phi} \mid \bar{\phi}^T = \underline{X}\bar{b}^T \ , \ \bar{b} = \sum_{k=1}^{s} \zeta_k \bar{b}_k \ , \ \zeta_k \geq 0 \ , \ \bar{b}_k \epsilon S_b \ , \ \bar{x}_i \epsilon S_x \ , \ \bar{\phi} \neq \bar{0}\} \qquad (A.3.2)$$

results in S_A' being identical to S_A; therefore the elements $\bar{\phi}_k$ in S_ϕ are given by

$$\bar{\phi}_k = \underline{X}\bar{b}_k^T \ , \quad k = 1, 2, \ldots, s \ .$$

Let the matrix $\underline{V}$ consist of the m, d-dimensional rows $\bar{v}$ of $\underline{X}$, then the set of $\bar{b}$'s in Equation A.3.2 which satisfy

$$\underline{X}\bar{b}^T = [\bar{x}_1^T \ \bar{x}_2^T \ \cdots \ \bar{x}_d^T]\bar{b}^T = \begin{bmatrix} \bar{v}_1 \\ \bar{v}_2 \\ \vdots \\ \bar{v}_m \end{bmatrix} \bar{b}^T = \underline{V}\bar{b}^T \geq \bar{0}^T \qquad (A.3.3)$$

form the set

$$S_v^* = \{\bar{b} \mid \underline{V}\bar{b} \geq \bar{0}^T\} \qquad (A.3.4)$$

known as the polar of the set

$$S_v = \{\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_m\} \ .$$

A theorem by Minkowski as presented on page 30 of Goldman and Tucker (8)

states that S_v^* is equal to the convex cone hull of the set S_b. The convex cone hull of S_b is defined as

$$S_b^< = \{\bar{b} \mid \bar{b} = \sum_{k=1}^{s} \zeta_k \bar{b}_k \ , \ \zeta_k \geq 0 \ , \ \bar{b}_k \epsilon S_b\} \ . \qquad (A.3.5)$$

Minkowski's theorem is:

Theorem A.3.1. Given a finite set S_v of vectors in the space of $\bar{v}$'s there exists a finite set S_b of vectors in the space of $\bar{b}$'s such that $S_v^* = S_b^<$.

This thoerem means that every member of the convex polyhedral cone S_v^* can be expressed as a nonnegative, linear combination of a finite set S_b of vectors. Thus every $\bar{b}$, which satisfies

$$\underline{X}\bar{b}^T = \underline{V}\bar{b}^T \geq \bar{0}^T \ ,$$

can be expressed in terms of a nonnegative, linear combination as in Equation A.3.5. As a result, the set of $\bar{\phi}$'s, such that

$$\bar{\phi}^T = \underline{X}\bar{b}^T \geq \bar{0}$$

is satisfied, can be specified as in S_A' in Equation A.3.2. The following theorem is intended to more clearly show that S_A is identical to S_A'.

Theorem A.3.2. If the sets S_A and S_A' are defined as in Equations A.3.1 and A.3.2 respectively, then $S_A = S_A'$.

Proof: Consider the implication that if $\bar{\phi}\epsilon S_A$ then $\bar{\phi}\epsilon S_A'$. If $\bar{\phi}\epsilon S_A$, then $\bar{\phi} = \sum_{i=1}^{d} b_i \bar{x}_i$, $\bar{x}_i \epsilon S_x$, for some set of b_i's, i = 1, 2, ..., d. The constraint of S_A that $\bar{\phi} \geq \bar{0}$ implies that every $\bar{b}$, which satisfies $\underline{X}\bar{b}^T \geq \bar{0}$, can be expressed as $\bar{b} = \sum_{k=1}^{s} \zeta_k \bar{b}_k$, $\zeta_k \geq 0$, $\bar{b}_k \epsilon S_b$, by Theorem

A.3.1. Finally, since $\bar{\phi} \in S_A$ implies $\bar{\phi} \neq \bar{0}$, all the constraints on the elements of S_A' are satisfied.

Consider the implication that if $\bar{\phi} \in S_A'$ then $\bar{\phi} \in S_A$. This implication is proved by contradiction. Assume that there exists a $\bar{\phi} \in S_A'$ such that $\bar{\phi} \notin S_A$. If $\bar{\phi} \notin S_A$ then three cases arise; either $\bar{\phi} \neq \sum_{i=1}^{d} b_i \bar{x}_i$, $\bar{x}_i \in S_x$, for any set of b_i's, $i = 1, 2, \ldots, d$; $\bar{\phi} = \sum_{i=1}^{d} b_i \bar{x}_i$ but $\bar{\phi} \leq \bar{0}$; or $\bar{\phi} = \bar{0}$. Each of these cases contradict some constraint on the elements of S_A'.

The problem remaining is to demonstrate a valid technique for finding the elements of a set S_b.

A.4 Determination of S_b from S_x. The discussion here is based heavily on the work done by Goldman and Tucker (8) beginning on page 19. The theory presented there is fairly general, but the application of the theory here permits some very convenient shortcuts.

It should be intuitively clear that an extremal set S_b is related in some way to some sort of boundaries of the cone S_v^*. A major conceptual difficulty in the theory surrounding the extremal vectors lies in the fact that these boundaries are, in general, contained in multidimensional linear subspaces and are not easily visualized. In order to develop the technique for finding a set S_b, some concepts must be introduced.

Consider the convex cone S_v^* defined by Equation A.3.4 or by

$$S_v^* = \{\bar{b} \mid \bar{v}_1 \bar{b}^T \geq 0, \bar{v}_2 \bar{b}^T \geq 0, \ldots, \bar{v}_m \bar{b}^T \geq 0\}$$

and the empty or nonempty subset S_H of the subscripts of the vectors $\bar{v}_q$; that is, $S_H = \{q_1, q_2, \ldots, q_t\}$ where $t < m$ is arbitrary. Based on these definitions a "face" S_{F_H} of the convex cone S_v^* is the subset of S_v^*

whose members satisfy both

$$\bar{v}_q \bar{b}^T > 0 \ , \quad q \varepsilon S_H \qquad (A.4.1)$$

and

$$\bar{v}_q \bar{b}^T = 0 \ , \quad q \notin S_H \ . \qquad (A.4.2)$$

The set of $\bar{b}$'s satisfying Inequality A.4.1 is an open set S_{L_H} while the set satisfying Equation A.4.2 is a linear subspace S_{Γ_H}. The face S_{F_H} is the intersection of the corresponding sets. The subspace S_{Γ_H} has a dimension given by

$$d_H = d - r_H$$

where d is the number of components of each vector $\bar{v}_q$ and r_H is the rank of the matrix composed of the vectors $\bar{v}_q$ for $q \notin S_H$. The fact that S_{Γ_H} has dimension d_H means that the set of equations defined by Equation A.4.2 has a solution space of dimension d_H. The face S_{F_H} is also said to have dimension d_H.

Notice that the minimum value of d_H for a given value of d is determined by the maximum value of r_H which is the rank r of the matrix $\underline{V}$. If $r_H = r$ then the minimum value of d_H (call it d_{min}) is $d_{min}=d-r$. Corresponding to the minimum value of d_H there is a unique face S_{F_H} as determined by the homogeneous system of equations

$$\underline{V}\bar{b}^T = \bar{0}^T$$

where S_H is empty since Inequality A.4.1 is not involved in defining the face. The reason that some nonempty set S_H will not produce a face of the same dimension, when there are r linearly independent vectors $\bar{v}_q$ for

$q \notin S_H$, is that every vector $\bar{v}_q$, $q \in S_H$, can be expressed as a linear combination of the vectors $\bar{v}_q$, $q \notin S_H$. Therefore a vector $\bar{b}$ in S_{Γ_H} cannot satisfy Inequality A.4.1 if it satisfies Equation A.4.2. The resulting fact that there exists only one face of dimension d_{min} is very important when coupled with the following theorem and corollary from Goldman and Tucker (8).

Theorem A.4.1. S_v^* is either just its unique d_{min}-face or the convex hull of this d_{min}-face and the $(d_{min} + 1)$-dimensional faces.

It is very significant that in the particular problem considered here the rank of $\underline{V}$ is identically d which means that $d_{min} = 0$. The rank of $\underline{V}$ is d because $\underline{V}$ consists of $d \leq m$ linearly independent columns of dimension m from $\underline{X}$ which is a matrix of basis vectors. As a result of d_{min} being zero the d_{min}-dimensional face S_{Γ_0} (0 means the set S_H is empty) is the zero dimensional space or vertex of the cone; furthermore the $(d + 1)$-dimensional faces are one dimensional half-lines called "edges" of S_v^* radiating from the vertex. A corollary to Theorem A.4.1 is:

Corollary A.4.1. If the rank of $\underline{V}$ is d then S_v^* is either just $\bar{0}$ or the convex hull of $\bar{0}$ and the edges of S_v^*.

The task of finding the elements of a set S_b is now reduced to computing edges of S_v^*. Goldman and Tucker (8) describe the procedure for computing these edges. The procedure is simple but quite lengthy since it involves performing a test on all possible combinations of $r_H = d - 1$ linearly independent rows from the m rows of $\underline{V}$. There are $\binom{m}{d-1}$ rows to be tested for linear independence. Given a set of $d - 1$ linearly

independent rows from $\underline{V}$, these rows are used to form a system of d - 1 equations in d unknowns. Since these rows are linearly independent a one dimensional solution space exists and a basis vector $\bar{b}_k$ can be computed by conventional matrix theory techniques. If the basis vector $\bar{b}_k$ of the solution space satisfies $\underline{V}\bar{b}_k^T \geq \bar{0}$, then $\zeta\bar{b}_k$, $\zeta > 0$, is an edge and $\bar{b}_k$ is an element of S_b. If $\bar{b}_k$ satisfies $\underline{V}\bar{b}_k^T \leq \bar{0}$, then $\zeta(-\bar{b}_k)$ is an edge and $-\bar{b}_k$ is an element of S_b. If all possible combinations of d - 1 rows of $\underline{V}$ are selected and if the above operations are performed, this procedure will yield a set S_b. Obviously if S_b is empty, then the only solution to $\underline{V}\bar{b}^T \geq \bar{0}$ is $\bar{b} = \bar{0}$.

A.5 Determination of S_ϕ from S_x and S_b. Since S_A and S_A' are equal, every $\bar{\phi}$ in S_A can be expressed as a linear sum

$$\bar{\phi} = \underline{X}\bar{b}^T \ , \ \bar{b} = \sum_{k=1}^{s} \zeta_k \bar{b}_k \ , \ \bar{b}_k \epsilon S_b \ , \ \zeta_k \geq 0 \ ,$$

as in S_A'. Thus every $\bar{\phi}$ in S_A is given by

$$\bar{\phi} = \sum_{k=1}^{s} \zeta_k \underline{X}\bar{b}_k^T \ , \ \bar{b}_k \epsilon S_b \ , \ \zeta_k \geq 0 \ ,$$

or

$$\bar{\phi} = \sum_{k=1}^{s} \zeta_k \bar{\phi}_k \ , \ \zeta_k \geq 0 \ ,$$

where

$$\bar{\phi}_k = \underline{X}\bar{b}_k^T \ , \ \bar{b}_k \epsilon S_b \ , \quad k = 1, 2, \ldots, s \ .$$

A desired extremal set of the vectors in the convex polyhedral cone of S_A is given by

$$S_\phi = \{\underline{Xb}_1^T, \underline{Xb}_2^T, \ldots, \underline{Xb}_s^T\} \quad .$$

A.6 The Inequality Constraints on $\bar{\theta}$. From Theorem A.2.1 (Gordon's Theorem) it can be seen that in order for

$$[\underline{A} \mid \bar{\theta}^T]\bar{w}_1^T > \bar{0}^T$$

to have a solution $\bar{w}_1$ when

$$\underline{A}_1 \bar{w}_o^T > \bar{0}^T$$

has no solution, $\bar{\theta}$ must satisfy

$$\begin{bmatrix} \bar{\phi}_1 \\ \bar{\phi}_2 \\ \vdots \\ \bar{\phi}_s \end{bmatrix} \bar{\theta}^T \gtrless \bar{0}^T$$

where $\bar{\phi}_k \varepsilon S_\phi$, $k = 1, 2, \ldots, s$.

A.7 Computer Program Flow-Chart for Computing S_ϕ or for LS Testing. The following flow-chart is for a computer program which computes a set S_ϕ from the matrix $\underline{A}$ in Inequalities A.1.1, A.1.2, or A.1.3. If it turns out that S_ϕ is empty for some $\underline{A}$, the patterns corresponding to the rows of $\underline{A}$ are LS; thus a test for linear separability of patterns is an alternate use of this program.

In this chart the diamond-shaped symbol signifies a decision step. If the answer to the question is "yes," the logic flow is out the right or left side. If the answer is "no," the logic flow is out the bottom of the symbol. All input or output statements are explained in the

listing in Section A.8.

A.8 Computer Program Listing for Computing S_ϕ or for LS Testing.

Based on the flow-chart in Figure A.7.1 the following computer program in FORTRAN IV language can be used to compute the elements of S_ϕ or for LS testing. COMMENT statements are inserted in the listing in locations which correspond roughly to the input to blocks in the flow-chart. The variables in the program (except in the COMMENT statements) do not necessarily correspond to the notation in the previous sections of this Appendix.

The running characteristics of this program are highly dependent upon specific problems. For example, the running time cannot be accurately predicted because some problems may permit certain phases to be skipped more frequently than in other problems. In general it can be said that the greatest limitation of this program is the size of the number $\binom{m}{m-1-\text{rank}(\underline{A})}$ where m is the number of rows in $\underline{A}$. The reason for this limitation is that the basic iteration in the program must be performed this number of times. Also as m gets large the length of time required for each iteration increases. Each iteration involves sweeping out a submatrix of size $[m-1-\text{rank}(\underline{A})]\times[m-\text{rank}(\underline{A})]$ if the submatrix is of maximum rank. If it is not of maximum rank, the particular iteration is shortened in time.

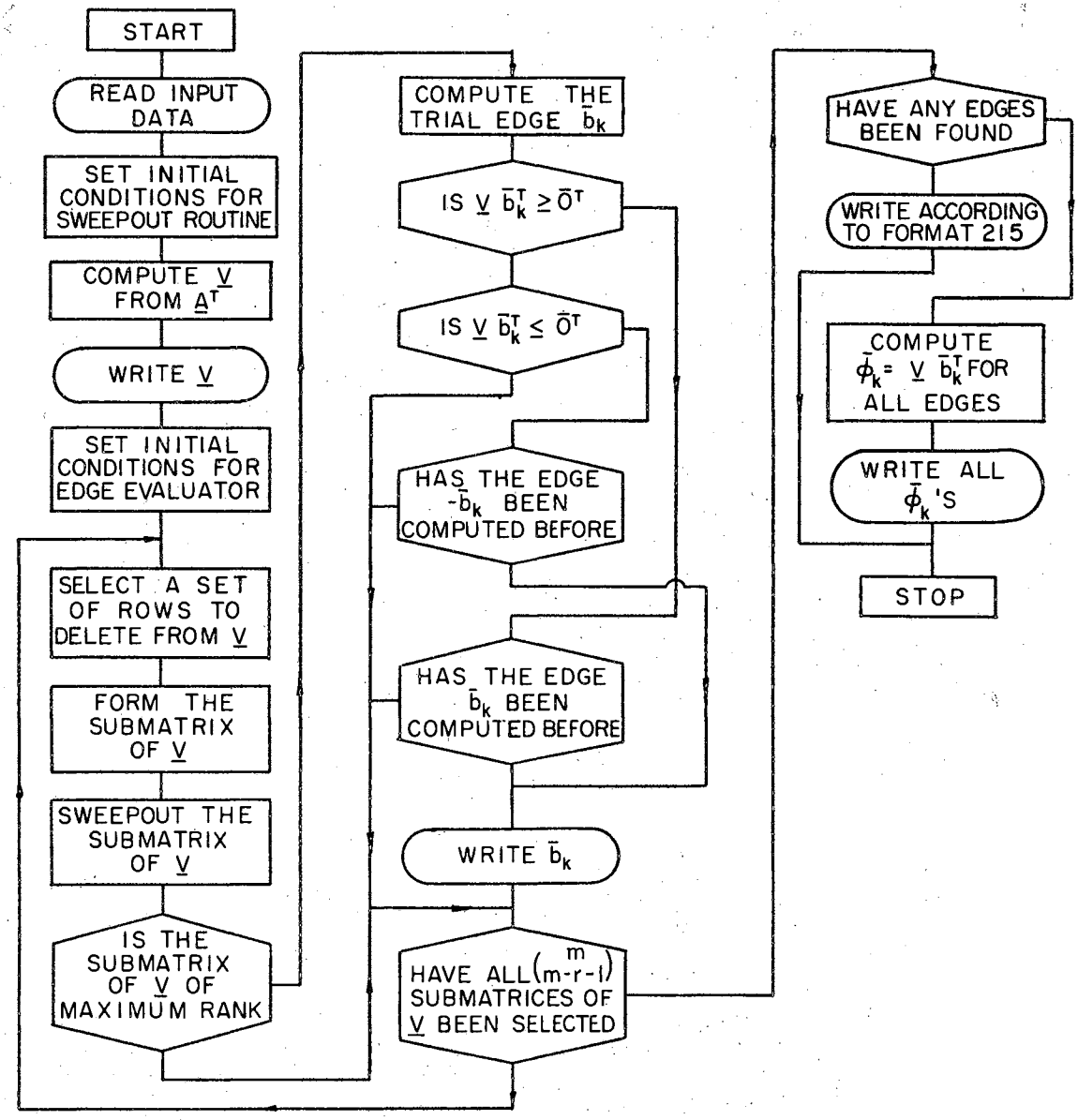


Figure A.7.1. Flow-Chart for Computing S_ϕ and LS Testing

```
C     THIS PROGRAM COMPUTES A SET OF EXTREMAL VECTORS FOR A CONVEX
C     POLYHEDRAL CONE OR TESTS FOR LINEAR SEPARABILITY OF THE CORRES-
C     PONDING PATTERN VECTORS.
      DIMENSION A(5,14),INDPA(14),IDPA(5),AA(14)
      DIMENSION P(9,10),V(14,10),PP(10),B(10),BASIS(25,10),Z(14)
      LOGICAL BNC(15), BNR(15)
      LOGICAL ZERO, POS, POS1, AN
C     READ INPUT DATA
C     THE FIRST CARD CONTAINS NUMBER OF ROWS AND COLUMNS IN MATRIX A
C     ACCORDING TO FORMAT 100.
C     THE NEXT M*N CARDS CONTAIN THE ELEMENTS OF MATRIX A.  ENTER ONE
C     ELEMENT PER CARD BY ROWS ACCORDING TO FORMAT 101.
      READ (5,100) IROW,ICOL
100   FORMAT(2I5)
      READ (5,101) ((A(I,J),I=1,IROW),J=1,ICOL)
101   FORMAT(F10.0)
C     SET INITIAL CONDITIONS FOR SWEEPOUT ROUTINE
      M = ICOL
      IDPA1 = 0
      INDPA1 = 0
      IROW1 = IROW - 1
      L = 1
      JJ = 1
C     COMPUTE MATRIX V FROM MATRIX A TRANSPOSE
1000  IT = 0
1001  IT = IT + 1
      IF(A(L,JJ)) 1003,1002,1003
1002  IF(IT - IROW + L - 1) 1004,1005,1005
1004  DO 1006 J = 1,ICOL
1006  AA(J) = A(L,J)
      DO 1007 L1 = L,IROW1
      DO 1007 J = 1,ICOL
1007  A(L1,J) = A(L1 + 1,J)
      DO 1008 J = 1,ICOL
1008  A(IROW,J) = AA(J)
      GO TO 1001
1005  INDPA1 = INDPA1 + 1
      INDPA(INDPA1) = JJ
      IF(JJ - ICOL) 1009,1010,1010
1009  JJ = JJ + 1
      GO TO 1000
1003  IDPA1 = IDPA1 + 1
      IDPA(IDPA1) = JJ
      DO 1011 L1 = 1,IROW
      IF(L1 - L) 1012,1011,1012
1012  CONST = A(L1,JJ)/A(L,JJ)
      DO 1021 J = JJ,ICOL
1021  A(L1,J) = A(L1,J) - CONST*A(L,J)
1011  CONTINUE
      DIV = A(L,JJ)
      DO 1013 J = JJ,ICOL
1013  A(L,J) = A(L,J)/DIV
      IF(L - IROW) 1014,1015,1015
1014  IF(JJ - ICOL) 1017,1010,1010
1017  JJ = JJ + 1
      L = L + 1
      GO TO 1000
1015  JJ = JJ + 1
      DO 1016 J = JJ,ICOL
      INDPA1 = INDPA1 + 1
1016  INDPA(INDPA1) = J
1010  DO 1018 I1 = 1,IDPA1
```

```
      I = IDPA(I1)
      DO 1018 J1 = 1,INDPA1
      J = INDPA(J1)
1018  V(I,J1) = -A(I1,J)
      DO 1022 J1 = 1,INDPA1
      J = INDPA(J1)
      DO 1022 J11 = 1,INDPA1
      IF(J11 - J1) 1019,1020,1019
1020  V(J,J11) = 1.0
      GO TO 1022
1019  V(J,J11) = 0.0
1022  CONTINUE
C     WRITE MATRIX V
      WRITE (6,6600)
6600  FORMAT(1H ,49HTHE MATRIX V, WHOSE COLUMNS ARE BASIS VECTORS FOR/56
     1H THE SOLUTIONS TO A(TRANSPOSE)*X(TRANSPOSE) = 0, FOLLOWS/)
      DO 1023 I = 1,ICOL
1023  WRITE (6,220) (V(I,J),J = 1,INDPA1)
220   FORMAT(1H ,9(2X,E12.5))
C     SET INITIAL CONDITIONS FOR EDGE EVALUATOR
      WRITE (6,6601)
6601  FORMAT(1H1,23HTHE UNIQUE EDGES FOLLOW/)
      N = INDPA1
      ISET = 0
      IA = 0
      IB = 0
      N1 = N - 1
      N2 = N - 2
      MFAC = 1
      DO 1 M1 = 1,M
1     MFAC = MFAC*M1
      N1FAC = 1
      DO 2 N11 = 1,N1
2     N1FAC = N1FAC*N11
      MNFAC = 1
      MN = M - N1
      DO 3 MN1 = 1,MN
3     MNFAC = MNFAC*MN1
      IR = MFAC/(N1FAC*MNFAC)
      NOBNO = M
      NOBTK = N - 1
      MM1 = NOBNO + 1
      IA = 0
      DO 2000 I = 2,MM1
      BNC(I) = .FALSE.
2000  BNR(I) = .FALSE.
      BNC(1) = .FALSE.
      BNR(1) = .TRUE.
C     SELECT A SET OF ROWS TO DELETE FROM MATRIX V
4     CONTINUE
2001  ICT = 0
      IA = IA + 1
      DO 2002 I = 2,MM1
      BNR(I) = BNC(I).AND.BNR(I-1)
      BNC(I) = ((.NOT.BNC(I)).AND.BNR(I-1)).OR.(BNC(I).AND.(.NOT.BNR(I-1
     1)))
      IF(BNR(I)) GO TO 2002
2003  ICT = ICT + 1
      GO TO 2004
2002  CONTINUE
      GO TO 55
2004  IF(ICT - NOBTK) 2006,2006,2001
```

```
2006  IF(I - MM1) 2013,2001,2001
2013  IBN1 = I + 1
      DO 2008 IBN = IBN1,MM1
      IF(BNC(IBN)) GO TO 2009
      GO TO 2008
2009  ICT = ICT + 1
      IF(ICT - NOBTK) 2008,2008,2001
2008  CONTINUE
      IF(ICT - NOBTK) 2001,2010,2001
2010  CONTINUE
C     FORM THE SUBMATRIX OF MATRIX V
9     L = 0
      DO 10 K = 1,M
      IF(BNC(K + 1)) GO TO 11
      GO TO 10
11    L = L + 1
      DO 36 J = 1,N
36    P(L,J) = V(K,J)
10    CONTINUE
C     SWEEPOUT THE SUBMATRIX OF MATRIX V AND TEST TO SEE IF SUBMATRIX IS
C     OF MAXIMUM RANK
      ISET = ISET + 1
      IID1 = 0
      JJ = 1
      L = 1
13    IT = 0
14    IT = IT + 1
      IF(P(L,JJ)) 15,16,15
16    IF(IT - N1 + L - 1) 17,18,18
17    DO 34 J = 1,N
34    PP(J) = P(L,J)
      DO 19 L1 = L,N2
      DO 19 J = 1,N
19    P(L1,J) = P(L1 + 1,J)
      DO 35 J = 1,N
35    P(N1,J) = PP(J)
      GO TO 14
18    IID1 = IID1 + 1
      IF(IID1 - 1) 20,20,50
20    IID = JJ
      IF(JJ - N) 22,23,23
22    JJ = JJ + 1
      GO TO 13
15    DO 24 L1 = 1,N1
      IF(L1 - L) 25,24,25
25    CONST = P(L1,JJ)/P(L,JJ)
      DO 33 J = JJ,N
33    P(L1,J) = P(L1,J) - CONST*P(L,J)
24    CONTINUE
      DIV = P(L,JJ)
      DO 32 J = JJ,N
32    P(L,J) = P(L,J)/DIV
      IF(JJ - N) 26,23,23
26    JJ = JJ + 1
      IF(JJ - N) 27,57,57
27    L = L + 1
      GO TO 13
57    IF(IID1 - 1) 60,27,27
60    IID = N
23    IIDL = IID - 1
C     COMPUTE THE TRIAL EDGE B(K)
      IF(IIDL) 62,61,62
```

```
62      DO 28 J = 1,IIDL
28      B(J) = -P(J,IID)
61      B(IID) = 1.0
        IF(IID - N) 29,31,31
29      IIDU = IID + 1
        DO 30 J = IIDU,N
30      B(J) = 0.0
31      IP = 1
C       TEST TO SEE IF B(K) IS AN EDGE
        ZERO = .TRUE.
        DO 39 L1 = 1,M
        F = 0.0
        DO 38 J = 1,N
38      F = F + V(L1,J)*B(J)
        IF(F) 40,39,41
40      POS = .FALSE.
        GO TO 42
41      POS = .TRUE.
42      IF(IP - 1) 44,44,43
44      POS1 = POS
        IP = 2
        ZERO = .FALSE.
        GO TO 39
43      AN = POS.AND.POS1
        IF(AN) GO TO 39
        GO TO 50
39      CONTINUE
46      IF(ZERO) GO TO 50
        IF(POS1) GO TO 49
48      DO 58 J = 1,N
58      B(J) = -B(J)
C       TEST TO SEE IF B(K) HAS BEEN COMPUTED BEFORE
49      IF(IB) 71,70,71
71      DO 72 IB1 = 1,IB
        DO 73 J = 1,N
        TEST = BASIS(IB1,J) - B(J)
        IF(TEST) 72,73,72
73      CONTINUE
        GO TO 50
72      CONTINUE
70      IB = IB + 1
        DO 74 J = 1,N
74      BASIS(IB,J) = B(J)
C       WRITE B(K)
        WRITE (6,210) IB,ISET
210     FORMAT(1H0,9HEDGE NO. ,I3,20H FROM SUBMATRIX NO. ,I5,8H FOLLOWS/)
        WRITE (6,211) (BASIS(IB,J),J = 1,N)
211     FORMAT(1H ,9(2X,E12.5))
C       HAVE ALL SUBMATRICES OF MATRIX V BEEN SELECTED
50      IF(ISET - IR) 4,55,55
55      WRITE (6,214) IB
214     FORMAT(1H1,4HTHE ,I3,24H EXTREMAL VECTORS FOLLOW/)
C       HAVE ANY EDGES BEEN FOUND
        IF(IB) 80,81,80
C       WRITE THAT PATTERNS ARE LINEARLY SEPARABLE
81      WRITE (6,215)
215     FORMAT(1H ,35HTHE PATTERNS ARE LINEARLY SEPARABLE)
        GO TO 82
C       COMPUTE AND WRITE THE EXTREMAL VECTORS
80      DO 75 IB1 = 1,IB
        DO 76 I = 1,M
        Z(I) = 0.0
```

```
      DO 76 J = 1,N
76    Z(I) = Z(I) + V(I,J)*BASIS(IB1,J)
      WRITE (6,212) IB1
212   FORMAT(1H0,32HTHE ELEMENTS OF EXTREMAL VECTOR ,I3,7H FOLLOW/)
      WRITE (6,213) (Z(I), I = 1,M)
213   FORMAT(1H ,9(2X,E12.5))
75    CONTINUE
82    STOP
      END
```

APPENDIX B

THE EXTREMAL SET EXTENSION TECHNIQUE

B.1 Introduction. Consider an m x n matrix

$$\underline{A}' = \begin{bmatrix} \overline{a}_1 \\ \overline{a}_2 \\ \vdots \\ \overline{a}_m \end{bmatrix}$$

and an extremal set

$$S_\phi' = \{\overline{\phi}_1', \overline{\phi}_2', \ldots, \overline{\phi}_s'\}$$

of vectors with m components

$$\overline{\phi}' = (\phi_{k1}', \phi_{k2}', \ldots, \phi_{km}')$$

for the set of all solutions $\overline{\phi}'$ to

$$(\underline{A}')^T(\overline{\phi}')^T = \overline{0}^T \ , \quad \overline{\phi}' \geq \overline{0} \ , \quad \overline{\phi}' \neq \overline{0} \ . \qquad \text{(B.1.1)}$$

The prime notation on $\underline{A}'$ is not related to that on $\underline{A}_i'$ in Chapters III and IV. The problem here is to determine the effect of adding a row $\overline{a}_{m+1}$ to $\underline{A}'$ resulting in

$$\underline{A} = \begin{bmatrix} \bar{a}_1 \\ \bar{a}_2 \\ \vdots \\ \bar{a}_m \\ \bar{a}_{m+1} \end{bmatrix}$$

where $\bar{a}_{m+1}$ is equal to some row in $\underline{A}'$. Without loss of generality it can be assumed that $\bar{a}_{m+1} = \bar{a}_m$. The resulting effect is that an extremal set

$$S_\phi = \{\bar{\phi}_1, \bar{\phi}_2, \ldots, \bar{\phi}_{s+r}\}$$

of extremal vectors for the set of all solutions $\bar{\phi}$ to

$$A^T\bar{\phi}^T = \bar{0}^T \ , \quad \bar{\phi} \geq \bar{0} \ , \quad \bar{\phi} \neq 0 \ , \qquad \text{(B.1.2)}$$

consists of s vectors

$$\bar{\phi}_k = (\phi'_{k1}, \phi'_{k2}, \ldots, \phi'_{km}, 0) \ , \quad k = 1, 2, \ldots, s$$

plus r vectors of the form

$$\bar{\phi}_k = (\phi'_{k1}, \phi'_{k2}, \ldots, \phi'_{k(m-1)}, 0, \phi'_{k(m+1)}), \quad k = s+1, \ldots, s+r \ .$$

The integer r is the number of nonzero components ϕ'_{km} in the mth positions of the vectors $\bar{\phi}'_k$, k = 1, 2, ..., s.

B.2 The Argument for the Validity of the Technique. The approach to this problem is to show that every solution to Equation B.1.2 can be expressed in terms of a nonnegative, linear combination of the elements

in S_ϕ. Rewriting Equation B.1.2 as

$$\bar{a}_1^T\phi_1 + \bar{a}_2^T\phi_2 + \cdots + \bar{a}_m^T\phi_m + \bar{a}_{m+1}^T\phi_{m+1} = \bar{0}^T,\ \phi_i \geq 0,\ i = 1, 2, \ldots, m+1, \tag{B.2.1}$$

it can be seen that since $\bar{a}_m = \bar{a}_{m+1}$ it simplifies to

$$\bar{a}_1^T\phi_1 + \bar{a}_2^T\phi_2 + \cdots + \bar{a}_m^T(\phi_m + \phi_{m+1}) = \bar{0}^T,\ \phi_i \geq 0,\ i = 1, 2, \ldots, m+1. \tag{B.2.2}$$

Equation B.2.2 is identical to Equation B.1.1 except that different notation is used; therefore, every solution $\bar{\phi}'$,

$$(\bar{\phi}')^T = \begin{bmatrix} \phi_1' \\ \phi_2' \\ \vdots \\ \phi_m' \end{bmatrix} \quad \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_m + \phi_{m+1} \end{bmatrix},$$

can be expressed as

$$\begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_m + \phi_{m+1} \end{bmatrix} = \sum_{k=1}^{s} \alpha_k' \begin{bmatrix} \phi_{k1}' \\ \phi_{k2}' \\ \vdots \\ \phi_{km}' \end{bmatrix},\ \alpha_k' \geq 0. \tag{B.2.3}$$

From the last element in the vector given by Equation B.2.3 it can be seen that for every element ϕ_m' of a vector $\bar{\phi}'$ the solutions for ϕ_m and ϕ_{m+1} form a one parameter family; that is, ϕ_m and ϕ_{m+1} in

$$\phi_m + \phi_{m+1} = \sum_{k=1}^{s} \alpha_k' \phi_{km}' \tag{B.2.4}$$

can be written as

$$\begin{bmatrix} \phi_m \\ \phi_{m+1} \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{s} \alpha_k' \phi_{km}' \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix} \beta \qquad \text{(B.2.5)}$$

where

$$0 \leq \beta \leq \sum_{k=1}^{s} \alpha_k' \phi_{km}' \ , \ \forall \beta \ . \qquad \text{(B.2.6)}$$

The reason for Inequality B.2.6 is that ϕ_m and ϕ_{m+1} in Equation B.2.4 must be nonnegative.

By introducing the real numbers $\beta_1, \beta_2, \ldots, \beta_s$, β can be more conveniently written as

$$\beta = \sum_{k=1}^{s} \beta_k \ , \quad 0 \leq \beta_k \leq \alpha_k' \phi_{km}' \ , \ \forall k \qquad \text{(B.2.7)}$$

thus permitting Equation B.2.5 to be written in the form

$$\begin{bmatrix} \phi_m \\ \phi_{m+1} \end{bmatrix} = \sum_{k=1}^{s} \left\{ \alpha_k' \begin{bmatrix} \phi_{km}' \\ 0 \end{bmatrix} + \beta_k \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right\} . \qquad \text{(B.2.8)}$$

Since the desired result is a set S_ϕ of extremal vectors for the solutions $\bar{\phi}$ to Equation B.2.1, Equations B.2.3 and B.2.8 can be combined to give

$$\bar{\phi}^T = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{m-1} \\ \phi_m \\ \phi_{m+1} \end{bmatrix} = \sum_{k=1}^{s} \left\{ \alpha_k' \begin{bmatrix} \phi_{k1}' \\ \phi_{k2}' \\ \vdots \\ \phi_{k(m-1)}' \\ \phi_{km}' \\ 0 \end{bmatrix} + \beta_k \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ -1 \\ 1 \end{bmatrix} \right\} . \qquad \text{(B.2.9)}$$

Notice that each vector term in the summation is actually an infinite number of vectors because β_k is arbitrary within the bounds expressed in Equation B.2.7. As a result, each of the families of vectors (terms in the summation) can be written as

$$\alpha_k' \begin{bmatrix} \phi_{k1}' \\ \phi_{k2}' \\ \vdots \\ \phi_{k(m-1)}' \\ \phi_{km}' \\ 0 \end{bmatrix} + \beta_k \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ -1 \\ 1 \end{bmatrix} = \alpha_k'' \begin{bmatrix} \phi_{k1}' \\ \phi_{k2}' \\ \vdots \\ \phi_{k(m-1)}' \\ \phi_{km}' \\ 0 \end{bmatrix} + \alpha_k''' \begin{bmatrix} \phi_{k1}' \\ \phi_{k2}' \\ \vdots \\ \phi_{k(m-1)}' \\ 0 \\ \phi_{km}' \end{bmatrix} \qquad \text{(B.2.10)}$$

where

$$\alpha_k'' + \alpha_k''' = \alpha_k' \geq 0 \ , \quad \alpha_k'' \geq 0 \ , \quad \alpha_k''' \geq 0 \ . \qquad \text{(B.2.11)}$$

This change is permitted because of the upper and lower bounds placed on β_k. Substituting Equation B.2.10 back into Equation B.2.9 the result is

$$\underline{\phi}^T = \sum_{k=1}^{s} \alpha_k'' \begin{bmatrix} \phi_{k1}' \\ \phi_{k2}' \\ \vdots \\ \phi_{k(m-1)}' \\ \phi_{km}' \\ 0 \end{bmatrix} + \sum_{k=1}^{s} \alpha_k''' \begin{bmatrix} \phi_{k1}' \\ \phi_{k2}' \\ \vdots \\ \phi_{k(m-1)}' \\ 0 \\ \phi_{km}' \end{bmatrix} . \qquad \text{(B.2.12)}$$

There are two points which need to be discussed before continuing. First, the relationship between α_k', α_k'', and α_k''' in Equation B.2.11 is not a restrictive constraint on their values when they are used in Equation

B.2.12. The reason for this is that it is sufficient to know that there exists some nonnegative, linear combination of vectors which equals a given vector $\bar{\phi}$. Second, if $\phi'_{km} = 0$, then

$$\begin{bmatrix} \phi'_{k1} \\ \phi'_{k2} \\ \vdots \\ \phi'_{k(m-1)} \\ \phi'_{km} \\ 0 \end{bmatrix} = \begin{bmatrix} \phi'_{k1} \\ \phi'_{k2} \\ \vdots \\ \phi'_{k(m-1)} \\ 0 \\ \phi'_{km} \end{bmatrix} ;$$

therefore, only one of the vectors for which $\phi'_{km} = 0$ needs to be included in a set S_ϕ of extremal vectors in Equation B.2.12. If there are r vectors in S'_ϕ for which $\phi'_{km} \neq 0$, then there are s + r vectors in S_ϕ.

B.3 The Extension Technique. Based on the previous development it is possible to write any solution $\bar{\phi}$ to Equation B.1.2 as

$$\bar{\phi}^T = \sum_{k=1}^{s+r} \alpha_k \bar{\phi}_k^T , \quad \alpha_k \geq 0 ,$$

where

$$\bar{\phi}_k = (\phi'_{k1}, \phi'_{k2}, \ldots, \phi'_{k(m-1)}, \phi'_{km}, 0) , \quad k = 1, \ldots, s, \tag{B.3.1}$$

and

$$\bar{\phi}_k = (\phi'_{i1}, \phi'_{i2}, \ldots, \phi'_{i(m-1)}, 0, \phi'_{im}), \quad k = s+1, \ldots, s+r. \tag{B.3.2}$$
$$\phi'_{im} \neq 0$$

In Equation B.3.2 a vector $\bar{\phi}_k$ is generated for each value of i for which $\phi'_{im} \neq 0$. Therefore the extremal set

$$S_\phi = \{\bar{\phi}_1, \bar{\phi}_2, \ldots, \bar{\phi}_{s+r}\}$$

can be formed from the set S'_ϕ by the procedure indicated by Equations B.3.1 and B.3.2.

B.4 The Effect of Repetition of the Rows of $\underline{A}^*_p$. In Chapter IV it is stated that if $\underline{A}^*_p$ has no PLD's then any matrix obtained from $\underline{A}^*_p$ by repeating its rows contains no PLD's. It is shown here without loss of generality that, if a matrix

$$\underline{A} = \begin{bmatrix} \bar{a}_1 \\ \bar{a}_2 \\ \vdots \\ \bar{a}_m \\ \bar{a}_{m+1} \end{bmatrix} \qquad \text{(B.4.1)}$$

is obtained from

$$\underline{A}' = \begin{bmatrix} \bar{a}_1 \\ \bar{a}_2 \\ \vdots \\ \bar{a}_m \end{bmatrix} \qquad \text{(B.4.2)}$$

by repeating the row $\bar{a}_m$ as $\bar{a}_{m+1}$, then $\underline{A}$ contains no PLD's if $\underline{A}'$ contains none. This is actually a trivial implication.

Theorem B.4.1. If the matrix $\underline{A}'$ in Equation B.4.1 has no PLD's between its rows, then the matrix $\underline{A}$ has no PLD's where

$$\underline{A} = \begin{bmatrix} \underline{A}' \\ \hline \bar{a}_{m+1} \end{bmatrix}$$

and where $\bar{a}_{m+1} = \bar{a}_m$.

Proof: (By contradiction) Assume that the addition of the row $\bar{a}_{m+1}$ to $\underline{A}'$ implies that there exists a vector

$$\bar{\phi} = (\phi_1, \phi_2, \cdots, \phi_{m+1}) \ , \quad \phi_i \geq 0 \ , \quad \bar{\phi} \neq 0 \ ,$$

such that

$$\bar{a}_1^T\phi_1 + \bar{a}_2^T\phi_2 + \cdots + \bar{a}_m^T\phi_m + \bar{a}_{m+1}^T\phi_{m+1} = \bar{0}^T \ .$$

Then

$$\bar{a}_1^T\phi_1 + \bar{a}_2^T\phi_2 + \cdots + \bar{a}_m^T(\phi_m + \phi_{m+1}) = \bar{0}^T \ , \quad \phi_i \geq 0 \ , \quad \bar{\phi} \neq \bar{0} \ ,$$

since $\bar{a}_m = \bar{a}_{m+1}$. This implies that there exists a PLD among the rows of $\underline{A}'$ which contradicts the hypothesis.

APPENDIX C

EXTREMAL SET FOR AN mxl MATRIX

C.1 Introduction. In the computer program of Section D.2 it is necessary to compute extremal vectors for the solutions to

$$[\bar{c}_1 \mid \mu]\underline{D}'_{\gamma 1}\bar{\lambda}^T = 0 \ , \quad \bar{\lambda} \neq \bar{0} \ , \quad \bar{\lambda} \geq \bar{0} \ ,$$

or to

$$[\bar{c}_i \mid \mu]\underline{D}'_{\gamma i}\underline{\Lambda}^T_{\gamma(i-1)}\bar{w}^T = 0 \ , \quad \bar{w} \neq 0 \ , \quad \bar{w} \geq \bar{0} \ ,$$
$$i = 2, 3, \ldots, \gamma-1 \ .$$

Each of the matrices

$$[\bar{c}_1 \mid \mu]\underline{D}'_{\gamma 1}$$

and

$$[\bar{c}_i \mid \mu]\underline{D}'_{\gamma i}\underline{\Lambda}^T_{\gamma(i-1)} \ , \quad i = 2, 3, \ldots, \gamma-1 \ ,$$

have only one row which makes this calculation easy. The purpose of this appendix is to present and verify the method for finding the PLD's or extremal vectors for a general 1 x m matrix.

For the purposes here the 1 x m matrix

$$\underline{A}^T = (a_1, a_2, a_3, \ldots, a_m)$$

is investigated by applying the theory developed in Appendix A in the

same manner that it is applied in the computer program of Appendix A. This should also help in the understanding of the theory in Appendix A. The notation used there is used here also.

C.2 Derivation of $\underline{X}$. The matrix $\underline{X}$ whose columns are basis vectors for the solutions $\bar{x}$ to

$$\underline{A}^T \bar{x}^T = \bar{0}^T$$

can be found by a sweepout routine. For the case here of only one row the routine is trivial. Nevertheless, if a_1 is assumed to be nonzero, $\underline{X}$ is given by

$$\underline{X} = \begin{bmatrix} -\frac{a_2}{a_1} & -\frac{a_3}{a_1} & \cdots & -\frac{a_m}{a_1} \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{m\times(m-1)} . \qquad \text{(C.2.1)}$$

Recall that every $\bar{x}$, such that

$$\underline{A}^T \bar{x}^T = \bar{0}^T$$

is satisfied, can be written as

$$\bar{x} = \underline{X}\bar{b}^T$$

where

$$\bar{b} = (b_1, b_2, \ldots, b_{m-1}) .$$

As in Appendix A the set S_b of extremal vectors for the solutions $\bar{b}$ to

$$\underline{X}\bar{b}^T \geq \bar{0}$$

can now be found.

C.3 Derivation of S_b. The matrix $\underline{V}$ whose rows $\bar{v}_q$, $q = 1, 2, \ldots, m$, are the rows of $\underline{X}$ is used in the procedure for finding a set S_b as indicated in Appendix A. These rows are

$$\bar{v}_1 = \left(-\frac{a_2}{a_1}, -\frac{a_3}{a_1}, \ldots, -\frac{a_m}{a_1}\right)$$

and

$$\bar{v}_i = (0, 0, \ldots, 0, 1, 0, \ldots, 0) , \quad i = 2, \ldots, m,$$

where the element 1 is in the (i-1)st position of $\bar{v}_i$, $i = 2, \ldots, m$.

Define the matrix $\underline{V}_{jk}$ as the submatrix of $\underline{V}$ from which the rows $\bar{v}_j$ and $\bar{v}_k$ are removed. If a vector $\bar{b}_\ell$ satisfies

$$\underline{V}_{jk}\,\bar{b}_\ell^T = \bar{0}^T$$

and

$$\underline{V}\,\bar{b}_\ell^T \geq \bar{0}^T ,$$

then $\bar{b}_\ell$ is an element of S_b. If $\bar{b}_\ell$ satisfies

$$\underline{V}_{jk}\,\bar{b}_\ell^T = \bar{0}^T$$

and

$$\underline{V}\,\bar{b}_\ell^T \leq \bar{0}^T$$

then $-\bar{b}_\ell$ is an element of S_b. The intent here is to determine how many

unique elements are in S_b for a given vector $\bar{v}_1$. The problem is broken into two cases, one where $\bar{v}_1$ is a row of $\underline{V}_{jk}$ and one where $\bar{v}_1$ is not a row of $\underline{V}_{jk}$.

Case I: Consider the selection of a set of rows from $\underline{V}$ not including $\bar{v}_1$ and one other row $\bar{v}_i$, $2 \leq i \leq m$. The matrix $\underline{V}_{1i}$ is of the form

$$\underline{V}_{1i} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}_{(m-2)x(m-1)}.$$

Obviously a basis of all solutions to

$$\underline{V}_{1i}\,\bar{b}_\ell^T = \bar{0}^T$$

is

$$\bar{b}_\ell = (0, 0, \ldots, 0, \underset{\substack{\uparrow \\ \text{ith} \\ \text{column}}}{1}, 0, \ldots, 0) \qquad (C.3.1)$$

and the only way for $\bar{b}_\ell$ to satisfy

$$\underline{V}\bar{b}_\ell^T \geq \bar{0}^T$$

is for

$$\left(-\frac{a_i}{a_1}\right) \geq 0 \qquad (C.3.2)$$

to be true. There are s-1 distinct vectors which can be generated and tested in this manner and each of these vectors corresponds to a constraint like that in Inequality C.3.2. Therefore for Case I the number of PLD's contributed is equal to the number of pairs (a_1, a_i), $i = 2, 3, \ldots, m$, of components which are of opposite sign plus the number of components a_i, $i = 2, 3, \ldots, m$, which are zero. It turns out that a zero component a_i, $i = 2, 3, \ldots, m$, produces an extremal vector in S_b whether Case I or Case II is being considered; furthermore, a particular zero component produces the same basis vector for both cases.

Case II: Consider the selection of a set of m - 2 rows from $\underline{V}$ including $\bar{v}_1$ but not including two other rows $\bar{v}_j$ and $\bar{v}_k$ for $2 \leq j \leq m$, $2 \leq k \leq m$, and $j \neq k$. The matrix $\underline{V}_{jk}$ is of the form shown in Equation C.3.3.

$$\underline{V}_{jk} = \begin{bmatrix} -\frac{a_2}{a_1} & -\frac{a_3}{a_1} & \cdots & -\frac{a_{i-1}}{a_1} & -\frac{a_i}{a_1} & -\frac{a_{i+1}}{a_1} & \cdots & -\frac{a_{k-1}}{a_1} & -\frac{a_k}{a_1} & -\frac{a_{k+1}}{a_1} & \cdots & -\frac{a_m}{a_1} \\ 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

$(m-2)\times(m-1)$

(C.3.3)

From this equation it can be seen that if $\bar{b}$ is a solution to

$$\underline{V}_{jk}\bar{b}^T = \bar{0}^T$$

then the jth and kth components of $\bar{b}$ must satisfy

$$\left(-\frac{a_i}{a_1}\right) b_i + \left(-\frac{a_k}{a_1}\right) b_k = 0 \quad .$$

All other components must be zero. Assume for the present that $a_i \neq 0$, then

$$\begin{bmatrix} b_i \\ \\ b_k \end{bmatrix} = \begin{bmatrix} -\frac{a_k}{a_i} \\ \\ 1 \end{bmatrix} \zeta,$$

for every scalar ζ, will permit writting a basis vector

$$\bar{b}_k = (0, 0, \ldots, 0, -\frac{a_k}{a_i}, 0, \ldots, 0, 1, 0, \ldots, 0) \quad . \qquad (C.3.4)$$

ith position; kth position

Notice that

$$\bar{v}_1 \bar{b}_k^T = 0$$

but that, in order for $\bar{b}_k$ to satisfy

$$\underline{V}\bar{b}_k^T \geq \bar{0}^T \quad ,$$

the inequality

$$\left(-\frac{a_k}{a_i}\right) \geq 0 \qquad (C.3.5)$$

must be true. If this inequality is true then $\bar{b}_k$ is an element of S_b.

If $a_k = 0$ then $\bar{b}_k$ is an element of S_b but it has already been considered in Case I.

If it is assumed that $a_k \neq 0$ rather than $a_i \neq 0$, then

$$\bar{b}_k = (0, 0, \ldots, 0, 1, 0, \ldots, 0, -\frac{a_i}{a_k}, 0, \ldots, 0) \qquad (C.3.6)$$

is an element of S_b if

$$\left(-\frac{a_i}{a_k}\right) \geq 0$$

is true. This is the same constraint on the pair (a_i, a_k) as is imposed by Inequality C.3.5, however the basis vectors given by Equation C.3.4 and C.3.6 are not the same if the magnitudes of a_i and a_k are not equal. Thus one selection of m - 2 rows may produce two different basis vectors; however the two vectors are related by a constant. Therefore, regardless of how the trial basis $\bar{b}_k$ is computed, the pair (a_i, a_k) must satisfy Inequality C.3.5 in order for $\bar{b}_k$ to be an element of S_b. Notice that, if $a_i = 0$ for the $\bar{b}_k$ given by Equation C.3.6, the resulting basis has already been considered in Case I.

Therefore for Case II an element of S_b, which is not considered by Case I, is generated by every pair (a_i, a_k), $2 \leq i \leq m$, $2 \leq k \leq m$, $i \neq k$, where a_i and a_k are of opposite sign.

The net result of Case I and Case II is that a unique element of S_b is generated for every pair (a_i, a_k) where a_i and a_k are of opposite sign and for every element $a_i = 0$. This means that the total number of elements in S_b can be determined. It can be shown that no two elements in S_b produce the same element in S_ϕ.

C.4 Uniqueness of the Elements in S_ϕ for an m x 1 Matrix. The

elements in S_b from Case I above produce elements of S_ϕ of the form

$$\bar{\phi}_{\ell_1}^T = \underline{X}\bar{b}_{\ell_1} = \begin{bmatrix} -\frac{a_j}{a_1} \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \text{jth position} \\ \\ \\ \\ \end{matrix}, \quad a_1 \neq 0, \; 2 \leq j \leq m, \tag{C.4.1}$$

where $\underline{X}$ is given by Equation C.2.1 and $\bar{b}_{\ell_1}$ by Equation C.3.1. For Case II the elements of S_ϕ are given by

$$\bar{\phi}_{\ell_2}^T = \underline{X}\bar{b}_{\ell_2}^T \begin{bmatrix} 0 \\ \vdots \\ 0 \\ -\frac{a_k}{a_i} \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \\ \\ \\ \text{ith position} \\ \\ \\ \\ \text{kth position} \\ \\ \\ \end{matrix}, \quad \begin{matrix} a_k \neq 0, \; a_i \neq 0, \\ 2 \leq k \leq m, \\ 2 \leq i \leq m, \end{matrix} \tag{C.4.2}$$

where $\overline{b}_{\ell_2}$ is given by Equation C.3.4. Since no two vectors $\overline{b}_{\ell_1}$ in S_b have their elements "1" in the same position, no two vectors $\overline{\phi}_{\ell_1}$ given by Equation C.4.1 are equal. Similarly, no two vectors $\overline{\phi}_{\ell_2}$ are equal because every $\overline{b}^T_{\ell_2}$ in S_b corresponds to a different pair (a_i, a_k). Notice that, since $a_k \neq 0$ in Equation C.4.2, $\overline{\phi}_{\ell_1} \neq \overline{\phi}_{\ell_2}$. Therefore elements $\overline{\phi}_\ell$ in S_ϕ are unique and can be easily determined. Corresponding to every pair (a_i, a_k) of opposite sign, an element $\overline{\phi}_\ell$ in S_ϕ results. This element will have all zero components except for a "+1" component in positions i and k. Also, every zero element a_i produces an element $\overline{\phi}_\ell$ in S_ϕ having all zero components except for a "+1" in the ith position.

APPENDIX D

COMPUTER PROGRAMS TO SEARCH FOR THE REDUNDANT TLU OUTPUT VECTORS $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_\gamma$

D.1 Introduction. Contained in this appendix are two computer programs in FORTRAN IV language which can be used to search for the vectors $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_\gamma$. These programs are basically the same with the difference being in the criteria used to select a vector $\bar{c}_i$. The program presented in Section D.2 searches for the vectors $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_{\gamma-1}$, while the one in Section D.3 searches for $\bar{c}_\gamma$. In Section D.4 the time and memory limitations of these programs are discussed.

In the flow-charts for these programs the diamond shaped symbols signify decision steps. If the answer to the question asked or implied is "yes," the logic flow is out the right or left side of the symbol. If the answer is "no," the flow is out the bottom. The oval shaped symbols signify input or output steps and are explained by COMMENT or FORMAT statements in the listings. Contained within the ovals for data inputs are numbers which correspond to the "location in sequence" shown in Tables D.2.1 and D.3.1 which describe the sequencing of the input data. In order to make the programs self-explanatory, COMMENT statements are inserted in the listings in locations which correspond roughly to the inputs to blocks in the flow-charts.

In both of these programs the test for realizability is performed according to the theory of Appendix A. In Section 4.2 the realizability

requirements on $\bar{c}_i$ are discussed and notation is introduced which is used in the flow-charts and the COMMENT statements in the listings. It should be pointed out that the notation in the listing (except in COMMENT statements) does not necessarily correspond to the notation used in the remainder of the work presented here.

D.2 The Program to Search for $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_{\gamma-1}$. The synthesis algorithm in Chapter IV requires that each vector $\bar{c}_i$, $i < \gamma$, be selected such that the number of unique constraints on $\bar{c}_{i+1}$ is minimized. This section contains the flow-chart and listing for a digital computer program which searches for such a vector. Definition 4.6.1 and Definition 4.6.2 provide the criteria for the selection of $\bar{c}_i$, $2 \leq i < \gamma$, and $\bar{c}_1$ for $\gamma > 1$ respectively.

Table D.2.1 describes the sequencing of the input data and gives the FORMAT. The flow-chart is in Figure D.2.1 and the listing follows.

TABLE D.2.1

DESCRIPTION OF INPUT DATA FOR THE PROGRAM TO SEARCH FOR $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_{\gamma-1}$

Location in Sequence	Number of Cards	Description of Contents	Fortran Variables	Format
1	1	Dimension of the pattern space, the number of pattern space vectors, the number of pattern space vectors with a +1 network output, the number of columns for the matrices in the third and fourth READ statements, the value of μ.	ID, NOPAT, NOPOS, IS, XMU	415, F5.0
2	NOPAT	The augmented pattern space vectors. (one vector per card).	A(I,J)	43F3.0
3	IS	Case I: Searching for $\bar{c}_1$. The columns of the matrix $\underline{D}'_{\gamma 1}$ (one column per card). Case II: Searching for $\bar{c}_i$, $2 \leq i < \gamma$. The columns of the matrix $\underline{D}'_{\gamma i}\underline{\Lambda}^T_{\gamma(i-1)}$ (one column per card).	D(I,J)	43F3.0
4	IS	Case I: Searching for $\bar{c}_1$. The columns of the matrix $\underline{D}'_{\gamma 2}$ (one column per card). Case II: Searching for $\bar{c}_2$. The columns of the matrix $\underline{D}'_{\gamma(i+1)}\underline{\Lambda}^T_{\gamma(i-1)}$ (one column per card).	DD(I,J)	43F3.0

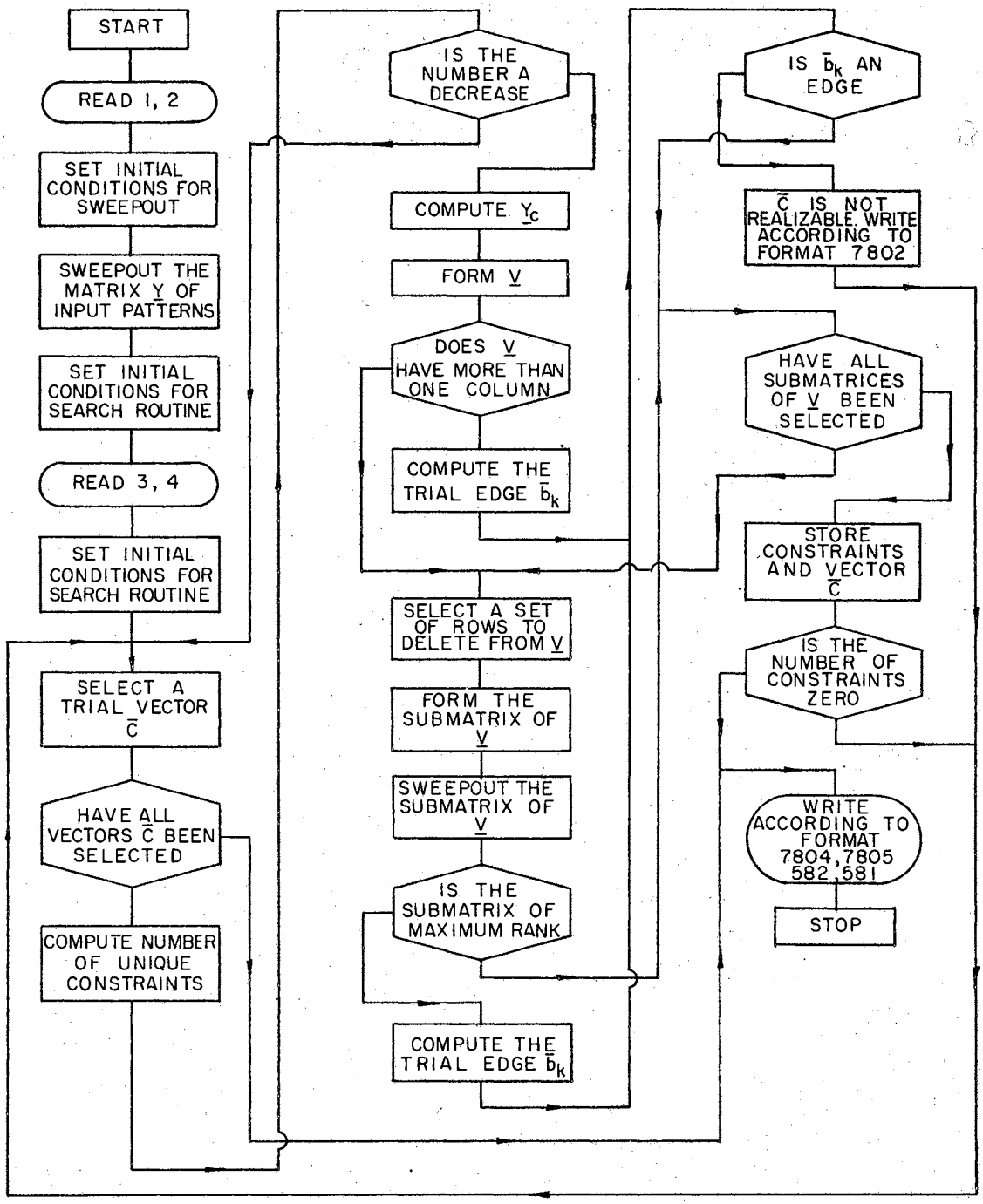


Figure D.2.1. Flow-Chart for the Program to Search for $\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_{\gamma-1}$

```
C      THIS PROGRAM SEARCHES FOR A REDUNDANT TLU OUTPUT VECTOR
C      ACCORDING TO DEFINITION 4.6.1 OR 4.6.2.
       DIMENSION A(5,8),AA(8),AAA(5,8),INDPA(8),IDPA(5),C(9),CL(9),V(8,5)
      1,P(4,5),PP(5),B(5),D(120,9),DD(120,9),IP(120),IN(120),CN(50,9),PAI
      1R(9),CNL(50,9),CTRIV(9)
       LOGICAL ZERO,POS,POS1,AN,BNC(9),BNR(9),BC(9),BR(9)
C      READ INPUT DATA FOR SEQUENCE LOCATIONS 1,2,3
       READ(5,100) ID,NOPAT,NOPOS,IS,NORPT,XMU
100    FORMAT (5I5,F5.0)
       READ(5,100) ID,NOPAT,NOPOS,IS,XMU
100    FORMAT (4I5,F5.0)
       IDP1 = ID + 1
       ICOL = NOPAT
       IROW = IDP1
       DO 6000 J = 1,ICOL
6000   READ(5,101)(A(I,J),I = 1,IROW)
C      SET INITIAL CONDITIONS FOR SWEEPOUT OF MATRIX Y
       IDPA1 = 0
       INDPA1 = 0
       IROW1 = IROW - 1
       L = 1
       JJ = 1
C      SWEEPOUT MATRIX Y
1000   IT = 0
1001   IT = IT + 1
       IF(A(L,JJ)) 1003,1002,1003
1002   IF(IT - IROW + L - 1) 1004,1005,1005
1004   DO 1006 J = 1,ICOL
1006   AA(J) = A(L,J)
       DO 1007 L1 = L,IROW1
       DO 1007 J = 1,ICOL
1007   A(L1,J) = A(L1 + 1,J)
       DO 1008 J = 1,ICOL
1008   A(IROW,J) = AA(J)
       GO TO 1001
1005   INDPA1 = INDPA1 + 1
       INDPA(INDPA1) = JJ
       IF(JJ - ICOL) 1009,1010,1010
1009   JJ = JJ + 1
       GO TO 1000
1003   IDPA1 = IDPA1 + 1
       IDPA(IDPA1) = JJ
       DO 1011 L1 = 1,IROW
       IF(L1 - L) 1012,1011,1012
1012   CONST = A(L1,JJ)/A(L,JJ)
       DO 1021 J = JJ,ICOL
1021   A(L1,J) = A(L1,J) - CONST*A(L,J)
1011   CONTINUE
       DIV = A(L,JJ)
       DO 1013 J = JJ,ICOL
1013   A(L,J) = A(L,J)/DIV
       IF(L - IROW) 1014,1015,1015
1014   IF(JJ - ICOL) 1017,1010,1010
1017   JJ = JJ + 1
       L = L + 1
       GO TO 1000
1015   JJ = JJ + 1
       DO 1016 J = JJ,ICOL
       INDPA1 = INDPA1 + 1
1016   INDPA(INDPA1) = J
C      SET INITIAL CONDITIONS
       N = INDPA1
```

```
      M = ICOL
      IATOT = 2
      DO 37 K = 2,M
37    IATOT = 2*IATOT
      N1 = N - 1
      N2 = N - 2
      MFAC = 1
      DO 1 M1 = 1,M
1     MFAC = MFAC*M1
      N1FAC = 1
      IF(N1) 300,300,301
301   DO 2 N11 = 1,N1
2     N1FAC = N1FAC*N11
300   MNFAC = 1
      MN = M - N1
      DO 3 MN1 = 1,MN
3     MNFAC = MNFAC*MN1
      IR = MFAC/(N1FAC*MNFAC)
      NOBNO = M
      NOBTK = N - 1
      MM1 = NOBNO + 1
      NPATP1 = NOPAT + 1
      C(NPATP1) = XMU
C     READ INPUT DATA FOR SEQUENCE LOCATIONS 3,4
      DO 6001 I = 1,IS
6001  READ(5,101)(D(I,J),J = 1,NPATP1)
      DO 590 I = 1,IS
590   READ(5,101) (DD(I,J),J = 1,NPATP1)
101   FORMAT(43F3.0)
C     SET INITIAL CONDITIONS
      NPOSP1 = NOPOS + 1
      DO 403 I = 1,NOPOS
403   CTRIV(I) = +1.0
      DO 404 I = NPOSP1,NOPAT
404   CTRIV(I) = -1.0
      CTRIV(NPATP1) = XMU
      WRITE (6,7800)
7800  FORMAT (5X,9HITERATION,10X,6HRESULT/)
      S = IS
      IF(S - 4.) 900,900,901
900   FL = S
      GO TO 902
901   FL = (S/2.0)**2
902   DO 7000 I = 2,NPATP1
      BC(I) = .FALSE.
7000  BR(I) = .FALSE.
      BC(1) = .FALSE.
      BR(1) = .TRUE.
      ITT = 0
C     COMPUTE VECTOR C AND TEST TO SEE IF ALL HAVE BEEN TRIED
7001  ITT = ITT + 1
      IF(ITT - 1) 7003,7003,304
304   DO 7002 I = 2,NPATP1
      BR(I) = BC(I).AND.BR(I - 1)
      BC(I)=((.NOT.BC(I)).AND.BR(I-1)).OR.(BC(I).AND.(.NOT.BR(I-1)))
      IF(BR(I)) GO TO 7002
      GO TO 7003
7002  CONTINUE
      GO TO 7004
7003  DO 7005 I = 2,NPATP1
      IF(BC(I)) GO TO 7007
      GO TO 7006
```

```
7006  C(I - 1) = -1.0
      GO TO 7005
7007  C(I - 1) = 1.0
7005  CONTINUE
      DO 402 I = 1,NOPAT
      IF(C(I) - CTRIV(I)) 400,402,400
402   CONTINUE
      GO TO 7001
400   DO 6005 I = 1,NOPAT
      IF(C(I) + CTRIV(I)) 401,6005,401
6005  CONTINUE
      GO TO 7001
C     COMPUTE THE NUMBER OF UNIQUE CONSTRAINTS ON THE VECTOR C(I+1) FOR
C     THE GIVEN VALUE OF C(I) AND SEE IF IT IS A DECREASE
401   IIN = 0
      KF = 0
      IIP = 0
      DO 7008 I = 1,IS
      ALPHA = 0.0
      DO 7009 J = 1,NPATP1
7009  ALPHA = ALPHA + C(J)*D(I,J)
      IF(ALPHA) 7010,7011,7012
7010  IIN = IIN + 1
      IN(IIN) = I
      IF(IIP) 500,500,501
501   DO 502 L = 1,IIP
      IIIP = IP(L)
      DO 503 J = 1,NPATP1
503   PAIR(J) = DD(IIIP,J) + DD(I,J)
      IF(KF) 505,505,504
504   DO 506 ICN = 1,KF
      DO 507 J = 1,NPATP1
      IF(CN(ICN,J) - PAIR(J)) 4000,507,4000
507   CONTINUE
      GO TO 502
4000  IRTST = 0
      DO 4001 J = 1,NPATP1
      IF(PAIR(J)) 4002,4003,4002
4003  IF(CN(ICN,J)) 506,4001,506
4002  RTEST = (CN(ICN,J))/PAIR(J)
      IF(RTEST) 4004,506,4004
4004  IF(IRTST) 4005,4005,4006
4005  IRTST = 1
      RTSTR = RTEST
      GO TO 4001
4006  IF(RTSTR - RTEST) 506,4001,506
4001  CONTINUE
      GO TO 502
506   CONTINUE
505   KF = KF + 1
      DO 508 J = 1,NPATP1
508   CN(KF,J) = PAIR(J)
502   CONTINUE
      GO TO 500
7012  IIP = IIP + 1
      IP(IIP) = I
      IF(IIN) 500,500,510
510   DO 511 L = 1,IIN
      IIIN = IN(L)
      DO 512 J = 1,NPATP1
512   PAIR(J) = DD(IIIN,J) + DD(I,J)
      IF(KF) 514,514,513
```

```
513   DO 515 ICN = 1,KF
      DO 516 J = 1,NPATP1
      IF(CN(ICN,J) - PAIR(J)) 4010,516,4010
516   CONTINUE
      GO TO 511
4010  RTST = 0
      DO 4011 J = 1,NPATP1
      IF(PAIR(J)) 4012,4013,4012
4013  IF(CN(ICN,J)) 515,4011,515
4012  RTEST = (CN(ICN,J))/PAIR(J)
      IF(RTEST) 4014,515,4014
4014  IF(IRTST) 4015,4015,4016
4015  IRTST = 1
      RTSTR = RTEST
      GO TO 4011
4016  IF(RTSTR - RTEST) 515,4011,515
4011  CONTINUE
      GO TO 511
515   CONTINUE
514   KF = KF + 1
      DO 517 J = 1,NPATP1
517   CN(KF,J) = PAIR(J)
511   CONTINUE
      GO TO 500
7011  IF(KF) 521,521,520
520   DO 522 ICN = 1,KF
      DO 523 J = 1,NPATP1
      IF(CN(ICN,J) - DD(I,J)) 4020,523,4020
523   CONTINUE
      GO TO 500
4020  IRTST = 0
      DO 4021 J = 1,NPATP1
      IF(DD(I,J)) 4022,4023,4022
4023  IF(CN(ICN,J)) 522,4021,522
4022  RTEST = (CN(ICN,J))/DD(I,J)
      IF(RTEST) 4024,522,4024
4024  IF(IRTST) 4025,4025,4026
4025  IRTST = 1
      RTSTR = RTEST
      GO TO 4021
4026  IF(RTSTR - RTEST) 522,4021,522
4021  CONTINUE
      GO TO 500
522   CONTINUE
521   KF = KF + 1
      DO 524 J = 1,NPATP1
524   CN(KF,J) = DD(I,J)
500   XKF = KF
      IF(XKF - FL) 7008,7030,7030
7008  CONTINUE
C     NEGATE CERTAIN ROWS AND COLUMNS OF SWEPTOUT FORM OF MATRIX Y AND
C     FORM MATRIX V
      DO 2000 I = 2,MM1
      BNC(I) = .FALSE.
2000  BNR(I) = .FALSE.
      BNC(1) = .FALSE.
      BNR(1) = .TRUE.
      GO TO 7031
7030  CONTINUE
      GO TO 7001
7031  DO 7014 I = 1,NOPOS
      DO 7014 J = 1,IDP1
```

```
7014  AAA(J,I) = C(I)*A(J,I)
      DO 7015 I = NPOSP1,NOPAT
      DO 7015 J = 1,IDP1
7015  AAA(J,I) = -C(I)*A(J,I)
      DO 7016 II = 1,IDPA1
      I = IDPA(II)
      IF(AAA(II,I)) 7017,7016,7016
7017  DO 7018 J = 1,NOPAT
7018  AAA(II,J) = -AAA(II,J)
7016  CONTINUE
1010  DO 1018 I1 = 1,IDPA1
      I = IDPA(I1)
      DO 1018 J1 = 1,INDPA1
      J = INDPA(J1)
1018  V(I,J1) = -AAA(I1,J)
      DO 1022 J1 = 1,INDPA1
      J = INDPA(J1)
      DO 1022 J11 = 1,INDPA1
      IF(J11 - J1) 1019,1020,1019
1020  V(J,J11) = 1.0
      GO TO 1022
1019  V(J,J11) = 0.0
1022  CONTINUE
C     COMPUTE TRIAL EDGE FOR TRIVIAL M*1 MATRIX V
      IF(N1) 302,302,303
302   ISET = 1
      B(1) = 1.0
      GO TO 31
303   ISET = 0
C     BEGIN TEST TO SEE IF THERE EXISTS AN EDGE
C     SELECT A SET OF ROWS TO DELETE FROM MATRIX V
4     CONTINUE
2001  ICT = 0
      DO 2002 I = 2,MM1
      BNR(I) = BNC(I).AND.BNR(I-1)
      BNC(I) = ((.NOT.BNC(I)).AND.BNR(I-1)).OR.(BNC(I).AND.(.NOT.BNR(I-1
     1)))
      IF(BNR(I)) GO TO 2002
2003  ICT = ICT + 1
      GO TO 2004
2002  CONTINUE
      GO TO 7019
2004  IF(ICT - NOBTK) 2006,2006,2001
2006  IF(I - MM1) 2013,2001,2001
2013  IBN1 = I + 1
      DO 2008 IBN = IBN1,MM1
      IF(BNC(IBN)) GO TO 2009
      GO TO 2008
2009  ICT = ICT + 1
      IF(ICT - NOBTK) 2008,2008,2001
2008  CONTINUE
      IF(ICT - NOBTK) 2001,2010,2001
2010  CONTINUE
C     FORM THE SUBMATRIX OF MATRIX V
9     L = 0
      DO 10 K = 1,M
      IF(BNC(K + 1)) GO TO 11
      GO TO 10
11    L = L + 1
      DO 36 J = 1,N
36    P(L,J) = V(K,J)
10    CONTINUE
```

```
C     SWEEPOUT THE SUBMATRIX OF MATRIX V AND TEST TO SEE IF IT IS OF
C     MAXIMUM RANK
      ISET = ISET + 1
      IID1 = 0
      JJ = 1
      L = 1
13    IT = 0
14    IT = IT + 1
      IF(P(L,JJ)) 15,16,15
16    IF(IT - N1 + L - 1) 17,18,18
17    DO 34 J = 1,N
34    PP(J) = P(L,J)
      DO 19 L1 = L,N2
      DO 19 J = 1,N
19    P(L1,J) = P(L1 + 1,J)
      DO 35 J = 1,N
35    P(N1,J) = PP(J)
      GO TO 14
18    IID1 = IID1 + 1
      IF(IID1 - 1) 20,20,50
20    IID = JJ
      IF(JJ - N) 22,23,23
22    JJ = JJ + 1
      GO TO 13
15    DO 24 L1 = 1,N1
      IF(L1 - L) 25,24,25
25    CONST = P(L1,JJ)/P(L,JJ)
      DO 33 J = JJ,N
33    P(L1,J) = P(L1,J) - CONST*P(L,J)
24    CONTINUE
      DIV = P(L,JJ)
      DO 32 J = JJ,N
32    P(L,J) = P(L,J)/DIV
      IF(JJ - N) 26,23,23
26    JJ = JJ + 1
      IF(JJ - N) 27,57,57
27    L = L + 1
      GO TO 13
57    IF(IID1 - 1) 60,27,27
60    IID = N
23    IIDL = IID - 1
C     COMPUTE THE TRIAL EDGE B(K)
      IF(IIDL) 62,61,62
62    DO 28 J = 1,IIDL
28    B(J) = -P(J,IID)
61    B(IID) = 1.0
      IF(IID - N) 29,31,31
29    IIDU = IID + 1
      DO 30 J = IIDU,N
30    B(J) = 0.0
C     TEST TO SEE IF B(K) IS AN EDGE
31    IPCNT = 1
      ZERO = .TRUE.
      DO 39 L1 = 1,M
      F = 0.0
      DO 38 J = 1,N
38    F = F + V(L1,J)*B(J)
      IF(F) 40,39,41
40    POS = .FALSE.
      GO TO 42
41    POS = .TRUE.
42    IF(IPCNT - 1) 44,44,43
```

```
44    POS1 = POS
      IPCNT = 2
      ZERO = .FALSE.
      GO TO 39
43    AN = POS.AND.POS1
      IF(AN) GO TO 39
      GO TO 50
39    CONTINUE
46    IF(ZERO) GO TO 50
C     WRITE THAT VECTOR C IS NOT REALIZABLE
      WRITE (6,7802) ITT,KF
7802  FORMAT (7X,I3,14X,I3,50H CONSTRAINTS IS A DECREASE BUT C IS NOT RE
     1ALIZABLE)
      GO TO 7001
C     HAVE ALL SUBMATRICES OF MATRIX V BEEN SELECTED
50    IF(ISET - IR) 4,7019,7019
C     STORE THE SMALLER NUMBER OF CONSTRAINTS, THE VECTOR C, AND THE
C     COEFFICIENTS OF THE CONSTRAINTS
7019  FL = XKF
      KFL = KF
      DO 7020 I = 1,NPATP1
7020  CL(I) = C(I)
      DO 570 I = 1,KF
      DO 570 J = 1,NPATP1
570   CNL(I,J) = CN(I,J)
      WRITE (6,7803) ITT,KF
7803  FORMAT (7X,I3,14X,I3,32H CONSTRAINTS WITH A REALIZABLE C)
C     IS THE NUMBER OF CONSTRAINTS ZERO
      IF(KF) 7001,7004,7001
C     WRITE THE MINIMUM NUMBER OF CONSTRAINTS, THE VECTOR C, AND THE
C     MINIMAL SET OF CONSTRAINT COEFFICIENTS
7004  WRITE (6,7804) KFL
7804  FORMAT(1H0,37HTHE SMALLEST NUMBER OF CONSTRAINTS IS,I5,20H AND THE
     1 VECTOR C IS/)
      WRITE (6,7805) (CL(I),I = 1,NPATP1)
7805  FORMAT (20(2X,F4.1))
      WRITE(6,582)
582   FORMAT(1H1,43HTHE FOLLOWING ROWS CONTAIN THE COEFFICIENTS/52H OF T
     1HE INEQUALITY CONSTRAINTS FOR THE NEXT VECTOR C//)
      DO 580 I = 1,KFL
580   WRITE(6,581)(CNL(I,J),J = 1,NPATP1)
581   FORMAT(1H ,9(2X,E12.5))
      STOP
      END
```

D.3 The Program to Search for $\bar{c}_\gamma$. The synthesis algorithm in Chapter IV requires that, for an assumed value of γ, a vector $\bar{c}_\gamma$ corresponding to a realizable $\bar{c}'_\gamma$ be found such that

$$[\bar{c}_\gamma \mid \mu]\underline{D}''_{\gamma\gamma} \gtrless \bar{0}$$

is satisfied. The appendix contains the flow-chart and listing for a digital computer program to perform a search for a $\bar{c}_\gamma$ satisfying the above requirement or to determine the nonexistence of one. Table D.3.1 describes the input data and its sequencing. The flow-chart is in Figure D.3.1 and the listing follows.

TABLE D.3.1

DESCRIPTION OF INPUT DATA FOR THE PROGRAM TO SEARCH FOR $\overline{c}_Y$

Location in Sequence	Number of Cards	Description of Contents	Fortran Variables	Format
1	1	Dimension of the pattern space, the number of pattern space vectors, the number of pattern space vectors with a +1 network output, the number of columns for the matrix in the third READ statement, the value of μ.	ID, NOPAT, NOPOS IS, XMU	415, F5.0
2	NOPAT	The augmented pattern space vectors. (one vector per card).	A(I,J)	43F3.0
3	IS	The columns of the matrix $\underline{D}''_{YY}$ (one column per card).	D(I,J)	43F3.0

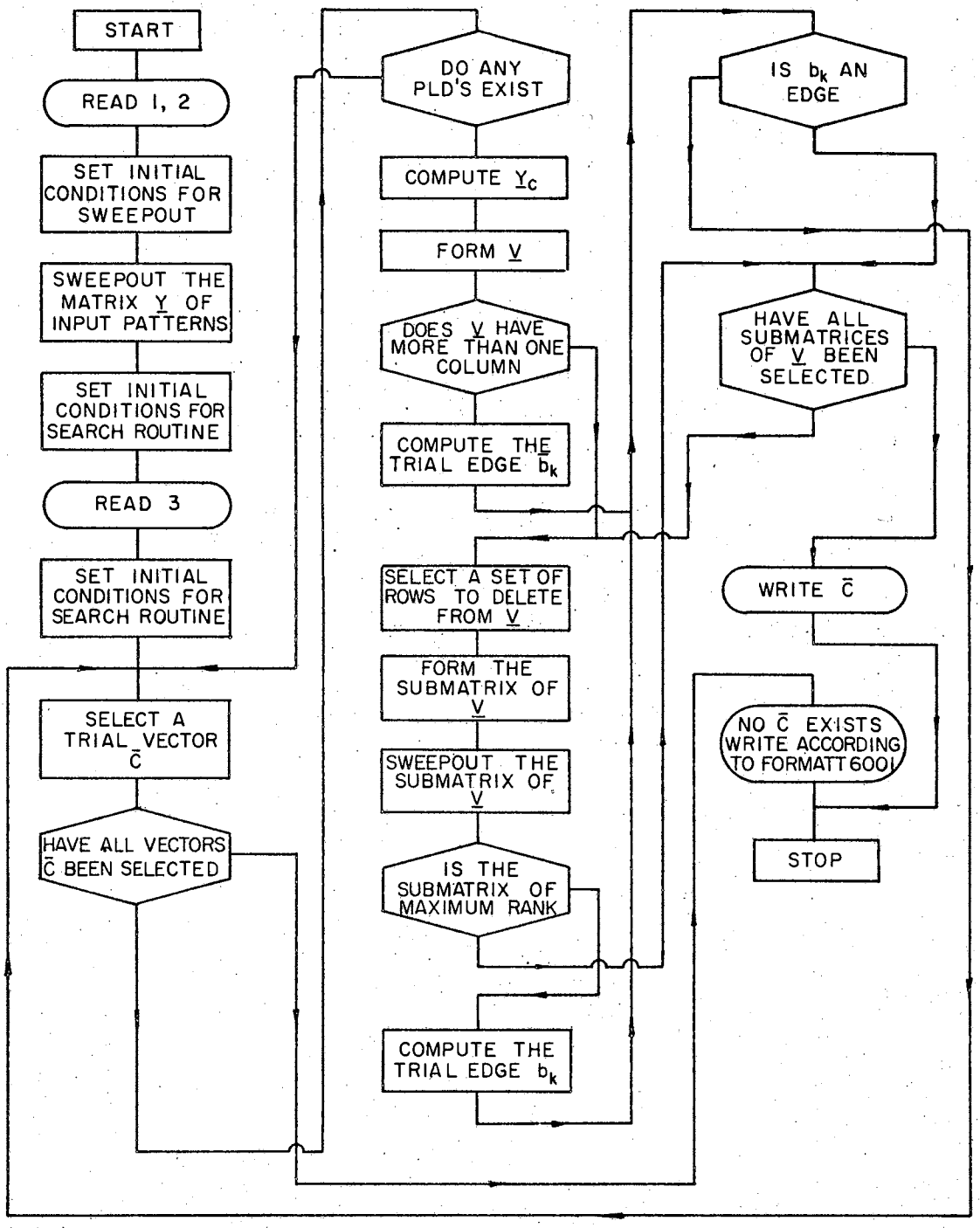


Figure D.3.1. Flow-Chart for the Program to Search for $\bar{c}_Y$

```
C      THIS PROGRAM SEARCHES FOR A REDUNDANT TLU OUTPUT VECTOR FOR
C      WHICH THE SYNTHESIS ALGORITHM TERMINATES.
       DIMENSION A(4,8),AA(8),AAA(4,8),INDPA(8),IDPA(4),C(9),CL(9),V(8,4)
      1,P(3,4),PP(4),B(4),D(62,9),DD(62,9),IP(62),IN(62),CN(10,9),PAIR(9)
      1,CNL(10,9),CTRIV(9)
       LOGICAL ZERO,POS,POS1,AN,BNC(9),BNR(9),BC(9),BR(9)
C      READ INPUT DATA FOR SEQUENCE LOCATIONS 1,2,3
       READ(5,100) ID,NOPAT,NOPOS,IS,NORPT,XMU
100    FORMAT (5I5,F5.0)
       IDP1 = ID + 1
       ICOL = NOPAT
       IROW = IDP1
       DO 6000 J = 1,ICOL
6000   READ(5,101)(A(I,J),I = 1,IROW)
C      SET INITIAL CONDITIONS FOR SWEEPOUT OF MATRIX Y
       IDPA1 = 0
       INDPA1 = 0
       IROW1 = IROW - 1
       L = 1
       JJ = 1
1000   IT = 0
C      SWEEPOUT MATRIX Y
1001   IT = IT + 1
       IF(A(L,JJ)) 1003,1002,1003
1002   IF(IT - IROW + L - 1) 1004,1005,1005
1004   DO 1006 J = 1,ICOL
1006   AA(J) = A(L,J)
       DO 1007 L1 = L,IROW1
       DO 1007 J = 1,ICOL
1007   A(L1,J) = A(L1 + 1,J)
       DO 1008 J = 1,ICOL
1008   A(IROW,J) = AA(J)
       GO TO 1001
1005   INDPA1 = INDPA1 + 1
       INDPA(INDPA1) = JJ
       IF(JJ - ICOL) 1009,1010,1010
1009   JJ = JJ + 1
       GO TO 1000
1003   IDPA1 = IDPA1 + 1
       IDPA(IDPA1) = JJ
       DO 1011 L1 = 1,IROW
       IF(L1 - L) 1012,1011,1012
1012   CONST = A(L1,JJ)/A(L,JJ)
       DO 1021 J = JJ,ICOL
1021   A(L1,J) = A(L1,J) - CONST*A(L,J)
1011   CONTINUE
       DIV = A(L,JJ)
       DO 1013 J = JJ,ICOL
1013   A(L,J) = A(L,J)/DIV
       IF(L - IROW) 1014,1015,1015
1014   IF(JJ - ICOL) 1017,1010,1010
1017   JJ = JJ + 1
       L = L + 1
       GO TO 1000
1015   JJ = JJ + 1
       DO 1016 J = JJ,ICOL
       INDPA1 = INDPA1 + 1
1016   INDPA(INDPA1) = J
C      SET INITIAL CONDITIONS
       N = INDPA1
       M = ICOL
       IATOT = 2
```

```
      DO 37 K = 2,M
37    IATOT = 2*IATOT
      N1 = N - 1
      N2 = N - 2
      MFAC = 1
      DO 1 M1 = 1,M
1     MFAC = MFAC*M1
      N1FAC = 1
      IF(N1) 300,300,301
301   DO 2 N11 = 1,N1
2     N1FAC = N1FAC*N11
300   MNFAC = 1
      MN = M - N1
      DO 3 MN1 = 1,MN
3     MNFAC = MNFAC*MN1
      IR = MFAC/(N1FAC*MNFAC)
      NOBNO = M
      NOBTK = N - 1
      MM1 = NOBNO + 1
      NPATP1 = NOPAT + 1
      C(NPATP1) = XMU
C     READ INPUT DATA FOR SEQUENCE LOCATION 3
      DO 6001 I = 1,IS
6001  READ(5,101)(D(I,J),J = 1,NPATP1)
      DO 590 I = 1,IS
590   READ(5,101) (DD(I,J),J = 1,NPATP1)
101   FORMAT(43F3.0)
C     SET INITIAL CONDITIONS
      NPOSP1 = NOPOS + 1
      DO 403 I = 1,NOPOS
403   CTRIV(I) = +1.0
      DO 404 I = NPOSP1,NOPAT
404   CTRIV(I) = -1.0
      CTRIV(NPATP1) = XMU
      WRITE (6,7800)
7800  FORMAT (5X,9HITERATION,10X,6HRESULT/)
      S = IS
      IF(S - 4.) 900,900,901
900   FL = S
      GO TO 902
901   FL = (S/2.0)**2
902   DO 7000 I = 2,NPATP1
      BC(I) = .FALSE.
7000  BR(I) = .FALSE.
      BC(1) = .FALSE.
      BR(1) = .TRUE.
      ITT = 0
C     COMPUTE VECTOR C AND TEST TO SEE IF ALL HAVE BEEN TRIED
7001  ITT = ITT + 1
      IF(ITT - 1) 7003,7003,304
304   DO 7002 I = 2,NPATP1
      BR(I) = BC(I).AND.BR(I - 1)
      BC(I)=((.NOT.BC(I)).AND.BR(I-1)).OR.(BC(I).AND.(.NOT.BR(I-1)))
      IF(BR(I)) GO TO 7002
      GO TO 7003
7002  CONTINUE
      GO TO 7004
7003  DO 7005 I = 2,NPATP1
      IF(BC(I)) GO TO 7007
      GO TO 7006
7006  C(I - 1) = -1.0
      GO TO 7005
```

```
7007  C(I - 1) = 1.0
7005  CONTINUE
      DO 402 I = 1,NOPAT
      IF(C(I) - CTRIV(I)) 400,402,400
402   CONTINUE
      GO TO 7001
400   DO 6005 I = 1,NOPAT
      IF(C(I) + CTRIV(I)) 401,6005,401
6005  CONTINUE
      GO TO 7001
C     TEST TO SEE IF THE VECTOR C SATISFIES THE INEQUALITY
401   IN = 0
      IZ = 0
      IP = 0
      DO 7008 I = 1,IS
      ALPHA = 0.0
      DO 7009 J = 1,NPATP1
7009  ALPHA = ALPHA + C(J)*D(I,J)
      IF(ALPHA) 7010,7030,7012
7010  IN = IN +1
      GO TO 7013
7011  IZ = IZ + 1
      GO TO 7013
7012  IP = IP + 1
7013  KF = IP*IN + IZ
      IF(KF) 7008,7008,7030
7008  CONTINUE
C     NEGATE CERTAIN ROWS AND COLUMNS OF SWEPTOUT FORM OF MATRIX Y AND
C     FORM MATRIX V
      DO 2000 I = 2,MM1
      BNC(I) = .FALSE.
2000  BNR(I) = .FALSE.
      BNC(1) = .FALSE.
      BNR(1) = .TRUE.
      GO TO 7031
7030  CONTINUE
      GO TO 7001
7031  DO 7014 I = 1,NOPOS
      DO 7014 J = 1,IDP1
7014  AAA(J,I) = C(I)*A(J,I)
      DO 7015 I = NPOSP1,NOPAT
      DO 7015 J = 1,IDP1
7015  AAA(J,I) = -C(I)*A(J,I)
      DO 7016 II = 1,IDPA1
      I = IDPA(II)
      IF(AAA(II,I)) 7017,7016,7016
7017  DO 7018 J = 1,NOPAT
7018  AAA(II,J) = -AAA(II,J)
7016  CONTINUE
1010  DO 1018 I1 = 1,IDPA1
      I = IDPA(I1)
      DO 1018 J1 = 1,INDPA1
      J = INDPA(J1)
1018  V(I,J1) = -AAA(I1,J)
      DO 1022 J1 = 1,INDPA1
      J = INDPA(J1)
      DO 1022 J11 = 1,INDPA1
      IF(J11 - J1) 1019,1020,1019
1020  V(J,J11) = 1.0
      GO TO 1022
1019  V(J,J11) = 0.0
1022  CONTINUE
```

```
C     COMPUTE TRIAL EDGE FOR TRIVIAL M*1 MATRIX V
      IF(N1) 302,302,303
302   ISET = 1
      B(1) = 1.0
      GO TO 31
303   ISET = 0
C     BEGIN TEST TO SEE IF THERE EXISTS AN EDGE
C     SELECT A SET OF ROWS TO DELETE FROM MATRIX V
4     CONTINUE
2001  ICT = 0
      DO 2002 I = 2,MM1
      BNR(I) = BNC(I).AND.BNR(I-1)
      BNC(I) = ((.NOT.BNC(I)).AND.BNR(I-1)).OR.(BNC(I).AND.(.NOT.BNR(I-1
     1)))
      IF(BNR(I)) GO TO 2002
2003  ICT = ICT + 1
      GO TO 2004
2002  CONTINUE
      GO TO 7019
2004  IF(ICT - NOBTK) 2006,2006,2001
2006  IF(I - MM1) 2013,2001,2001
2013  IBN1 = I + 1
      DO 2008 IBN = IBN1,MM1
      IF(BNC(IBN)) GO TO 2009
      GO TO 2008
2009  ICT = ICT + 1
      IF(ICT - NOBTK) 2008,2008,2001
2008  CONTINUE
      IF(ICT - NOBTK) 2001,2010,2001
2010  CONTINUE
C     FORM THE SUBMATRIX OF MATRIX V
9     L = 0
      DO 10 K = 1,M
      IF(BNC(K + 1)) GO TO 11
      GO TO 10
11    L = L + 1
      DO 36 J = 1,N
36    P(L,J) = V(K,J)
10    CONTINUE
C     SWEEPOUT THE SUBMATRIX OF MATRIX V AND TEST TO SEE IF IT IS OF
C     MAXIMUM RANK
      ISET = ISET + 1
      IID1 = 0
      JJ = 1
      L = 1
13    IT = 0
14    IT = IT + 1
      IF(P(L,JJ)) 15,16,15
16    IF(IT - N1 + L - 1) 17,18,18
17    DO 34 J = 1,N
34    PP(J) = P(L,J)
      DO 19 L1 = L,N2
      DO 19 J = 1,N
19    P(L1,J) = P(L1 + 1,J)
      DO 35 J = 1,N
35    P(N1,J) = PP(J)
      GO TO 14
18    IID1 = IID1 + 1
      IF(IID1 - 1) 20,20,50
20    IID = JJ
      IF(JJ - N) 22,23,23
22    JJ = JJ + 1
```

```
      GO TO 13
15    DO 24 L1 = 1,N1
      IF(L1 - L) 25,24,25
25    CONST = P(L1,JJ)/P(L,JJ)
      DO 33 J = JJ,N
33    P(L1,J) = P(L1,J) - CONST*P(L,J)
24    CONTINUE
      DIV = P(L,JJ)
      DO 32 J = JJ,N
32    P(L,J) = P(L,J)/DIV
      IF(JJ - N) 26,23,23
26    JJ = JJ + 1
      IF(JJ - N) 27,57,57
27    L = L + 1
      GO TO 13
57    IF(IID1 - 1) 60,27,27
60    IID = N
23    IIDL = IID - 1
C     COMPUTE THE TRIAL EDGE B(K)
      IF(IIDL) 62,61,62
62    DO 28 J = 1,IIDL
28    B(J) = -P(J,IID)
61    B(IID) = 1.0
      IF(IID - N) 29,31,31
29    IIDU = IID + 1
      DO 30 J = IIDU,N
30    B(J) = 0.0
C     TEST TO SEE IF B(K) IS AN EDGE
31    IPCNT = 1
      ZERO = .TRUE.
      DO 39 L1 = 1,M
      F = 0.0
      DO 38 J = 1,N
38    F = F + V(L1,J)*B(J)
      IF(F) 40,39,41
40    POS = .FALSE.
      GO TO 42
41    POS = .TRUE.
42    IF(IPCNT - 1) 44,44,43
44    POS1 = POS
      IPCNT = 2
      ZERO = .FALSE.
      GO TO 39
43    AN = POS.AND.POS1
      IF(AN) GO TO 39
      GO TO 50
39    CONTINUE
46    IF(ZERO) GO TO 50
      GO TO 7001
C     HAVE ALL SUBMATRICES OF MATRIX V BEEN SELECTED
50    IF(ISET - IR) 4,7019,7019
C     WRITE THE VECTOR C
7019  WRITE (6,7804)
7804  FORMAT(1H0,57HTHE ASSUMED VALUE OF GAMMA IS CORRECT AND THE VECTOR
     1 C IS/)
      WRITE (6,7805) (CL(I),I = 1,NPATP1)
      GO TO 6000
7805  FORMAT (20(2X,F4.1))
7004  WRITE (6,6001)
6001  FORMAT(1H0,41HTHERE EXISTS NO VECTOR C AND THE ASSUMED /28H VALUE
     1OF GAMMA IS INCORRECT)
6000  STOP
      END
```

D.4 Limiting Factors for Time and Memory. The running characteristics of these programs are highly dependent upon specific problems. However, certain limitations can be pointed out. Perhaps the greatest limitation is caused by the number of components in the vector $\bar{c}_\gamma$. In the program this number is called NPATP1 which is equal to NOPAT+1 where NOPAT is the number of augmented, input pattern space vectors $\bar{y}_i$. Each time that a vector $\bar{c}_i$ is selected and satisfies the inequality

$$[\bar{c}_i \,\vdots\, \mu]D''_{\gamma\gamma} \gtrless \bar{0} ,$$

it must be tested for realizability. In this test the vectors $\bar{y}_i$ (or $-\bar{y}_i$) form a matrix which must be tested for PLD's. From an examination of the theory of Appendix A it can be seen that, for a realizable $\bar{c}_\gamma$, a major operation in the test must be performed

$$\begin{pmatrix} \text{NOPAT} \\ \text{NOPAT-ID-1} \end{pmatrix}$$

times where ID is the dimension of the pattern space and ID+1 is the size of each vector $\bar{y}_i$. If $\bar{c}_\gamma$ is not realizable, all of these operations will not be performed. Nevertheless, the size of NOPAT determines to a large degree the running time.

Another limitation is that the number NOPAT must be greater than ID+1. This restriction is basic to the theory as developed in Appendix A.

The major portion of the data storage for these programs is for the matrices read in the sequence locations 3 and 4. For the program in Section D.2, an increase in γ by one causes the number of columns in $\underline{D}'_{\gamma i}$ and $\underline{D}''_{\gamma(i+1)}$ to increase by the number of columns in $\underline{R}_1$. Similarly for

the altered forms of these matrices appearing in Section 4.7 the number is increased by twice the number of columns in $\underline{R}_1^a$. The change in the size of $\underline{D}''_{\gamma\gamma}$ is more difficult to determine; however, there should be no significant problems with storage here since $\underline{D}''_{\gamma\gamma}$ contains only the unique columns of $\underline{D}'_{\gamma\gamma}\underline{\Lambda}^T_{\gamma(\gamma-1)}$.

VITA

John L. Youngblood

Candidate for the Degree of

Doctor of Philosophy

Thesis: A STUDY OF THE APPLICABILITY AND SYNTHESIS OF REDUNDANT, THRESHOLD LOGIC DECISION-MAKERS

Major Field: Electrical Engineering

Biographical:

Personal Data: Born on March 14, 1941 and reared at Cayuga, Texas, the son of J. Douglas and Hessie B. Youngblood.

Education: Attended Cayuga Public School through May, 1957 and graduated from R. L. Paschal High School in Fort Worth, Texas in 1959; received the Bachelor of Science degree in Electrical Engineering from Arlington State College in May, 1963; received the Master of Science degree in Electrical Engineering from Oklahoma State University in May, 1965; completed requirements for the Doctor of Philosophy degree in May, 1967.

Professional Experience: Employed by the Fort Worth Division of General Dynamics Corporation during the summer of 1963. Employed by the School of Electrical Engineering at Oklahoma State University as a graduate research assistant from September, 1965 through September, 1966. Employed by the Fort Worth Division of General Dynamics Corporation since October, 1966.